Modul Praktikum Mobile Programming



Di Susun Oleh : Muhammad Fahmi

Informatika (S1) Universitas Nusa Mandiri 2021

Kata Pengantar

Puji Dan Syukur Senantiasa Kami Panjatkan Kepada Kehadirat Allah Swt Yang Telah Melimpahkan Segala Nikmat-Nya. Dengan Limpahan Rahmat-Nya, Penulis Berhasil Menyelesaikan Modul Ajar Berjudul Mobile Programming Ini Tanpa Mengalami Kendala Yang Berarti. Tujuan Penyusunan Modul Ini Adalah Untuk Memberikan Kemudahan Kepada Para Mahasiswa Dalam Memahami Dunia Mobile Programming Dan Praktik Pembuatan Api. Modul Ini Dirancang Dalam Tahapan Agar Pemula Dapat Dengan Lebih Mudah Mempelajari Mobile Programming Serta Mengembangkan Aplikasi Mobile Yang Khusus Ditujukan Untuk Sistem Operasi Android.

Keberhasilan Penyusunan Modul Ini Tidak Hanya Hasil Dari Usaha Penulis Semata, Tetapi Juga Melibatkan Berbagai Pihak Yang Turut Serta Membantu Dan Memberikan Dukungan, Baik Secara Moril Maupun Materi, Demi Kesuksesan Penulisan Modul Ini. Oleh Karena Itu, Penulis Mengucapkan Terima Kasih Sebesar-Besarnya Kepada Semua Pihak Yang Telah Memberikan Dukungan.

Tentu Saja, Modul Ini Tidak Luput Dari Kekurangan. Selalu Terbuka Ruang Untuk Perbaikan. Kritik, Saran, Dan Masukan Dari Pembaca Sangat Kami Harapkan Guna Penyempurnaan Di Masa Mendatang.

Jakarta, 09 Juli 2021

Muhammad Fahmi

Aplikasi Mobile dengan Flutter

Menyiapkan perangkat

Install Git

Buka laman https://git-scm.com/downloads, kemudian klik tombol download

🚸 gitlocal-b	ranching-on-the-cheap	Q. Search entire site.
About Documentation Downloads	Downloads	
GUI Clients Logos	🍯 Mac OS X 🛛 🐉 Windows	Litest source Release 2.29.2 Release Notes (2020-10-29)
The entire Pro Git book	Older releases are available and the Git source repository is on GitHub.	Download 2:29.2 for Windows
written by South Chacon and Bon Straub is available to read unline for fire. Dead tree versions are available on Amazon.com.	GUI Clients	Logos
	Git comes with built-in GUI tools (git-gui, gitk), but there are several third-party tools for users looking for a platform-specific experience.	Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

Kemudian lakukan installasi git dari file yang telah diunduh.

Install JDK

JDK (Java Development Kit) adalah sebuah perangkat lunak yang digunakan untuk melakukan proses kompilasi dari kode java ke bytecode yang dapat dimengerti dan dapat dijalankan oleh JRE (Java Runtime Envirotment). JDK wajib terinstall pada komputer yang akan melakukan proses pembuatan aplikasi berbasis java, namun tidak wajib terinstall di komputer yang akan menjalankan aplikasi yang dibangun dengan java.

JDK dapat diunduh pada laman https://jdk.java.net/



Java Development Kit builds, from Oracle

Ready for use: JDK 15, JMC 7

Early access: JDK 17, JDK 16, JMC 8, Lanai, Loom, Metropolis, Panama, & Valhalla

Reference implementations: Java SE 15, 14, 13, 12, 11, 10, 9, 8, & 7



© 2021 Oracle Corporation and/or its officiates Terms of Use · Privacy - Trademarks Pilih JDK 15, kemudian download file zip untuk windows, jika menggunakan windows

jdk.java.net	JDK 15.0.2 Genera	al-Availabili	ty Release
GA Releases JDK 15 JMC 7	This page provides produ Kit, version 15, an imple	uction-ready ope mentation of the	en-source builds of the Java Development e Java SE 15 Platform under the GNU
Early-Access	General Public License, v	version 2, with t	he Classpath Exception.
JDK 17 JDK 16 JMC 8 Lanai	Commercial builds of JDR be found at the Oracle Te	K 15.0.2 from Or echnology Netw	acle, under a non-open-source license, can ork.
Loom	Documentation		
Panama Valhalla	 Features 		
Reference Implementations Java SE 15	Release notesAPI Javadoc		
java SE 14 Java SE 13 Java SE 12	Builds		
Java SE 11	Linux/AArch64	tar.gz (sna256)	170507166 bytes
Java SE 9	Linux/x64	tar.gz (sna256)	195340587
Java SE 8 Java SE 7	macOS/x64	tar.gz (sha256)	192067136
Feedback Report a bug	Windows/x64	Zip (sha256)	195939486

Kemudian extrak berkas file tersebut pada laptop/komputer

avigation Details pane	Extra larg	e itons 📰 Large itons ns 👫 List 📰 Content	Medi EE Detai	um icons + Is - T	Sort by▼	Group by • Add columns •	 ☐ Item chec ☑ File name ☐ Hidden it 	k boxes extension: ems
Panes		Layout				Current view		Show/hid
🗧 🤿 🕆 👖 🕹 Thi	s PC → Local D	Disk (C:) ⇒ Program Files	⇒ Java ≯					
	Name	~		Date modified	ł	Туре	Size	
* Quick access	idk-1	15.0.1		15/09/2020 20).51	File folder		
Desktop	* eper	njdk-15.0.1_windows-x64_	bin zip	13/01/2021 11	.02	WinRAR ZIP archive	191,345 KB	
Downloads								
Bistures	*							
Pictures								
DATA (D-)								
DATA (D:)	*							
DATA (D:) PROJECT	* * *							
DATA (D:) PROJECT Tutorial	* * *							
DATA (D:) PROJECT Tutorial htdocs Tuic OC	* * *							
DATA (D:) PROJECT Tutorial htdocs This PC Dubinstr	* * *							
PROJECT Tutorial Htdocs This PC 3D Objects Desiston	A A A							
ATA (D:) PROJECT Tutorial htdocs This PC 3D Objects Desktop	* * *							
PROJECT Tutorial Htdocs This PC JD Objects Desktop Documents Donynloads	* * *							
ATA (D.) PROJECT Tutorial Htdocs This PC Dobjects Dobjects Dosktop Dosktop Dosktop Downloads Music	* * *							
an DATA (D.) PROJECT Tutorial Htdocs This PC Dobjects Desktop Documents Documents Music Ptures	* * *							
ana DATA (D.) PROJECT Tutorial Htdocs This PC Desktop Desktop Documents Documents Music Plotures Videos	* * *							
ana DATA (D.) PROJECT Tutorial Htdocs This PC Docktop Docktop Docktop Docktop Docktop Docktop Plotures Videos Locel Disk (C;)	* * *							
a DATA (D:) PROJECT Tutorial Htdocs This PC Documents Documents Documents Downloads Music Pictures Videos Local Disk (C:) DATA (D:)	* * *							

Kemudian buka Control Panel, pilih "System"

📧 All Control Panel Items				- • ×
🗧 🔶 👻 🛧 🛅 > Control Panel 🤉	All Control Panel Items			y ο P Search C.,
Adjust your computer's settings				View by: Large icons 🔻
Administrative Tools	AutoPlay	Backup and Restore (Windows 7)	BitLocker Drive Encryption	Color Management
Credential Manager	Date and Time	Default Programs	Dell Audio	Dell Touchpad
Device Manager	Sa Devices and Printers	Ease of Access Center	File Explorer Options	a File History
Flash Player (32-bit)	A Fonts	Indexing Options	Intel(R) Computing Improvement Program	Intel® HD Graphics
Internet Options	Java	👞 Keyboard	Mail (Microsoft Outlook) (32-bit)	Mouse
Network and Sharing Center	Phone and Modern	Power Options	Programs and Features	a Recovery
Region	RemoteApp and Desktop Connections	Security and Maintenance	Sound	Speech Recognition
Storage Spaces	Sync Center	System	Taskbar and Navigation	Troubleshooting
User Accounts	Windows Defender	Windows Mobility Center	🤛 Windows To Go	Work Folders

Kemudian pilih "Advanced System Setting"



Kemudian klik tombol "Environtment Variables"



Pilih Path pada bagian System Variables kemudian Klik tombol "Edit"

Environment Variables		×
<u>U</u> ser variables for Hendi		
Variable	Value	^
JAVA_HOME	C:\Program Files\Java\jdk-11	
MOZ_PLUGIN_PATH	C:\Program Files (x86)\Foxit Software\Foxit Reader\plugins\	
OneDrive	C:\Users\Hendi\OneDrive	
Path	C:\Users\Hendi\AppData\Local\Programs\Python\Python39\Script	
TEMP	C:\Users\Hendi\AppData\Local\Temp	
TMP	C:\Users\Hendi\AppData\Local\Temp	
		*
	<u>N</u> ew <u>E</u> dit <u>D</u> elete	
System variables		
Variable	Value	^
DriverData	C:\Windows\System32\Drivers\DriverData	
NUMBER_OF_PROCESSORS	4	
OS	Windows_NT	
Path	C:\Program Files\Java\jdk-11;C:\Program Files (x86)\Common Files	
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC	
PROCESSOR_ARCHITECTURE	AMD64	
PROCESSOR IDENTIFIER	Intel64 Family 6 Model 60 Stepping 3. GenuineIntel	*
	Ne <u>w</u> Ed <u>i</u> t De <u>l</u> ete	
	OK Cancel	

Kemudian klik tombol "New"



Kemudian masukkan alamat folder bin pada jdk yang telah kita ekstrak dalam hal ini misalnya

"C:\Program Files\Java\jdk-15.0.1\bin"

Edit environment va	ariable			×
C:\Program File	s\Git\cmd		^	New
C:\laragon\bin\	php\php-7.4.7-nts-Win32-	vc15-x64		
C:\ProgramData	\ComposerSetup\bin			<u>E</u> dit
C:\Users\Hendi\	AppData\Roaming\Pub\C	ache\bin		
C:\WINDOWS\S	ystem32\OpenSSH\			<u>B</u> rowse
C:\Program Files	s\PuTTY\			
C:\Program Files	s (x86)\Yarn\bin\			<u>D</u> elete
~\AppData\Roa	ming\Composer\vendor\b	oin		
D:\Android\flutt	ter\bin			
C:\Program Files	s\MongoDB\Server\4.2\bir	1 IIII		Move <u>U</u> p
D:\Android\cmd	lline-tools\tools\bin			
D:\Android\cmd	line-tools\platform-tools			Move D <u>o</u> wn
D:\Android\grad	lle\bin			
D:\Android\scrc	ру			
C:\Program Files	s\nodejs\			Edit <u>t</u> ext
C:\Users\Hendi\	AppData\Roaming\npm			
C:\cygwin64\bir	ı			
C:\Program Files	s\Docker\Docker\resource	s\bin		
C:\ProgramData	\DockerDesktop\version-b	oin		
C:\Program File	s\Java\jdk-15.0.1\bin			
			•	
		Γ	OK	Const
			OK	Cancel
				.::

Biasanya agar JDK dapat berfungsi perlu dilakukan restart laptop/komputer. Untuk memeriksa apakah installasi berhasil, buka command prompt kemudian ketikkan **java -version** atau **javac -version**



Install Android Studio

Android studio dapat diunduh pada laman <u>https://developer.android.com/studio</u> . Setelah diunduh klik dua kali pada file yang telah diunduh tersebut, kemudian lakukan installasi dengan mengikuti langkah-langkah yang telah disediakan

Android Studio Setup Wizard	3443	۵	
Welcome Android Studio			
Welcome! This wizard will set up your development environment for Android Studio. Additionally, the wizard will help port existing Android apps into Android Studio or create a new Android application project.			
Previo Next Cancel	F	inish	

Kemudian pilih tipe standar dan klik next

Android Studio Setup Wizard	3223		8
Install Type			
Choose the type of setup you want for Android Studio:			
 Standard Android Studio will be installed with the most common settings and options. Recommended for most users. Custom You can customize installation settings and components installed. 			
Previo <u>N</u> ext <u>C</u> ancel) [inish	

Pilih thema tampilan kemudian klik next

Android Studi	io Setup Wizard 🗆
Darcula module src & HelloWorld	🔘 Light module) 🖿 src) 💣 HelloWorld
<pre>import javax.swing.*; import javax.awt.*; public class HelloWorld { public HelloWorld() { JFrame frame = new JFrame ("Hello w JLabel label = new JLabel(); label. label. Breakpoints frame fr</pre>	<pre> HelloWorld.java × import javax.swing.*; import javax.awt.*; public class HelloWorld { public HelloWorld() { JFrame frame = new JFrame ("Hello we JLabel label = new JLabel(); label.setFont(new Font("Serif", Font label. Frame frame</pre>

Kemudian klik tombol next

	Android	Studi	io Setu	p Wizar	rd				 	0
Verify Settings										
If you want to review or change any Current Settings:	of your inst	tallati	on setti	ngs, cli	ck Pre	vious.				
Setup Type: Standard										
SDK Folder: /home/ibnuzakariyya/Android/Sc	lk									
Total Download Size: 601 MB										
SDK Components to Download:										
Android Emulator	292 MB									
Android SDK Build-Tools 29.0.2	39.7 MB									
Android SDK Platform 29	74.6 MB									
			<u>P</u> rev	ious		lext		Cance		
							- 2			

Pastikan komputer terhubung dengan internet yang stabil, karena android studio akan mengunduh komponen-komponen yang diperlukan

Android Studio Setup Wizard			8
Downloading Components			
Downloading https://dl.google.com/android/repository/sdk-tools-linux-4333796.zip			
Show Details			
Previous Next Cancel	Ē	Inish	N

Setelah selesai klik finish

Install Flutter

Flutter adalah sebuah framework open-source yang dikembangkan oleh Google untuk membangun antarmuka (*user interface/UI*) aplikasi Android dan iOS.

Apa bedanya membuat aplikasi android menggunakan Java/Kotlin *(native)* dengan Flutter. Dari bahasa pemrograman yang digunakan, Flutter menggunakan bahasa pemrograman Dart, sedangkan Android Native menggunakan bahasa pemrograman Java dan Kotlin. Aplikasi yang kita buat dengan Flutter dapat di-build ke Android dan iOS. Sedangkan Android Native hanya bisa di-build ke Android saja.

Untuk menginstall flutter, buka laman <u>https://flutter.dev/docs/get-started/install</u> Kemudian

pilih "Windows"



Kemudian pilih flutter_windows_ untuk mengunduh file flutter

< Flutter	Docs Showcase Community Q	У	0	Get started
Get started I. Install S. Set up an editor T. S. Learn more From another platform?	Get the Flutter SDK 1. Download the following installation bundle to get the latest stable release of the Flutter SDK: flutter_windows_1.22.5-stable.zip For other release channels, and older builds, see the SDK releases page.		Conter System re Get the FI Update y Run flutt Android s Install A Set up y	nts equirements lutter SDK your path ter doctor etup ndroid Studio our Android device
Flutter for Android devs	 Extract the zip file and place the contained flutter in the desired installation location for the Flutter SDK (for example, C:\src\flutter). 		Set up th emulato	he Android r
Flutter for IOS devs Flutter for React Native devs	& Warning: Do not install Flutter in a directory like C:\Program Files\ that requires elevated privileges.		Next step	
Flutter for web devs Flutter for Xamarin, Forms devs	If you don't want to install a fixed version of the installation bundle, you can skip steps 1 and 2. Instead, get the source code from the Flutter repo on GitHub, and change branches or tags as needed. For example:			
Introduction to declarative UI Dart language overview [®]	C:\src>git clone https://github.com/flutter/flutter.git -b stable			

Kemudian ekstrak file zip flutter misalnya di "D:/Android"



Kemudian buka Control Panel, pilih "System"

🔝 All Co	ntrol Panel Items								- œ ×
$\leftarrow \rightarrow$	 	All Control I	Panel Items						👻 👸 🔑 Search C.,
Adjus	t your computer's settings								View by: Large icons 🔻
	Administrative Tools		AutoPlay	*	Backup and Restore (Windows 7)	-	BitLocker Drive Encryption	1	Color Management
-	Credential Manager	P	Date and Time	16	Default Programs	Indi	Dell Audio		Dell Touchpad
1	Device Manager		Devices and Printers	0	Ease of Access Center	E	File Explorer Options	6	File History
£	Flash Player (32-bit)	A	Fonts	R	Indexing Options	Intel	Intel(R) Computing Improvement Program	1	Intel® HD Graphics
	Internet Options	4 ,	Java	~	Keyboard	0	Mail (Microsoft Outlook) (32-bit)	0	Mouse
1	Network and Sharing Center	0	Phone and Modern	1	Power Options	i i	Programs and Features	Þ	Recovery
1	Region	-	RemoteApp and Desktop Connections	p	Security and Maintenance	9	Sound	Ģ	Speech Recognition
	Storage Spaces	0	Sync Center		System		Taskbar and Navigation		Troubleshooting
88	User Accounts	2	Windows Defender Firewall	5	Windows Mobility Center	-	Windows To Go	6	Work Folders

Kemudian pilih "Advanced System Setting"



Kemudian klik tombol "Environtment Variables"

System Properties	×
Computer Name Hardware Advanced	System Protection Remote
You must be logged on as an Administr	ator to make most of these changes.
Performance	
Visual effects, processor scheduling,	memory usage, and virtual memory
	Settings
	<u>J</u> etungs
User Profiles	
Desktop settings related to your sign-	n
	S <u>e</u> ttings
Startup and Recovery	
System startup, system failure, and de	bugging information
	Se <u>t</u> tings
	Enviro <u>n</u> ment Variables
0	Cancel Apply
0	

Kemudian pilih pada "System Variables" pilih "Path" dan klik tombol "Edit"

Environment Variables						
<u>U</u> ser variables for Hendi						
Variable	Value	^				
JAVA_HOME	C:\Program Files\Java\jdk-11					
MOZ_PLUGIN_PATH C:\Program Files (x86)\Foxit Software\Foxit Reader\plugins\						
OneDrive	C:\Users\Hendi\OneDrive					
Path	C:\Users\Hendi\AppData\Local\Programs\Python\Python39\Script					
TEMP	C:\Users\Hendi\AppData\Local\Temp					
TMP	C:\Users\Hendi\AppData\Local\Temp					
		*				
	<u>N</u> ew <u>E</u> dit <u>D</u> elete					
System variables						
Variable	Value	^				
DriverData	C:\Windows\System32\Drivers\DriverData					
NUMBER_OF_PROCESSORS	4					
OS	Windows_NT					
Path	C:\Program Files\Java\jdk-11;C:\Program Files (x86)\Common Files					
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC					
PROCESSOR_ARCHITECTURE	AMD64					
PROCESSOR IDENTIFIER	Intel64 Family 6 Model 60 Stepping 3. GenuineIntel	*				
	Ne <u>w</u> Ed <u>i</u> t De <u>l</u> ete					
	OK Cancel					

Kemudian klik tombol "New"

Edit environment variable						
C:\Program Files\Java\jdk-11	^	<u>N</u> ew				
C:\Program Files (x86)\Common Files\Intel\Shared Files\cpp\bin\I						
C:\ProgramData\Oracle\Java\javapath		<u>E</u> dit				
C:\Program Files (x86)\Common Files\Oracle\Java\javapath						
C:\WINDOWS\system32		Browse				
C:\WINDOWS						
C:\WINDOWS\System32\Wbem		<u>D</u> elete				
C:\WINDOWS\System32\WindowsPowerShell\v1.0\						
C:\Program Files\Dart\dart-sdk\bin						
C:\Program Files\Git\cmd		Move <u>U</u> p				
C:\laragon\bin\php\php-7.4.7-nts-Win32-vc15-x64						
C:\ProgramData\ComposerSetup\bin		Move D <u>o</u> wn				
C:\Users\Hendi\AppData\Roaming\Pub\Cache\bin						
C:\WINDOWS\System32\OpenSSH\						
C:\Program Files\PuTTY\		Edit <u>t</u> ext				
C:\Program Files (x86)\Yarn\bin\						
~\AppData\Roaming\Composer\vendor\bin						
D:\Android\flutter\bin						
C:\Program Files\MongoDB\Server\4.2\bin						
D:\Android\cmdline-tools\tools\bin	~					
OK		Cancel				
UK UK		Cancel				

Kemudian masukkan alamat folder bin pada flutter yang telah kita ekstrak dalam hal ini misalnya "D:\Android\flutter\bin"

dit environment variable		×
C:\Program Files\Dart\dart-sdk\bin	^	New
C:\Program Files\Git\cmd		
C:\laragon\bin\php\php-7.4.7-nts-Win32-vc15-x64		<u>E</u> dit
C:\ProgramData\ComposerSetup\bin		
C:\Users\Hendi\AppData\Roaming\Pub\Cache\bin		Browse
C:\WINDOWS\System32\OpenSSH\		
C:\Program Files\PuTTY\		<u>D</u> elete
C:\Program Files (x86)\Yarn\bin\		
~\AppData\Roaming\Composer\vendor\bin		
D:\Android\flutter\bin		Move <u>U</u> p
C:\Program Files\MongoDB\Server\4.2\bin		
D:\Android\cmdline-tools\tools\bin		Move D <u>o</u> wn
D:\Android\cmdline-tools\platform-tools		
D:\Android\gradle\bin		
D:\Android\scrcpy		Edit <u>t</u> ext
C:\Program Files\nodejs\		
C:\Users\Hendi\AppData\Roaming\npm		
C:\cygwin64\bin		
C:\Program Files\Docker\Docker\resources\bin		
C:\ProgramData\DockerDesktop\version-bin	*	
	ОК	Cancel

Konfigurasi Android Studio dengan Flutter

Jalankan Android Studio kemudian pada menu "Configure" pilih "Plugins"



Pilih "Flutter" kemudian klik tombol "Install"

🞽 Plugins					
		Marketplace	installed 🔕 🛛 🗱	I.	
Q+ Type/				-	
II Id	leaVim			Flutter	
	.9,3M 12 4.4 JetBrains			±7,2M ☆4.3 flutter.dev	
	DB Idea 672.9K - か.4.7. Dhilippo Breault			Languages 52.1.1 Des 17, 2020	
	ore, sky in this presented on		Plugin homepage		
۹ م	o deGlance 1,3M ☆ 4.8 Vektah		Support for devel easy and product high-performanc plugin will also in	oping Flutter applications. Flutter gives ive way to build and deploy cross-platf e mobile apps for both Android and iO stall the Dart plugin.	developers an orm, S. Installing this
	ndroid ButterKnife Zelezny .702,8K ☆ 4 Avast		For some tools, th content from the to replace Chrom	his plugin uses Chromium through JxBi web. JxBrowser complies with LGPL an ium with another component. To do th	owser to display d offers an option iis:
M <u>+</u>	laterial Theme UI .9,7M 12 4.2 Mallowigi		 Find the JxBro /flutter-intelli Use the instru- modified con 	owser files stored in the plugins director j/jxbrowser. Ictions and build script from JxBrowser nponents.	y, under to relink it with
🖊 п	utter		Change Note	5	
× -	.7,2M ☆ 4.3 flutter.dev		Size: 9,7M		
G G	envmotion				
C -	. 1,9M ☆ 3.4 Genymotion				
e :	TO generator :85,9K ☆4.1 N Vinay shetty				
(?)				ок	Cancel

Setelah selesai, restart/tutup Android Studio

Membuat dan menjalankan projek

Jalankan Android Studio kemudian pilih "Create New Flutter Project"



Kemudian pilih "Flutter Application" dan klik "Next"

🚎 Create New I	Flutter Project				×
٨	New Flutter Project	t			
Fi Select an "A Select a "Pio Select a "Pa Select a "M	utter Application utter Application utter Application upplication	Flutter Plugin	Flutter Package	Flutter Module	
			Previous <u>N</u> ext	<u>C</u> ancel Finis	

Kemudian tentukan :

Project Name : belajarflutter

Flutter SDK path : D:\Android\flutter (masukkan sesuai alamat folder flutter diletakkan)
Project location : D:\Belajar (bebas memasukkan dimanapun)
Description : Membuat Aplikasi Mobile Flutter (bebas)

Kemudian klik tombol "Next"

🛋 Create New Flutter Project	×
New Flutter Application	~
Configure the new Flutter application	
belajarflutter	
Flutter SDK path	
D:\Android\flutter	👻 🛄 🛨 Install SDK
Project location	
D:\Belajar	
Description	
Membuat Aplikasi Mobile Flutter	
	Create project offline
	Previous Next Cancel Finish

Kemudian klik tombol "Finish". Pastikan perangkat terhubung ke internet, karena diperlukan untuk mengunduh projek yang dibuat



Setelah selesai akan muncul projek yang telah dibuat



Untuk menjalankan projek, kita memerlukan emulator android. Pertama kita akan membuat emulator android dengan cara klik "AVD Manager" pada pojok kiri atas



Kemudian klik tombol "Create Virtual Device"

🚈 Android Virtual Device Manager		\times
Your Virtual Devices		
	Virtual devices allow you to test your application without having to own the physical devices.	
	To prioritize which devices to test your application on, visit the Android Dashboards, where you can get up-to-date information on which devices are active in the Android and Google Play ecosystem.	

Pilih salah satu device yang dinginkan, misalnya "Nexus S" kemudian klik tombol "Next"

۰	Virtual Device Co	nfiguration						×	
	Select Hardware								
4	Choose a dev	vice definition							
							n Nexus S		
	Category	Name 🔻	Play Store		Resolution	Density			
		Pixel	⊳		1080x1920	420dpī			
		Nexus S					Size: normal		
	Wear OS	Nexus One		3,7"	480x300	hdpi	4.0" B00px Density hdpi		
	Tablet	Nexus 6P		5,7"	1440x2560	560dpi			
	Automotive	Nexus 6		5,96"	1440x2560	560dpī			
		Nexus 5X	⊳	5,2"	1080x1920	420dpī			
						3	Clone Device.		
1							Previous Next Cancel Fina		

Kemudian kita akan memilih Versi Sistem Operasi Android, dalam hal ini misalnya "Q", jika belum ada maka kita akan diminta untuk mengunduh terlebih dahulu.

📉 Virtual	Device Configuration						
	System In	nage					
Select	t a system image	r Other Images					
	Anneaded Xoo mages	s other mages			Q		
Relea	ase Name	API Level 🔻		larget			
R Do						API Level	
Q Do	wnload		<i>18</i> 6	Android 10.0 (Google APIs)		29	
						Android	
						10.0	
						Google Inc	
						System Image	
	pop Download					x86	
9 As	ystem image must be sele	ected to continue.			We recommend the and support Goog	hese images because they run the faste gle APIs.	
?					Previous	Next <u>C</u> ancel	Finish

Setelah itu klik Finish

🗯 Virtual Device Configuration	×
Android Virtual Device (AVD)	
Verify Configuration	
AVD Name Nexus \$ API 22	AVD Name
Nexus S 4.0 480x800 hdpi Change	" The name of this AVD.
Lollipop Android 5.1 x86 Change	
Startup orientation	
Show Advanced Settings	
	Previous Next Cancel Finish

Untuk Menjalankan Emulator, klik logo play

Andre 🚵	🗰 Android Virtuel Device Manager — 🗆 🔿										
	Your Virtual Devices										
Туре	Name		Resolution		Target						
Co								Þ	2		
								S			

Kemudian akan muncul emulator android seperti berikut:



Jika emulator sudah berjalan, kita dapat mengeksekusi projek dengan cara klik tombol play (berwarna hijau) pada Android Studio



Pada emulator akan tampil hasil projek seperti berikut



Membuat dan menjalankan projek dengan VSCode dan Handphone Android

Install VSCode sebagai alternatif editor

Unduh VSCode pada laman <u>https://code.visualstudio.com/download</u> kemudian install. Agar flutter dapat digunakan pada VSCode, perlu diinstall beberapa extension flutter yang dibutuhkan.



Setelah itu restart/tutup VSCode

Membuat projek flutter dengan VSCode

Jalankan VSCode, pada menubar pilih view -> command Palette... atau dapat juga dengan shortcut Ctrl + Shift + P

1 File Edit Selection	View Go Run Terminal	Help	Welcome - Visual Studio Code	σ	x
🖸 刘 Welcome 🗙	Command Palette	Ctrl+Shift+P			****
w	Open View_				
O Start	Appearance Editor Layout		Customize		
Open folds	Explorer Search	Ctrl+Shift+E Ctrl+Shift+F	Tools and languages Install support for JavaScript, Python, Java, PHP, Azure, Docker and more		
e ² Recent	SCM Run Extensions	Ctrl+Shift+G Ctrl+Shift+D Ctrl+Shift+X	Settings and keybindings Install the settings and keyboard shortcuts of Vim, Sublime, Atom and others		
toko-api tokokita i simis C/v fiatlabadim istini DAT	Output Debug Console Terminal	Ctrl+Shift+U Ctrl+Shift+Y Ctrl+	Color theme Make the editor and your code look the way you love d_pack		
More (C	Problems	Ctrl+Shift+M	Learn		
Help Printable k	Show Minimap Show Breadcrumbs V Render Whitespace		Find and run all commands Rapidly access and search commands from the Command Palette (Chrl+Shift+P)		
Introductor Tips and its Product doe	Render Control Characters		Interface overview Get a visual overlay highlighting the major components of the UI		
GitHub repo Stack Overfi Join our Nev			Interactive playground		
8			ing concession material reactives in a state ventoriouph		
Show w	elcome page on startup				
⊗0 ≜ 0				R	٥

Kemudian ketikkan flutter dan pilih Flutter: New Project

×1 1	ile Edit Selection View Go Ru	in Terminal Help Get Started - Visual S	itudio Code		٥	×
Q1	📢 Get Started 🗙					
Q & 4 [9	Start C∔ New File 10 Open File 12 Open Folder 12 Clone Git Repositi	Hutter: New Project Pub: List Outdated Packages Pub: Get Packages File: Save without Formatting Preferences: File Icon Theme Activate Add Browser Breakpoint Add Function Breakpoint Azure Container Registry: Build Image in Azure	cori + K Crif + shift + s other commands	make VS Code yours.		
₽ BB	Recent Azure Container Registry: Create Registry config-file DX/PROJECT\Bandiklat\APP minio DX/PROJECT\Bandiklat\APPUTILS config DX/PROJECT\Bandiklat\APPUTILS		😝 Boost your Productivity			
۲	latihan DABelajar\PA AdminLTE-3.1.0 DA More	Apemrograman dasar java	Getting Started with Spring Boo Code	ot in VS Updated		
			Get Started with Java Developm	ent Updated		
8						
-		🗸 Show welcom	ne page on startup			
× (図0 <u>本</u> 0				R	8

Kemudian pilih Application

M F	ile Edit Selection View Go Ru	in Terminal Help Get Sta	rted - Visual Studio Code		6	t s	
Q,	📢 Get Started 🛛 🗙	Which Flutter template?					
þ	Start	Application Generate a Flutter application.					
ę.o	C+ New File	Module Generate a project to add a Flutter module Package	e to an existing Android or iOS application.	1.200			
å	D Open Folder P Clone Git Repositi	Generate a shareable Flutter project conta Plugin	ining modular Dart code.	make VS Code yours.			
G	Recent	Skeleton Generate a List View / Detail View Flutter a	application that follows community best practices.	overview of the must-have			
щ	config-file D:\PROJE	T\Bandiklat\APP	🔗 Boost your Productivity				
A	soal D:\PROJECT\Bar	dildat\APP\UTILS\database					
۲	latihan D:\Belajar\PN AdminLTE-3.1.0 D:\	Npemrograman dasar java	Getting Started with Spring Boo Code	ot in VS Updated			
***			💟 Get Started with Java Developm	vent Updated			
2000							
8							
500		✓ s	show welcome page on startup				
× 6	00A0					R	C

Pilih folder tempat projek tersebut

Select Folder			>	Circual Studio Gode	٥	
← → ~ ↑ 🦲 > This	PC > DATA (D:) > Belajar	🗢 🐧 ,D Search B	slajar			
Organize 👻 New folder			81 - 🔞			
Tutonal 🖋 ^	Name	Date modified	Type	Customize		
htdocs 🖈	Android	15/10/2020 14.26	Filefolder			
This RC	AndroidKotlin	05/12/2020 10.15	Filefolder	Tools and languages		
This PC	belajarflutter		File folder	Install support for JavaScript, Python, Java, PHP, Azure, Docker and more		
and objects	Dart	13/10/2020 15:24	File folder			
Desktop		19/02/2021-09.13	File folder			
Documents	java 🛛	13/01/2021 11:25	File folder	Settings and keybindings		
🕹 Downloads	Kotlin	29/12/2020 11.19	Filefolder	install the settings and keyboard shortcuts of Vim, Sublime, Atom and others		
Music	📴 nodejs	29/10/2020 07.15	Filefolder			
Pictures	🧾 tokokita	22/12/2020 11.32	File folder	Calor theme		
Videos				Make the editor and your code look the way you love		
Local Disk (C:)						
3= DATA (D:)	¢			> Learn		
Folder:	Flutter			Ecunt		
	Select	a folder to create the project in	Cancel	Find and run all commands		
	Joect	a rolder to create the project in		Rapidly access and search commands from the Command Palette (Ctrl+Shift+P)		
Printuble R						
				Inforface complete		
				Intensice overview		
				der a visual ovenay ingining the major components of the or		
Stock Over	tion					
kein our Ne	evoletter			Interactive playground		
				Try out essential editor features in a short walkthrough		
Show v	welcome page on startup					

Kemudian tentukan nama projek flutter yang ingin dibuat misalnya aplikasi_flutter_pertamaku



Kemudian tekan Enter dan tunggu hingga proses unduhan selesai



Menjalankan aplikasi dengan Handphone Android

Untuk menjalankan projek flutter dari VSCode dapat menggunakan Emulator AVD yang telah dibuat sebelumnya menggunakan Android Studio ataupun menggunakan Device Handphone Android langsung

Untuk menggunakan android device secara langsung, pertama aktifkan dulu mode developer dengan cara buka **Setting** kemudian pilih **System** kemudian pilih **About Phone**, untuk masing-masing device mungkin terdapat perbedaan untuk lokasi **About Phone** ada pula yang berada pada **Additional Setting**



Kemudian pilih About Phone



Kemudian ketuk **Build number** beberapa kali, namun ini juga berbeda untuk beberapa versi misalnya untuk Xiaomi dengan mengetuk **MIUI Version** beberapa kali

🏺 Infinix X650C	1 <u>888</u> 8		×
09:24 🖬 🗊 🖬 🚥		≉ .ul.at	116 10's #(61)
< About phone	e		
Status Phone number, signal, etc.			
Device name Infinix HOT a			
Model Infinix X650C			
XOS version XOS v5.5.0 (P-P58-190722)			
Android version			
Build number x6600 H626DEJ-P-201207V314	\supset	>	
Device Info			
Legal information			
Certification info			
User Experience Improve	ement Program	n	
No need, you a	re already a dev	veloper.	

Selanjutnya mengaktifkan USB Debugger dengan cara pilih **Developer Option** pada **System**, **Developer Option** ini akan muncul setelah mode Developer diaktifkan dengan cara diatas



Kemudian aktifkan USB Debugging



Jika telah selesai, hubungkan Handphone android dengan laptop/komputer dengan kabel data, untuk memeriksa apakah sudah terhubung dengan Handphone, dapat dilihat pada VSCode bagian pojok kanan bawah akan tertera nama device yang terhubung


Atau jika pada Android Studio terletak pada toolbar bagian atas tengah



Agar laptop bekerja lebih ringan dapat digunakan Text Editor VSCode dan menjalankan projek langsung menggunakan Handphone Android. Untuk menjalankan projek melalui VSCode dengan klik logo play pada bagian pojok kanan atas



Struktur Folder Flutter

Adapun struktur folder Projek flutter adalah sebagai berikut:



- Image: android berisi source code untuk aplikasi android;
- Image: Image:
- Iib berisi source code Dart, di sini kita akan menulis kode aplikasi;
- Itest berisi source code Dart untuk testing aplikasi;
- gitignore adalah file <u>Git;</u>
- Imetadata merupakan file yang berisi metadata project yang di-generate otomatis;
- Image: packages merupakan file yang berisi alamat path package yang dibuat oleh pub;
- Image: Image:

pubspec.lock merupakan file yang berisi versi-versi library atau package. File ini dibuat oleh pub. Fungsinya untuk mengunci versi package.

pubspec.yaml merupakan file yang berisi informasi tentang project dan libraray yang dibutuhkan;

 README.md merupakan file markdown yang berisi penjelasan tentang source code.

Membuat Hello World

Buka projek **aplikasi_flutter_pertamaku** menggunakan VSCode agar lebih ringan. Buka file **main.dart** yang terletak pada folder **lib** kemudian ubah menjadi

```
1. import 'package:flutter/material.dart';
2.
3. void main() {
4. runApp(const MyApp());
5. }
```

```
6.
7. class MyApp extends StatelessWidget {
8. const MyApp({Key? key}) : super(key: key);
9.
10. @override
11. Widget build(BuildContext context) {
12. return MaterialApp(
          title: "Aplikasi Flutter Pertama",
home: Scaffold(
13.
14.
15.
            appBar: AppBar(
             title: const Text('Belajar Flutter'),
16.
            ),
17.
18.
         ),
19.
        );
20. }
21. }
```

Ketika dijalankan akan menghasilkan



Kemudian untuk menambahkan tampilan dibagian dalam (body), pada fungsi Scaffold terdapat parameter **body**. Silahkan modifikasi **main.dart** menjadi seperti berikut

```
1. import 'package:flutter/material.dart';
2.
3. void main() {
4. runApp(const MyApp());
5. }
6.
7. class MyApp extends StatelessWidget {
8. const MyApp({Key? key}) : super(key: key);
9.
10. @override
11. Widget build(BuildContext context) {
12. return MaterialApp(
13. title: "Aplikasi Flutter Pertama",
```

```
14. home: Scaffold(
              appBar: AppBar(
title: Text('Belajar Flutter'),
15.
16.
17.
              ),
              body: const Center(
    child: Text("Hello World"),
18.
19.
20.
              ),
21.
            ),
22.
         );
23.
       }
24. }
```

Sehingga akan menghasilkan



Pada aplikasi di atas, kita membuat *StatelessWidget* yang berisi widget MaterialApp(). Kemudian di dalam MateralApp() berisi widget lagi: Scaffold, AppBar, Center, dan Text. Ini adalah widget dasar.

Penjelasan:

- MyApp adalah StatelessWidget, merupakan widget induk;
- MaterialApp adalah widget yang membungkus beberapa widget yang menggunakan tema material design
- Scaffold adalah widget untuk struktur dasar material design;
- AppBar adalah widget untuk membuat AppBar;
- Center adalah Widget layout untuk membuat widget ke tengah;
- Text adalah widget untuk membuat teks.

Untuk mempermudah dalam pembacaan kode dan maintenance dapat dilakukan dengan memisahkan **MyApp** dengan halaman yang ingin ditampilkan.

Silahkan buat sebuah file dengan nama hello_world.dart di dalam folder lib



Kemudian bagian **Scaffold** pada **main.dart** yang telah dibuat tadi akan kita masukkan ke dalam **hello_world.dart**, sehingga pada **hello_world.dart** akan menjadi

```
1. import 'package:flutter/material.dart';
2.
3. class HelloWorld extends StatelessWidget {
4. @override
5.
     Widget build(BuildContext context) {
6.
       return Scaffold(
7.
         appBar: AppBar(
8.
          title: const Text('Belajar Flutter'),
9.
         ),
10.
         body: const Center(
           child: Text('Hello World'),
11.
12.
         ),
13.
       );
14.
     }
15.}
```

Pada file main.dart kita modifikasi kembali pada bagian home menjadi

```
1. import 'package:aplikasi_flutter_pertamaku/hello_world.dart';
2. import 'package:flutter/material.dart';
3.
4. void main() {
5.
      runApp(const MyApp());
6. }
7.
8. class MyApp extends StatelessWidget {
9.
      const MyApp({Key? key}) : super(key: key);
10.
11.
     @override
12. Widget build(BuildContext context) {
13.
        Return const MaterialApp(
14.
       title: "Aplikasi Flutter Pertama",
15.
          home: HelloWorld(),
16.
       );
-/. }
18.}
```

Pada bagian home, kita memanggil class **HelloWorld** yang telah kita buat sebelumnya pada file **hello_world.dart**

Jika kita perhatikan pada bagian body, terdapat Widget Center kemudian didalam Widget Center tersebut terdapat parameter child untuk meletakkan Widget lain didalam widget tersebut, dalam hal ini adalah Widget Text

```
Center(
child: Text('Hello World'),
),
```

Catatan : dalam Widget selain **child**, terdapat pula **children** dengan type data array yang dimana kita dapat menempatkan beberapa Widget didalamnya contohnya pada Widget Column dan Row

Untuk mempercepat dalam pembuatan class pada VSCode dapat dilakukan dengan mengetik **st** kemudian memilih **stateless widget** ataupun **stateless widget** kemudian ketikkan nama class yang diinginkan



Membuat Widget Column

Buat sebuah file dengan nama **column_widget.dart** didalam folder lib, kemudian ketikkan kode berikut

```
1. import 'package:flutter/material.dart';
2.
3. class ColumnWidget extends StatelessWidget {
4.
   const ColumnWidget({Key? key}) : super(key: key);
5.
6. @override
     Widget build(BuildContext context) {
7.
8. return Scaffold(
9.
           appBar: AppBar(
10.
             title: const Text('Widget Column'),
11.
            ),
12.
           body: Column(
13.
             children: const [
14.
              Text('Kolom 1'),
15.
               Text('Kolom 3'),
               Text('Kolom 2'),
16.
17.
               Text('Kolom 4')
18.
             ],
19.
            ));
20.
     }
21. }
```

Kemudian pada file main.dart ubah kodenya menjadi

```
1. import 'package:aplikasi_flutter_pertamaku/column_widget.dart';
2. import 'package:flutter/material.dart';
3.
4. void main() {
5. runApp(const MyApp());
6. }
7.
8. class MyApp extends StatelessWidget {
9. const MyApp({Key? key}) : super(key: key);
```

```
10.
11. @override
12. Widget build(BuildContext context) {
13. return const MaterialApp(
14. title: "Aplikasi Flutter Pertama",
15. home: ColumnWidget(),
16. );
17. }
18. }
```

Dan hasilnya akan menjadi seperti berikut



Column biasanya digunakan untuk membuat Form

Membuat Widget Row

Untuk menampilkan Widget dalam posisi horizontal dapat menggunakan Widget Row. Buat sebuah file didalam folder **lib** dengan nama **row_widget.dart**, kemudian ketikkan kode berikut

```
1. import 'package:flutter/material.dart';
2.
3. class RowWidget extends StatelessWidget {
4. const RowWidget({Key? key}) : super(key: key);
5.
6. @override

    Widget Duringen
    return Scaffold(
appBar: AppBar(

      Widget build(BuildContext context) {
          appBar: AppBar(
10.
            title: const Text('Widget Row'),
11.
          ),
12.
          body: Row(
13.
            children: const [
14.
              Text('Row 1'),
              Text('Row 2'),
15.
              Text('Row 3'),
16.
17.
              Text('Row 4')
18.
              ],
19.
          ),
20.
        );
21.
      }
22.}
```

Kemudian seperti sebelumnya masukkan class **RowWidget** tersebut kedalam home pada **main.dart**, dan hasilnya akan menjadi



StatelessWidget dan StatefullWidget

StatelessWidget adalah class widget yang propertinya *immutable*, artinya nilainya tidak bisa diubah, sedangkan **StatefullWidget** nilainya dapat berubah-ubah.

Contoh StatelessWidget :



9. } 10. }

Contoh StatefullWidget

```
1. class HelloWorld extends StatefulWidget {
2. const HelloWorld({ Key? key }) : super(key: key);
3.
5. _HelloWorldState createState() => _HelloWorldState();
6. }
     @override
7.
8. class _HelloWorldState extends State<HelloWorld> {
9.
     @override
10. Widget build(BuildContext context) {
11.
       return Container(
12.
13.
        );
14. }
15.}
```

Membuat Form

Selanjutnya kita akan belajar membuat form pada flutter, agar lebih rapi untuk tampilan halaman akan kita kelompokkan dalam sebuah folder tersendiri, dalam hal ini kita membuat folder dengan nama **ui** didalam folder **lib**.



Kemudian didalam folder **ui** tersebut kita buat sebuah file dengan nama **produk_form.dart**

	✓ APLIKASI_FLUTTER_PERTAMAKU
0	> Im .dart_tool
1	> 📪 Jidea
ă	> IIII android
P	> 📷 build
	> lis ios
2	∞ 👼 lib
3.4	
nD	S produk_form.dart
	column_widget.dart
	le hello_world.dart
Д	in main.dart
	😰 row_widget.dart
(L	> DU test
20	🚯 .gitignore
545	.metadata
-	.packages
	aplikasi flutter pertamaku.iml
	pubspec.lock
	de pubspec yaml
	README.md

Kemudian Ketikkan kode berikut

```
1. import 'package:flutter/material.dart';
2.
3. class ProdukForm extends StatefulWidget {
     const ProdukForm({Key? key}) : super(key: key);
4.
5.
6.
     @override
     _ProdukFormState createState() => _ProdukFormState();
7.
8. }
9.
10. class _ProdukFormState extends State<ProdukForm> {
11.
     @override
12. Widget build(BuildContext context) {
13.
       return Scaffold(
14.
         appBar: AppBar(
15.
           title: const Text('Form Produk'),
16.
         ),
17.
         body: SingleChildScrollView(
18.
           child: Column(
19.
             children: [
               TextField(decoration: const InputDecoration(labelText: "Kode Produk")),
20.
               TextField(decoration: const InputDecoration(labelText: "Nama Produk")),
21.
               TextField(decoration: const InputDecoration(labelText: "Harga")),
22.
23.
               ElevatedButton(onPressed: () {}, child: const Text('Simpan'))
24.
             ],
25.
           ),
         ),
26.
27.
       );
28.
     }
29.}
```

Ubah pada main.dart dengan memanggil class ProdukForm, sehingga hasilnya akan menjadi

🏺 Infinix X650C		<u>228</u>		×
13:02 🗉 🖬 î 🚥			r≉ala	1 🖧 Agg
Form Produk				
Kode Produk				
Nama Produk				
Harga				
	Simpan			
O	0		0	

Pemisahan Widget Kedalam fungsi-fungsi

Agar kode mudah dibaca dan dikembangkan, akan lebih baik jika widget-widget yang digunakan dipisahkan kedalam method/function tertentu, misalnya pada **produk_form.dart** terdapat widget seperti **TextField dan Button**, pada widget tersebut akan kita pisahkan kedalam method tersendiri didalam class, sehingga menjadi seperti berikut

```
1. import 'package:flutter/material.dart';
2.
3. class ProdukForm extends StatefulWidget {
4. const ProdukForm({Key? key}) : super(key: key);
5.
6. @override
7. _ProdukFormState createState() => _ProdukFormState();
```

```
8. }
9.
10. class _ProdukFormState extends State<ProdukForm> {
11.
     @override
12. Widget build(BuildContext context) {
13.
       return Scaffold(
14.
         appBar: AppBar(
15.
            title: const Text('Form Produk'),
16.
         ),
17.
         body: SingleChildScrollView(
18.
            child: Column(
19.
              children: [
20.
                _textboxKodeProduk(),
21.
                _textboxNamaProduk(),
22.
                _textboxHargaProduk(),
                _tombolSimpan()
23.
24.
              ],
25.
            ),
         ),
26.
27.
       );
28.
     }
29.
      _textboxKodeProduk() {
30.
       return TextField(decoration: const InputDecoration(labelText: "Kode Produk"));
31.
32.
     }
33.
34.
      _textboxNamaProduk() {
35.
       return TextField(decoration: const InputDecoration(labelText: "Nama Produk"));
36.
     }
37.
38.
      _textboxHargaProduk() {
39.
       return TextField(decoration: const InputDecoration(labelText: "Harga"));
40.
     }
41.
      _tombolSimpan() {
42.
       return ElevatedButton(onPressed: () {}, child: const Text('Simpan'));
43.
44.
     }
45.}
```

Membuat Detail Produk

Buat sebuah file dengan nama **produk_detail.dart** di dalam folder **ui**, kemudian ketikkan kode berikut

```
1. import 'package:flutter/material.dart';
2.
3. class ProdukDetail extends StatefulWidget {
4.
     final String? kodeProduk;
     final String? namaProduk;
5.
6. final int? harga;
7.
8.
     const ProdukDetail({Key? key, this.kodeProduk, this.namaProduk, this.harga})
9.
         : super(key: key);
10.
11.
     @override
     _ProdukDetailState createState() => _ProdukDetailState();
12.
13. }
14.
15. class _ProdukDetailState extends State<ProdukDetail> {
16. @override
17. Widget build(BuildContext context) {
18. return Scaffold(
```

```
19.
          appBar: AppBar(
20.
            title: const Text('Detail Produk'),
21.
          ).
22.
          body: Column(
23.
            children: [
              Text("Kode Produk : " + widget.kodeProduk.toString()),
24.
25.
              Text("Nama Produk : ${widget.namaProduk}"),
26.
              Text("Harga : ${widget.harga}")
27.
            ],
28.
          ),
29.
        );
30.
      }
31.}
```

Membuat fungsi tombol simpan dan menampilkan data pada Detail Produk

Buka kembali file produk_form.dart tambahkan attribute

```
final _kodeProdukTextboxController = TextEditingController();
final _namaProdukTextboxController = TextEditingController();
final _hargaProdukTextboxController = TextEditingController();
```

pada Class ProdukFormState sehingga menjadi

```
1. import 'package:aplikasi_flutter_pertamaku/ui/produk_detail.dart';
2. import 'package:flutter/material.dart';
3.
4. class ProdukForm extends StatefulWidget {
5.
      const ProdukForm({Key? key}) : super(key: key);
6.
7.
     @override
8.
      _ProdukFormState createState() => _ProdukFormState();
9. }
10.
11. class _ProdukFormState extends State<ProdukForm> {
12.
     final _kodeProdukTextboxController = TextEditingController();
            _namaProdukTextboxController = TextEditingController();
13.
      final
14.
     final _hargaProdukTextboxController = TextEditingController();
15.
16.
     @override
17.
     Widget build(BuildContext context) {
18.
       return Scaffold(
19.
          appBar: AppBar(
            title: const Text('Form Produk'),
20.
21.
          ),
22.
          body: SingleChildScrollView(
23.
            child: Column(
              children: [
24.
                _textboxKodeProduk(),
25.
26.
                _textboxNamaProduk(),
27.
                _textboxHargaProduk(),
                _tombolSimpan()
28.
29.
              ],
30.
            ),
31.
          ),
       );
32.
33.
     }
34.
     _textboxKodeProduk() {
35.
       return TextField(
36.
37.
            decoration: const InputDecoration(labelText: "Kode Produk"));
38.
39.
40.
     _textboxNamaProduk() {
```

```
41.
        return TextField(
42.
            decoration: const InputDecoration(labelText: "Nama Produk"));
43.
      }
44.
      _textboxHargaProduk() {
45.
      return TextField(decoration: const InputDecoration(labelText: "Harga"));
46.
47.
      }
48.
      _tombolSimpan() {
49.
50.
       return ElevatedButton(onPressed: () {}, child: const Text('Simpan'));
51.
      }
52.}
```

Pada setiap masing-masing TextField yang telah dibuat, data yang diinput dikirim ke attribute TextEditingController() yang telah kita buat sebelumnya

```
Pada fungsi _textboxKodeProduk() menjadi
```

```
_textboxKodeProduk() {
    return TextField(
        decoration: const InputDecoration(labelText: "Kode Produk"),
        controller: _kodeProdukTextboxController,
        );
}
```

Pada fungsi _textboxNamaProduk() menjadi

```
_textboxNamaProduk() {
    return TextField(
        decoration: const InputDecoration(labelText: "Kode Produk"),
        controller: _namaProdukTextboxController,
        );
}
```

```
Pada fungsi _textboxhargaProduk() menjadi
```

```
_textboxHargaProduk() {
    return TextField(
        decoration: const InputDecoration(labelText: "Kode Produk"),
        controller: _hargaProdukTextboxController,
        );
}
```

Kemudian pada fungsi **_tombolSimpan()** pada saat diklik akan mengirim data inputan dan menampilkan data tersebut pada **ProdukDetail** yang telah kita buat sebelumnya

```
namaProduk: namaProduk,
harga: harga,
)));
},
child: const Text('Simpan'));
}
```

Sehingga keseluruhan kode menjadi

```
    import 'package:aplikasi_flutter_pertamaku/ui/produk_detail.dart';

2. import 'package:flutter/material.dart';
З.
4. class ProdukForm extends StatefulWidget {
5.
     const ProdukForm({Key? key}) : super(key: key);
6.
7.
     @override
     _ProdukFormState createState() => _ProdukFormState();
8.
9.}
10.
11. class _ProdukFormState extends State<ProdukForm> {
12.
     final _kodeProdukTextboxController = TextEditingController();
     final _namaProdukTextboxController = TextEditingController();
13.
14.
     final _hargaProdukTextboxController = TextEditingController();
15.
16.
     @override
17.
     Widget build(BuildContext context) {
18.
        return Scaffold(
19.
          appBar: AppBar(
20.
            title: const Text('Form Produk'),
21.
          ),
          body: SingleChildScrollView(
22.
            child: Column(
23.
24.
              children: [
25.
                _textboxKodeProduk(),
26.
                _textboxNamaProduk(),
                _textboxHargaProduk(),
27.
                _tombolSimpan()
28.
29.
              ],
30.
            ),
31.
          ),
32.
        );
33.
     }
34.
     _textboxKodeProduk() {
35.
36.
        return TextField(
          decoration: const InputDecoration(labelText: "Kode Produk"),
37.
38.
          controller: _kodeProdukTextboxController,
39.
        );
40.
     }
41.
42.
     _textboxNamaProduk() {
43.
        return TextField(
44.
          decoration: const InputDecoration(labelText: "Nama Produk"),
45.
          controller: _namaProdukTextboxController,
46.
       );
47.
     }
48.
      _textboxHargaProduk() {
49.
50.
        return TextField(
          decoration: const InputDecoration(labelText: "Harga"),
51.
          controller: hargaProdukTextboxController,
52.
53.
        );
54.
     }
55.
```

```
56. _tombolSimpan() {
57.
        return ElevatedButton(
             onPressed: () {
58.
               String kodeProduk = _kodeProdukTextboxController.text;
String namaProduk = _namaProdukTextboxController.text;
59.
60.
61.
               int harga = int.parse(
62.
                    _hargaProdukTextboxController.text); //parsing dari String ke int
               // pindah ke halaman Produk Detail dan mengirim data
63.
64.
               Navigator.of(context).push(MaterialPageRoute(
65.
                    builder: (context) => ProdukDetail(
66.
                          kodeProduk: kodeProduk,
67.
                          namaProduk: namaProduk,
68.
                          harga: harga,
69.
                        )));
70.
             },
             child: const Text('Simpan'));
71.
72.
73.}
```

Form Produk	← Detail Produk
Kode Produk A001	Kode Produk : A001 Nama Produk : Kulkas Harga : 2400000
Nama Produk Kulkas	
Harge 2400000	
Simpan	
< Q 🖾 🖬 🖬 🛶 🌵	
$q^{^{\intercal}} w^{^{\intercal}} e^{^{\intercal}} r^{^{4}} t^{^{\$}} y^{^{6}} u^{^{7}} i^{^{\$}} o^{^{9}} p^{^{0}}$	
asd fghjkl	
☆ z x c v b n m ⊗	
?123 , 🕲 📃 . 🥑	
V 0	< 0 H

Membuat ListView Produk

Buat sebuah file dengan nama **produk_page.dart** di dalam folder **ui**, kemudian ketikkan kode berikut

```
1. import 'package:flutter/material.dart';
2.
3. class ProdukPage extends StatefulWidget {
4. const ProdukPage({Key? key}) : super(key: key);
5.
```

```
6.
      @override
      _ProdukPageState createState() => _ProdukPageState();
7.
8. }
9.
10. class _ProdukPageState extends State<ProdukPage> {
11.
      @override
     Widget build(BuildContext context) {
12.
13.
        return Scaffold(
14.
          appBar: AppBar(
            title: const Text('Data Produk'),
15.
16.
          ),
17.
          body: ListView(
18.
            children: const [
19.
              // List 1
20.
              Card(
21.
                child: ListTile(
                  title: Text('Kulkas'),
22.
                   subtitle: Text('2500000'),
23.
24.
                ),
25.
              ),
26.
              // List 2
27.
              Card(
28.
                child: ListTile(
                  title: Text('TV'),
subtitle: Text('5000000'),
29.
30.
31.
                ),
32.
              ),
              // List 3
33.
34.
              Card(
                child: ListTile(
35.
36.
                  title: Text('Mesin Cuci'),
                   subtitle: Text('1500000'),
37.
38.
                ),
39.
              )
40.
            ],
          ),
41.
42.
        );
43.
      }
44.}
```

Kemudian daftarkan ProdukPage pada main.dart, dan hasilnya akan menjadi seperti berikut



Membuat Route (Pindah Halaman)

Buka file produk_page.dart, kemudian modifikasi pada bagian AppBar menjadi seperti berikut

```
)
```

GestureDetector adalah widget yang digunakan untuk mendeteksi gesture pada widget seperti gesture ontap, doubletab dan lain-lain.

Secara Keseluruhan kode tersebut akan menjadi

```
1. import 'package:aplikasi_flutter_pertamaku/ui/produk_form.dart';
2. import 'package:flutter/material.dart';
3.
4. class ProdukPage extends StatefulWidget {
5.
     const ProdukPage({Key? key}) : super(key: key);
6.
7.
     @override
8.
      ProdukPageState createState() => ProdukPageState();
9. }
10.
11. class ProdukPageState extends State<ProdukPage> {
12. @override
13.
     Widget build(BuildContext context) {
14.
        return Scaffold(
15.
          appBar: AppBar(
16.
            title: const Text('Data Produk'),
17.
            actions: [
18.
              GestureDetector(
19.
                  // menampilkan icon +
20.
                  child: const Icon(Icons.add),
21.
                  onTap: () async {
22.
                    // berpindah ke halaman ProdukForm
23.
                    Navigator.push(
24.
                        context,
25.
                        MaterialPageRoute(
                             builder: (context) => const ProdukForm()));
26.
27.
                  })
28.
            ],
29.
          ),
30.
          body: ListView(
31.
            children: const [
32.
              // List 1
33.
              Card(
34.
                child: ListTile(
35.
                  title: Text('Kulkas'),
                  subtitle: Text('2500000'),
36.
37.
                ),
38.
              ),
              // List 2
39.
40.
              Card(
41.
                child: ListTile(
                  title: Text('TV'),
42.
                  subtitle: Text('5000000'),
43.
44.
                ),
              ),
45.
46.
              // List 3
47.
              Card(
                child: ListTile(
48.
49.
                  title: Text('Mesin Cuci'),
                  subtitle: Text('1500000'),
50.
51.
                ),
52.
              )
53.
            ],
         ),
54.
55.
        );
56.
     }
57.}
```

Hasilnya akan muncul icon + pada bagian kanan **AppBar**, jika diklik akan membuka P**rodukForm**

10:30 🗢 오 🖀 🕲	• 20	10:30 🗢 🗢 🖴 🕲	• 201
Data Produk	÷	← Form Produk	, and the second s
Kulkas 2500000		Kode Produk	
TV 5000000		Nama Produk Harga	
Mesin Cuci 500000		Simpan	
۰ ا	.,),	4 •	

Pemisahan Widget ke dalam Class StatelessWidget

Selain pemisahan widget ke dalam suatu function/method, pemisahan juga dapat dilakukan menggunakan class StatelessWidget, pada contoh kali ini kita akan memisahkan **Card** dengan membuat class tersendiri. Buka file **produk_page.dart**, kemudian buat sebuah class **ItemProduk** diluar class **ProdukPage**

```
class ItemProduk extends StatelessWidget {
  final String? kodeProduk;
  final String? namaProduk;
  final int? harga;
  const ItemProduk({Key? key, this.kodeProduk, this.namaProduk, this.harga})
      : super(key: key);
  @override
  Widget build(BuildContext context) {
    return Card(
      child: ListTile(
        title: Text(namaProduk.toString()),
        subtitle: Text(harga.toString()),
      ),
   );
 }
}
```

Sehingga kode pada produk_page.dart menjadi

```
1. import 'package:aplikasi_flutter_pertamaku/ui/produk_form.dart';
2. import 'package:flutter/material.dart';
3.
4. class ProdukPage extends StatefulWidget {
5.
      const ProdukPage({Key? key}) : super(key: key);
6.
      @override
7.
8.
     ProdukPageState createState() => ProdukPageState();
9. }
10.
11. class _ProdukPageState extends State<ProdukPage> {
12.
     @override
     Widget build(BuildContext context) {
13.
14.
        return Scaffold(
15.
          appBar: AppBar(
            title: const Text('Data Produk'),
16.
17.
            actions: [
18.
              GestureDetector(
19.
                  // menampilkan icon +
20.
                  child: const Icon(Icons.add),
21.
                  onTap: () async {
                    // berpindah ke halaman ProdukForm
22.
23.
                    Navigator.push(
24.
                        context,
25.
                        MaterialPageRoute(
26.
                             builder: (context) => const ProdukForm()));
27.
                  })
28.
            ],
29.
          ),
30.
          body: ListView(
            children: const [
31.
32.
              // List 1
33.
              ItemProduk(
                kodeProduk: "A001",
namaProduk: "Kulkas",
34.
35.
                harga: 2500000,
36.
37.
              ),
38.
              // List 2
39.
              ItemProduk(
                kodeProduk: "A002",
40.
                namaProduk: "TV",
41.
42.
                harga: 5000000,
43.
              ),
44.
              // List 3
45.
              ItemProduk(
                kodeProduk: "A003",
46.
                namaProduk: "Mesin Cuci",
47.
48.
                harga: 1500000,
49.
              ),
50.
            ],
          ),
51.
52.
        );
53.
      }
54.}
55.
56. class ItemProduk extends StatelessWidget {
      final String? kodeProduk;
57.
58.
      final String? namaProduk;
59.
      final int? harga;
60.
```

```
const ItemProduk({Key? key, this.kodeProduk, this.namaProduk, this.harga})
61.
62.
        : super(key: key);
63.
64. @override
     Widget build(BuildContext context) {
65.
66.
       return Card(
67.
         child: ListTile(
68.
           title: Text(namaProduk.toString()),
69.
            subtitle: Text(harga.toString()),
70.
         ),
71.
       );
72.
     }
73.}
```

Menampilkan Detail Produk saat ListView diklik

Pada bagian ini kita akan menambahkan sebuah fungsi dimana saat salah satu ListView Produk di klik, maka akan membuka halaman Detail Produk yang telah kita buat sebelumnya

Kita akan memodifikasi Class ItemProduk pada file **produk_page.dart**. Untuk menambahkan widget diatas widget yang telah dibuat dapat dilakukan dengan cara, arahkan kursor pada widget, misalnya dalam hal ini adalah widget **Card**



Pada bagian kiri akan muncul logo lampu, kemudian klik lampu tersebut dan pilih widget yang ingin ditambahkan atau dalam hal ini kita akan memilih **Warp with widget.**.

56	C	lass ItemProduk extends Sta	atelessWidget {					
57		<pre>final String? kodeProduk;</pre>						
58		<pre>final String? namaProduk;</pre>						
59		<pre>final int? harga;</pre>						
60								
61		<pre>const ItemProduk({Key? key</pre>	y, this.kodeProdu	ık, this.na	maProduk,	, this.	harga	})
62		: super(key: key);						
63								
64		Remove this widget						
65			context) {					
66	<u>`</u>	Wrap with Builder						
6/ 60		Wrap with Center	$\frac{1}{1}$					
60 60		Wrap with Column	u_{x} . u					
70		Wrap with Container	1. (05 cl 11g());					
71								
72		Wrap with Padding						
73	}	Wrap with Row						
74		Wrap with SizedBox						
		Wrap with StreamBuilder						
		Wrap with widget						
		Extract Method						
		Extract Local Variable						
		Extract Widget						
0 // (±è>	<u> </u>		Ln 66, Col 12	Spaces: 2	UTF-8	CRLF	Dart

Kemudian akan menjadi



Setelah itu ubah **widget** menjadi GestureDetector dan kita juga menambahkan onTap yang kemudian akan membuka halaman Detail Produk, sehingga kode untuk Class ItemProduk menjadi

```
class ItemProduk extends StatelessWidget {
  final String? kodeProduk;
  final String? namaProduk;
  final int? harga;
  const ItemProduk({Key? key, this.kodeProduk, this.namaProduk, this.harga})
      : super(key: key);
  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      child: Card(
        child: ListTile(
          title: Text(namaProduk.toString()),
          subtitle: Text(harga.toString()),
        ),
      ),
      onTap: () {
        Navigator.push(
            context,
            MaterialPageRoute(
                builder: (context) => ProdukDetail(
                      kodeProduk: kodeProduk,
                      namaProduk: namaProduk,
                      harga: harga,
                    )));
     },
   );
 }
}
```

Adapun untuk keseluruhan kode pada produk_page.dart sebagai berikut

```
    import 'package:aplikasi_flutter_pertamaku/ui/produk_detail.dart';

2. import 'package:aplikasi_flutter_pertamaku/ui/produk_form.dart';
3. import 'package:flutter/material.dart';
4.
5. class ProdukPage extends StatefulWidget {
6.
     const ProdukPage({Key? key}) : super(key: key);
7.
8.
     @override
9.
      _ProdukPageState createState() => _ProdukPageState();
10.}
11.
12. class _ProdukPageState extends State<ProdukPage> {
13.
     @override
14. Widget build(BuildContext context) {
15.
       return Scaffold(
16.
         appBar: AppBar(
            title: const Text('Data Produk'),
17.
18.
            actions: [
19.
             GestureDetector(
20.
                 // menampilkan icon +
21.
                  child: const Icon(Icons.add),
22.
                  onTap: () async {
23.
                    // berpindah ke halaman ProdukForm
```

```
Navigator.push(
24.
25.
                         context,
26.
                         MaterialPageRoute(
27.
                             builder: (context) => const ProdukForm()));
28.
                  })
29.
            ],
          ),
30.
31.
          body: ListView(
32.
            children: const [
              // List 1
33.
34.
              ItemProduk(
                kodeProduk: "A001",
35.
                namaProduk: "Kulkas",
36.
37.
                harga: 2500000,
38.
              ),
39.
               // List 2
              ItemProduk(
40.
                kodeProduk: "A002",
41.
                namaProduk: "TV",
42.
43.
                harga: 5000000,
44.
              ),
45.
              // List 3
46.
              ItemProduk(
                kodeProduk: "A003",
namaProduk: "Mesin Cuci",
47.
48.
49.
                harga: 1500000,
50.
              ),
51.
            ],
52.
          ),
53.
        );
54.
      }
55.}
56.
57. class ItemProduk extends StatelessWidget {
      final String? kodeProduk;
58.
59.
      final String? namaProduk;
60.
      final int? harga;
61.
62.
      const ItemProduk({Key? key, this.kodeProduk, this.namaProduk, this.harga})
          : super(key: key);
63.
64.
65.
      @override
      Widget build(BuildContext context) {
66.
67.
        return GestureDetector(
68.
          child: Card(
69.
            child: ListTile(
70.
              title: Text(namaProduk.toString()),
              subtitle: Text(harga.toString()),
71.
72.
            ),
73.
          ),
74.
          onTap: () {
75.
            Navigator.push(
76.
                context,
                MaterialPageRoute(
77.
78.
                     builder: (context) => ProdukDetail(
79.
                           kodeProduk: kodeProduk,
80.
                           namaProduk: namaProduk,
                           harga: harga,
81.
82.
                         )));
83.
          },
84.
        );
85.
      }
86.}
```



Membuat projek flutter yang terhubung dengan API

Apa itu API

API atau Application Programming Interface adalah sebuah interface yang dapat menghubungkan aplikasi satu dengan aplikasi lainnya. Jadi, API berperan sebagai perantara antar berbagai aplikasi berbeda, baik dalam satu platform yang sama atau lintas platform.

Perumpamaan yang bisa digunakan untuk menjelaskan API adalah seorang pelayan di restoran. Tugas pelayan tersebut adalah menghubungkan tamu restoran dengan juru masak. Tamu cukup memesan makanan sesuai daftar menu yang ada dan pelayan memberitahukannya ke juru masak. Nantinya, pelayan akan kembali ke tamu tadi dengan masakan yang sudah siap sesuai pesanan.

Itulah gambaran tugas dari API dalam pengembangan aplikasi.



Sumber : codepolitan.com

Arsitektur API

Ada tiga arsitektur API yang sering digunakan oleh developer dalam pembangunan aplikasi. Arsitektur ini berkaitan pada bentuk data yang dikirim. Adapun Arsitektur API yang sering digunakan adalah 1. RPC

RPC merupakan teknologi untuk membuat komunikasi antara client side dan server side bisa dilakukan dengan konsep sederhana.

RPC memiliki dua jenis, yaitu XML-RPC dan JSON-RPC. Sesuai namanya, XML-RPC menggunakan format XML sebagai media perpindahan data, sedangkan JSON-RPC menggunakan JSON untuk perpindahan data.

2. SOAP

Arsitektur API lainnya adalah SOAP (Simple Object Access Protocol). Arsitektur ini menggunakan XML (Extensible Markup Language) yang memungkinkan semua data disimpan dalam dokumen. 3. REST

REST atau Representational State Transfer adalah arsitektur API yang cukup populer karena kemudahan penggunaannya. Tak perlu coding yang panjang untuk menggunakannya.

REST menggunakan JSON sebagai bentuk datanya sehingga lebih ringan. Performa aplikasi pun menjadi lebih baik.

Membuat projek Toko API (Restful API)

Installasi Apache, MySql dan PHP (XAMPP)

XAMPP adalah perangkat lunak untuk membuat komputer menjadi web server. XAMPP dapat

di download melalui link url:

<u>)</u>.

Setelah itu, double klik file xampp yang baru di download. Jika muncul pesan error seperti gambar dibawah, abaikan saja dan lanjutkan klik tombol OK.



Berikutnya akan muncul jendela Setup-XAMPP. Klik tombol Next untuk melanjutkan proses

berikutnya.

	Setup - XAMPP Welcome to the XAMPP Setup Wizard.		Ŷ
💽 bitnami			

Selanjutnya akan muncul jendela Select Components, yang meminta untuk memilih aplikasi

yang akan diinstall. Centang saja semua kemudian klik tombol Next.

😫 Setup		_		×
Select Components				8
Select the components you want to install; dea Next when you are ready to continue.	r the components you	ı do not want t	o install.	Click
Apache Apache MySQL FileZilla FTP Server Mercury Mail Server Tomcat Program Languages PHP Perl Program Languages phpMyAdmin Webalizer Fake Sendmail	Click on a componer	nt to get a det	ailed des	cription
AAMINY INSTAILER	< Back	Next >	Ca	ncel

Kemudian akan muncul jendela Installation Folder, dimana anda diminta untuk menentukan lokasi penyimpanan folder xampp, secara bawaan akan diarahkan ke lokasi c:\xampp. Jika anda

ingin menyimpannya di folder lain, anda dapat menekan tombol bergambar folder (**Browse**), kemudian klik tombol **Next**.

🖾 Setup			<u> 84 –</u>		×
Installation f	older				ខា
Please, choose	a folder to install XAMPP				
Select a folder	C:\xampp	12			
XAMPP Installer -					
		< Back	Next >	Ca	ncel

Kita akan menjumpai jendela tawaran untuk mempelajari lebih lanjut tentang Bitnami. Silahkan checklist jika ingin mempelajari lebih lanjut. Bitnami adalah pustaka dari aplikasi client server yang populer seperti misalnya CMS WordPress atau Drupal, dengan penawaran kemudahan dalam installasi hanya dengan satu klik. Kemudian klik tombol **Next.**



Kemudian muncul jendela **Ready To Install** yang menunjukkan xampp sudah siap di install. Kemudian klik tombol **Next.**
🕅 Setup	_		×
Ready to Install			8
Setup is now ready to begin installing XAMPP on your computer.			
XAMPP Installer Kake Stack	Next >	Ca	ancel

Dan proses installasi pun berjalan.

Setup	
bitnami for	ХАМРР
Bitnami for XAMPP provides Drupal, Joomla!, WordPress apps on top of your existing	s free installers that can install s and many other popular open source g XAMPP installation.
Learn More	
	Installing
Unpacking files	
XAMPP Installer	
r ren n.e. ar ið Sallia	< Back Next > Cancel

Tunggu hingga proses install selesai dan muncul jendela sebagai berikut.

📴 Setup	
	Completing the XAMPP Setup Wizard Setup has finished installing XAMPP on your computer. Do you want to start the Control Panel now?
bitnami	< Back Finish Cancel

Klik tombol Finish setelah itu akan muncul jendela Xampp Control Panel yang berguna untuk

menjalankan server.

🔀 ХАМРР	Control Panel	v3.2.1 [Compiled	d: May 7th 2013]				-		×
8	XAI	MPP Contro	ol Panel v3	.2.1				🅜 Co	onfig
Modules Service	Module	PID(s)	Port(s)	Actions				🍥 Ne	tstat
	Apache			Start	Admin	Config	Logs	🗾 🔤 S	hell
	MySQL			Start	Admin	Config	Logs	🔁 Exp	plorer
	FileZilla			Start	Admin	Config	Logs	🚽 🛃	rvices
	Mercury			Start	Admin	Config	Logs	O H	lelp
	Tomcat			Start	Admin	Config	Logs		Quit
16.10.21 [16.10.21] 16.10.21 [16.10.21] 16.10.21 [16.10.24] 16.10.24 [16.10.24] 16.10.24]	main] main] main] main] main] main] main] main]	most application there will be a stabut running t XAMPP Installa Checking for pr All prerequisite Initializing Mod Starting Check Control Panel F	on stuff but when security dialogue his application w ation Directory: " erequisites s found ules -Timer Ready	ever you do e or things w vith administ c:\xampp\"	something wi ill break! So t rator rights!	ith services think			~

Klik tombol Start pada kolom Actions untuk module Apache dan MySQL.

🙁 XAMPP	Control Panel	v3.2.1 [Compil	ed: May 7th 2013]				-		×
ខា	XA	MPP Cont	rol Panel v3	.2.1				de Co	onfig
Modules Service	Module	PID(s)	Port(s)	Actions				Net	tstat
	Apache	5300 5628	80, 443	Stop	Admin	Config	Logs	🗾 S	hell
	MySQL	4928		Stop	Admin	Config	Logs	🔁 Exp	plorer
	FileZilla			Start	Admin	Config	Logs	🦻 Ser	vices
	Mercury			Start	Admin	Config	Logs	😡 H	lelp
	Tomcat			Start	Admin	Config	Logs		Quit
16.10.21 16.10.24 16.10.24 16.10.24 16.10.24 16.12.15 16.12.15 16.12.16 16.12.16	main] main] main] main] Apache] Apache] mysql] mysql]	Checking for All prerequisit Initializing Mo Starting Chec Control Panel Attempting to Status chang Attempting to Status chang	orerequisites es found dules k-Timer Ready start Apache app e detected: runnir start MySQL app e detected: runnir) 19 19					~

Untuk mengetahui apakah installasi telah berhasil atau tidak, ketikkan 'localhost/' pada browser, jika berhasil akan muncul halaman seperti gambar dibawah.

Apache Friends	Applications	FAQs	HOW-TO Guides	PHPInfo	phpMyAdmin
🔁 XAMPP Apac	he + MySC	QL -	+ PHP	+ F	Perl
Welcome to XAMPP fo	r Windows 5.6	6.12			
You have successfully installed XAMPP on this system/ more info in the FAQs section or check the HOW-TO Gu	Now you can start using Apache, ides for getting started with PHP	MySQL, applicati	PHP and other con ons	nponents. Y	/ou can find

Start the XAMPP Control Panel to check the server status.

Install Postman

Postman adalah sebuah aplikasi fungsinya adalah sebagai REST Client atau istilahnya adalah aplikasi yang digunakan untuk melakukan uji coba REST API yang telah kita buat. Postman ini merupakan tools wajib bagi para developer yang bergerak pada pembuatan API, fungsi utama postman ini adalah sebagai GUI API Caller Pemanggil. namun sekarang postman juga menyediakan fitur lain yaitu Sharing Collection API for Documentation (free), Testing API (free), Realtime Collaboration Team (paid), Monitoring API (paid), Integration (paid).

Postman tersedia sebagai aplikasi asli untuk sistem operasi macOS, Windows (32-bit dan 64-bit), dan Linux (32-bit dan 64-bit). Untuk mendapatkan aplikasi Postman, dapat diunduh

pada website resminya yanitu getpostman.com atau dapat diunduk pada halaman https://www.postman.com/downloads/

Setelah berhasil mengunduh paket instalasi postman, kemudian jalankan dengan cara klik dua kali. Pilih run jika muncul pop up seperti berikut :



Kemudian tunggu hingga proses instalasi selesai dan muncul seperti gambar berikut



API SPEC

API SPEC ini bermaksud untuk membuat standar API sebagai dokumentasi kepada pengembang baik itu frontend maupun backend

A. Registrasi

EndPoint	/registrasi
Method	POST

Header	Content-Type: application/json
Body	<pre>{ "nama" : "string", "email" : "string, unique", "password" : "string" }</pre>
Response	<pre>{ "code" : "integer", "status" : "boolean", "data" : "string" }</pre>

B. Login

EndPoint	/login
Method	POST
Header	Content-Type: application/json
Body	<pre>{ "email" : "string" "password" : "string" }</pre>
Response	<pre>{ "code" : "integer", "status" : "boolean", "data" : { "token" : "string", "user" : { "id" : "integer", "email" : "string", } } }</pre>

C. Produk

1. List Produk

EndPoint	/produk
Method	GET
Header	Content-Type: application/json
Response	<pre>{ "code" : "integer", "status" : "boolean", "data" : [</pre>

	{	
		"id" : "integer",
		"kode produk" : "string",
		"nama produk" : "string",
		"harga" : "integer",
	}.	0 3 7
	{	
	, i	"id" : "integer",
		"kode produk" : "string",
		"nama produk" : "string",
		"harga" : "integer".
	}	
1	J	
3		
J		

2. Create Produk

EndPoint	/produk
Method	POST
Header	Content-Type: application/json
Body	<pre>{ "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer" }</pre>
Response	<pre>{ "code" : "integer", "status" : "boolean", "data" : { "id" : "integer", "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer", } }</pre>

3. Update Produk

EndPoint	/produk/{id}/update			
Method	POST			
Header	Content-Type: application/json			
Body	<pre>{ "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer" }</pre>			

Response	{
	"code" : "integer",
	"status" : "boolean",
	"data" : "boolean"
	}

4. Show Produk

EndPoint	/produk/{id}
Method	GET
Header	Content-Type: application/json
Response	<pre>{ "code" : "integer", "status" : "boolean", "data" : { "id" : "integer", "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer", } }</pre>

5. Delete Produk

EndPoint	/produk/{id}
Method	DELETE
Header	Content-Type: application/json
Response	<pre>{ "code" : "integer", "status" : "boolean", "data" : "boolean" }</pre>

Pembuatan Database

Buat database dengan nama : toko_api

Kemudian buat tabel-tabel dengan perintah sebagai berikut :

1. SQL membuat tabel member

```
CREATE table member (
id INT NOT NULL AUTO_INCREMENT,
```

```
nama VARCHAR(255) NOT NULL,
email VARCHAR(255) NOT NULL,
password VARCHAR(255) NOT NULL,
PRIMARY KEY(id)
```

);

#	Name	Туре	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	None		AUTO_INCREMENT
2	nama	varchar(255)	utf8mb4_general_ci		No	None		
3	email	varchar(255)	utf8mb4_general_ci		No	None		
4	password	varchar(255)	utf8mb4_general_ci		No	None		

2. SQL membuat tabel member_token

```
CREATE table member_token (

id INT NOT NULL AUTO_INCREMENT,

member_id INT NOT NULL,

auth_key VARCHAR(255) NOT NULL,

FOREIGN KEY (member_id) REFERENCES member(id) on update cascade on delete no action,

PRIMARY KEY(id)
```

);

#	Name	Туре	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	None		AUTO_INCREMENT
2	member_id 🏈	int(11)			No	None		
3	auth_key	varchar(255)	utf8mb4_general_ci		No	None		

3. SQL membuat tabel produk

```
CREATE table produk (
id INT NOT NULL AUTO_INCREMENT,
kode_produk VARCHAR(255) NOT NULL,
nama_produk VARCHAR(255) NOT NULL,
harga INT NOT NULL,
PRIMARY KEY(id)
```

);

#	Name	Туре	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔎	int(11)			No	None		AUTO_INCREMENT
2	kode_produk	varchar(255)	utf8mb4_general_ci		No	None		
3	nama_produk	varchar(255)	utf8mb4_general_ci		No	None		
4	harga	int(11)			No	None		

Installasi Codelgniter 4 sebagai Restful API

Selanjutnya install projek Codelgniter pada web server (pada folder C:\xampp\htdocs)

Framework CodeIgniter 4 dapat di unduh di https://codeigniter.com/download

odelgniter 🔥	Download Documentation Community Contribute
Codelgniter has two supported versions: Code	elgniter 4 (current) and Codelgniter 3 (legacy)
Codelgniter 4	Codelgniter 3
Codelgniter 4 is the current version of the framework, intended for use with PHP 7.3+ (including 8.0).	Codelgniter 3 is the legacy version of the framework, intended for use with PHP 5.6+.
Development is ongoing, and the current version is v4.1.4.	This version is in maintenance, receiving mostly just security updates,
You "could" download this version of the framework using the button below, but we encourage you to check the Installation section of the User Guide, to choose from several different ways you can install the framework :)	and the current version is 3.1.11.
± Download ♦ Discuss ☐ Sources • Translations	
Codelg	initer 2
CodeIgniller 2.2.6 is the legacy version	on of the framework.
The 2.x branch was originally release version (2.2.5) came out in October,	ed January 2011, and the last 2015.
Codelgniter 2 has reached its end-or October 31, 2015. No further update	r-life for support and updates, as of is are planned.
▲ Dow	vintoad

Setelah di unduh ekstrak file CodeIgniter pada folder C:\xampp\htdocs

→ This PC → Local Disk (C:) → xampp → htdocs					
	Open				
	📜 Open with WinRAR				
	📜 Extract files				
	َ Extract Here				
framework-4.1.4	Extract to framework-4.1.4\				
zip	🖶 Scan with Microsoft Defender				
	🖻 Share				
	Open with	>			
	TeraCopy				
	New folder with selection				
	Restore previous versions				
	Send to	$\overline{}$			
	Send to				
	Cut				

Kemudian ubah nama folder menjadi **toko-api.** Pada projek buka file **Database.php** yang terletak pada folder **app\config**, kemudian cari kode berikut

```
public $default = [
        'DSN' => '',
        'hostname' => 'localhost',
        'username' => '',
'password' => '',
        'database' => '',
        'DBDriver' => 'MySQLi',
        'DBPrefix' => '',
         'pConnect' => false,
        'DBDebug' => (ENVIRONMENT !== 'production'),
        'charset' => 'utf8',
        'DBCollat' => 'utf8 general ci',
         'swapPre' => '',
         'encrypt' => false,
         'compress' => false,
         'strictOn' => false,
        'failover' => [],
'port' => 3306,
];
```

Untuk menghubungkan projek dengan database kita akan mengisikan hostname,

username, password dan database menjadi

```
public $default = [
    'DSN' => '',
    'hostname' => 'localhost',
```

```
'username' => 'root',
'password' => '',
'database' => 'toko_api',
'DBDriver' => 'MySQLi',
'DBPrefix' => '',
'pConnect' => false,
'DBDebug' => (ENVIRONMENT !== 'production'),
'charset' => 'utf8',
'DBCollat' => 'utf8_general_ci',
'swapPre' => '',
'encrypt' => false,
'compress' => false,
'strictOn' => false,
'failover' => [],
'port' => 3306,
```

];

Membuat hasil response

Buat sebuah file dengan nama RestfulControlller.php pada folder app\Controllers



Kemudian tambahkan kode pada file tersebut sehingga menjadi



```
15. 'code' => $code,
16. 'status' => $status,
17. 'data' => $data
18. ]);
19. }
20. }
```

Registrasi

Membuat model Registrasi

Buat sebuah file dengan nama MRegistrasi.php pada folder app/Models

∨ ТОКО-АРІ					
🗸 🗰 app					
> 📪 Config					
> 🌅 Controllers					
> 📑 Database					
> 🖿 Filters					
> 🤛 Helpers					
> 🔜 Language					
> 📑 Libraries					
🗸 🔄 Models					
🚸 .gitkeep					
🐵 MRegistrasi.php					
> 🖿 ThirdParty					
> 崎 Views					
👌 .htaccess					

Kemudian pada file **MRegistrasi.php** ketikkan kode berikut

```
1. <?php
2.
3. namespace App\Models;
4.
5. use CodeIgniter\Model;
6.
7. class MRegistrasi extends Model
8. {
9. protected $table = 'member';
10. protected $table = ['nama', 'email', 'password'];
11. }</pre>
```

Membuat controller Registrasi

Buat sebuah file dengan nama RegistrasiController.php pada folder app\Controllers



Kemudian pada file RegistrasiController.php tersebut masukkan kode berikut

```
1. <?php
2.
3. namespace App\Controllers;
4.
5. use App\Models\MRegistrasi;
6.
7. class RegistrasiController extends RestfulController
8. {
9.
        public function registrasi()
10.
11.
        {
12.
             $data = [
                 'nama' => $this->request->getVar('nama'),
'email' => $this->request->getVar('email'),
13.
14.
                  'password' => password_hash($this->request->getVar('password'),
15.
    PASSWORD DEFAULT)
16.
             ];
17.
18.
             $model = new MRegistrasi();
             $model->save($data);
19.
20.
             return $this->responseHasil(200, true, "Registrasi Berhasil");
21.
        }
22.}
```

Menambah route Registrasi

Buka file Routes.php pada folder app\Config

Kemudian lihat baris ke 34, telah terdapat routing dengan method get, kita akan menambahkan

routing untuk registrasi sehingga kode menjadi seperti berikut

Pada baris terakhir kita menambahkan routing registrasi agar dapat diakses dengan method POST. Untuk mengakses registrasi, kita gunakan Postman dengan alamat url **localhost/tokoapi/public/registrasi** dengan method **POST**

Adapun langkah menggunakan postman untuk menguji Rest API yang telah dibuat adalah sebagai berikut:

- 1. Buka aplikasi Postman yang telah terinstall
- 2. Masukkan alamat url untuk melakukan registrasi toko-api yang telah kita buat yaitu http://localhost/toko-api/public/registrasi
- 3. Selanjutnya pilih pengujian dengan type POST
- 4. Kemudian klik Body yang berada pada bagian bawah inputan url, kemudian pilih raw
- 5. Kemudian dibagian bawah pilihan raw, pilih JSON (application/json)



6. Kemudian isi dengan format JSON sesuai dengan field request pada RegistrasiController pada fungsi registrasi yaitu nama, email, password kemudian klik tombol Send

```
$data = [
                'nama' => $this->request->getPost('nama'),
                'email' => $this->request->getPost('email'),
               'password' => password hash($this->request-
>getPost('password'), PASSWORD DEFAULT)
          ];
Contoh isian data
{
       "nama": "Administrator",
       "email":"admin@admin.com",
        "password":"admin"
}
  POST V localhost/toko-api/public/registrasi
                                                       Save 🗸
                                            Send
 Auth Headers (2) Body 😐 Pre-req. Tests
                                                                 Bodv
  form-data • x-www-form-urlencoded • raw • binary
                                                                  Pretty
                                                                        "code": 200,
                                                                        "status": true,
"data": "Registrasi Berhasil"
        "nama":"Administrator",
```

Login

Membuat model Member

"email":"admin@admin.com", "password":"admin"

Buat sebuah file dengan nama MMember.php pada folder app\Models dan ketikkan kode

berikut

```
1. <?php
2.
namespace App\Models;
4.
use CodeIgniter\Model;
6.
7. class MMember extends Model
8. {
       protected $table = 'member';
9.
10.}
```

Membuat model Login

Buat sebuah file dengan nama MLogin.php pada folder app/Models dan ketikkan kode

berikut

json 🗸

```
1. <?php
2.
3. namespace App\Models;
4.
5. use CodeIgniter\Model;
6.
7. class MLogin extends Model
8. {
9. protected $table = 'member_token';
10. protected $allowedFields = ['member_id', 'auth_key'];
11. }</pre>
```

Membuat controller Login

Buat sebuah file dengan nama LoginController.php pada folder app/Controllers dan ketikkan

kode berikut

```
1. <?php
2.
namespace App\Controllers;
4.
5. use App\Models\MLogin;
6. use App\Models\MMember;
7.
8. class LoginController extends RestfulController
9. {
10.
11.
        public function login()
12.
        {
13.
            $email = $this->request->getVar('email');
14.
            $password = $this->request->getVar('password');
15.
16.
            $model = new MMember();
17.
            $member = $model->where(['email' => $email])->first();
18.
            if (!$member) {
                return $this->responseHasil(400, false, "Email tidak ditemukan");
19.
20.
            }
            if (!password_verify($password, $member['password'])) {
21.
                return $this->responseHasil(400, false, "Password tidak valid");
22.
23.
            }
24.
            $login = new MLogin();
25.
            $auth key = $this->RandomString();
26.
27.
            $login->save([
                'member_id' => $member['id'],
28.
                'auth_key' => $auth_key
29.
30.
            1);
            $data = [
31.
                'token' => $auth_key,
32.
                'user' => [
33.
34.
                    'id' => $member['id'],
35.
                     'email' => $member['email'],
36.
37.
            ];
38.
            return $this->responseHasil(200, true, $data);
39.
        }
40.
41.
        private function RandomString($length = 100)
42.
        {
43.
            $karakkter
    '012345678dssd9abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
44.
            $panjang_karakter = strlen($karakkter);
45.
            $str = '';
```

Menambahkan route Login

Pada route tambahkan kode untuk mengakses login sehingga menjadi sebagai berikut

```
$routes->post('/registrasi', 'RegistrasiController::registrasi');
$routes->post('/login', 'LoginController::login');
```

Mencoba Rest

Silahkan coba Rest API dengan memasukkan url http://localhost/toko-api/public/login dengan

method POST



CRUD Produk

Membuat model Produk

Buat sebuah file dengan nama MProduk.php pada folder app/Models dan ketikkan kode

berikut

```
1. <?php
2.
3. namespace App\Models;
4.
5. use CodeIgniter\Model;
6.
7. class MProduk extends Model
8. {
9. protected $table = 'produk';
10. protected $table = 'id';
11. protected $allowedFields = ['kode_produk', 'nama_produk', 'harga'];
12. }</pre>
```

Membuat controller produk

Buat sebuah file dengan nama ProdukController pada folder app/Controllers dan ketikkan kode

berikut

```
1. <?php
2.
namespace App\Controllers;
4.
5. use App\Models\MProduk;
6.
7. class ProdukController extends RestfulController
8. {
9.
10.}
```

Selanjutnya pada class ProdukController kita akan menambahkan fungsi (function) CRUD

produk, yaitu:

```
Membuat fungsi create produk
public function create()
       $data = [
            'kode produk' => $this->request->getVar('kode produk'),
           'nama produk' => $this->request->getVar('nama produk'),
           'harga' => $this->request->getVar('harga')
       ];
       $model = new MProduk();
       $model->insert($data);
       $produk = $model->find($model->getInsertID());
       return $this->responseHasil(200, true, $produk);
Membuat fungsi list produk
public function list()
     $model = new MProduk();
     $produk = $model->findAll();
     return $this->responseHasil(200, true, $produk);
}
Membuat fungsi tampil produk
public function detail($id)
     $model = new MProduk();
     $produk = $model->find($id);
     return $this->responseHasil(200, true, $produk);
}
Membuat fungsi update produk
public function ubah($id)
    $data = [
        'kode produk' => $this->request->getVar('kode produk'),
        'nama produk' => $this ->request->getVar('nama_produk'),
        'harga' => $this->request->getVar('harga')
    ];
    $model = new MProduk();
    $model->update($id, $data);
    $produk = $model->find($id);
    return $this->responseHasil(200, true, $produk);
}
```

```
Membuat fungsi delete produk
public function hapus($id)
{
    $model = new MProduk();
    $produk = $model->delete($id);

    return $this->responseHasil(200, true, $produk);
}
```

Sehingga adapun keseluruhan kode ProdukController.php adalah sebagai berikut

```
1. <?php
2.
namespace App\Controllers;
4.
5. use App\Models\MProduk;
6.
7. class ProdukController extends RestfulController
8. {
9.
        public function create()
10.
        {
11.
            $data = [
12.
                'kode_produk' => $this->request->getVar('kode_produk'),
                'nama_produk' => $this->request->getVar('nama_produk'),
13.
14.
                'harga' => $this->request->getVar('harga')
15.
            ];
16.
17.
            $model = new MProduk();
18.
            $model->insert($data);
19.
            $produk = $model->find($model->getInsertID());
20.
            return $this->responseHasil(200, true, $produk);
21.
        }
22.
23.
        public function list()
24.
        {
25.
            $model = new MProduk();
26.
            $produk = $model->findAll();
27.
            return $this->responseHasil(200, true, $produk);
28.
        }
29.
30.
        public function detail($id)
31.
        ł
32.
            $model = new MProduk();
33.
            $produk = $model->find($id);
            return $this->responseHasil(200, true, $produk);
34.
35.
        }
36.
        public function ubah($id)
37.
38.
        {
39.
            data = [
40.
                'kode_produk' => $this->request->getVar('kode_produk'),
                'nama_produk' => $this->request->getVar('nama_produk'),
41.
42.
                'harga' => $this->request->getVar('harga')
43.
            ];
44.
            $model = new MProduk();
45.
46.
            $model->update($id, $data);
47.
            $produk = $model->find($id);
48.
49.
            return $this->responseHasil(200, true, $produk);
50.
```

```
51.
52. public function hapus($id)
53. {
54. $model = new MProduk();
55. $produk = $model->delete($id);
56.
57. return $this->responseHasil(200, true, $produk);
58. }
59. }
```

Menambahkan route produk

Agar ProdukController dapat diakses, selanjutnya tambah route untuk mengakses produk pada

file app/Config/Routes.php

```
$routes->group('produk', function ($routes) {
    $routes->post('/', 'ProdukController::create');
    $routes->get('/', 'ProdukController::list');
    $routes->get('(:segment)', 'ProdukController::detail/$1');
    $routes->put('(:segment)', 'ProdukController::ubah/$1');
    $routes->delete('(:segment)', 'ProdukController::hapus/$1');
});
```

Adapun keseluruhan kode pada app/Config/Routes.php adalah sebagai berikut

```
$routes->get('/', 'Home::index');
$routes->post('/registrasi', 'RegistrasiController::registrasi');
$routes->post('/login', 'LoginController::login');
$routes->group('produk', function ($routes) {
    $routes->post('/', 'ProdukController::create');
    $routes->get('/', 'ProdukController::list');
    $routes->get('(:segment)', 'ProdukController::detail/$1');
    $routes->put('(:segment)', 'ProdukController::ubah/$1');
    $routes->delete('(:segment)', 'ProdukController::hapus/$1');
});
```

Mencoba Rest

Create Produk (localhost/toko-api/public/produk) dengan method POST



List Produk (localhost/toko-api/public/produk) dengan method GET





GET 🗸	localhost/toko-api/public/produk/1	Params	Send 🗠	Save 🗡			
Auth Head					Body Cook		
Туре	No Auth 🗸				Pretty Ra	aw Preview JSON 🛩 🚍	5 🖸 🔍 Save Response
					1 (3 "c 4 "d 6 7 8 9 10 10	ode": 200, status": frue, fata": ("id": "1", "A001", "kode_produk": "A001", "nama_produk": "Realme Smart T "harga": "5700000"	V 50 Inch",

Update Produk (localhost/toko-api/public/produk/{id}) dengan method PUT



Delete Produk (localhost/toko-api/public/produk) dengan method DELETE

DEL 🛩 localhost/toko-api/aublic/produk/3 Params Send 🌱 Save 🛩	
urth Headers (2) Body Pre-req. Tests Code	Body Cookies Headers (8) Fast Results
● form-data ● x www-form-urlencoded ● raw ● binary JSON (application/json) ∨	Pretty Raw Preview JSON V 📅 🖺 Q. Sove Response
	1 - { 2 "code": 200, 3 "status": true, 4 "data": true 5 }

Membuat projek flutter tokokita

Buat sebuah projek flutter dengan nama tokokita



Membuat Model Buat folder dengan nama model pada folder lib



Login

Buat sebuah file dengan nama login.dart pada folder model. Kemudian masukkan kode berikut

```
1. class Login {
      int? code;
2.
3.
      bool? status;
4.
      String? token;
      int? userID;
5.
      String? userEmail;
6.
7.
8.
      Login({this.code, this.status, this.token, this.userID, this.userEmail});
9.
10.
    factory Login.fromJson(Map<String, dynamic> obj) {
11.
        return Login(
            code: obj['code'],
12.
13.
             status: obj['status'],
            token: obj['data']['token'],
userID: obj['data']['user']['id'],
14.
15.
```

```
16. userEmail: obj['data']['user']['email']);
17. }
18. }
```

Registrasi

Buat sebuah file dengan nama registrasi.dart pada folder model. Kemudian masukkan kode

berikut

```
1. class Registrasi {
2.
      int? code;
      bool? status;
З.
4.
      String? data;
5.
6.
      Registrasi({this.code, this.status, this.data});
7.
      factory Registrasi.fromJson(Map<String, dynamic> obj) {
8.
9.
        return Registrasi(
         code: obj['code'],
status: obj['status'],
10.
11.
           data: obj['data']);
12.
13.
      }
14. }
```

Produk

Buat sebuah file dengan nama produk.dart pada folder model. Kemudian ketikkan kode berikut

```
1. class Produk {
2. int? id;
      String? kodeProduk;
3.
4.
      String? namaProduk;
5.
      int? hargaProduk;
6.
      Produk({this.id, this.kodeProduk, this.namaProduk, this.hargaProduk});
7.
8.
9.
      factory Produk.fromJson(Map<String, dynamic> obj) {
10.
        return Produk(
          code: obj['id'],
kodeProduk: obj['kode_produk'],
11.
12.
          namaProduk: obj['nama_produk'],
13.
          hargaProduk: obj['harga']
14.
15.
        );
16.
      }
17.}
```

Membuat Halaman

Pertama kita akan memecah bagian-bagian kode menjadi beberapa bagian, adapun untuk tampilan, dikelompokkan kedalam folder **ui**.



Registrasi

Buat sebuah file dengan nama **registrasi_page.dart** pada folder **ui**.



Pada file registrasi_page.dart ketikkan kode berikut

```
1. import 'package:flutter/material.dart';
2.
3. class RegistrasiPage extends StatefulWidget {
4.
      const RegistrasiPage({Key? key}) : super(key: key);
5.
      @override
6.
7.
      _RegistrasiPageState createState() => _RegistrasiPageState();
8. }
9.
10. class _RegistrasiPageState extends State<RegistrasiPage> {
     final _formKey = GlobalKey<FormState>();
bool _isLoading = false;
11.
12.
13.
14.
     final _namaTextboxController = TextEditingController();
15.
      final _emailTextboxController = TextEditingController();
16.
     final _passwordTextboxController = TextEditingController();
17.
18.
     @override
19.
     Widget build(BuildContext context) {
20.
        return Scaffold(
21.
          appBar: AppBar(
22.
            title: const Text("Registrasi"),
23.
          ).
          body: SingleChildScrollView(
24.
25.
            child: Padding(
              padding: const EdgeInsets.all(8.0),
26.
27.
              child: Form(
                key: _formKey,
28.
                child: Column(
29.
```

```
30.
                  mainAxisAlignment: MainAxisAlignment.center,
31.
                  children: [
                    _namaTextField(),
32.
33.
                    _emailTextField(),
34.
                    _passwordTextField(),
35.
                    _passwordKonfirmasiTextField(),
                    _buttonRegistrasi()
36.
37.
                  ],
38.
               ),
39.
              ),
            ),
40.
41.
          ),
42.
        );
43.
      }
44.
45.
      //Membuat Textbox Nama
46.
     Widget _namaTextField() {
47.
        return TextFormField(
48.
          decoration: const InputDecoration(labelText: "Nama"),
49.
          keyboardType: TextInputType.text,
50.
          controller: _namaTextboxController,
          validator: (value) {
51.
            if (value!.length < 3) {</pre>
52.
53.
              return "Nama harus diisi minimal 3 karakter";
54.
            }
55.
            return null;
56.
          },
57.
        );
58.
      }
59.
60.
      //Membuat Textbox email
61.
      Widget _emailTextField() {
62.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Email"),
63.
64.
          keyboardType: TextInputType.emailAddress,
65.
          controller: emailTextboxController,
66.
          validator: (value) {
67.
            //validasi harus diisi
68.
            if (value!.isEmpty) {
69.
              return 'Email harus diisi';
70.
            }
71.
            //validasi email
72.
            Pattern pattern =
                r'^{(([^{>})()[]),;:\s@^{]}+(.[^{>})()[]),;:\s@^{]}+)*)|(^{.+}))@((([[0])))
73.
    -9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-
    Z]{2,}))$';
74.
            RegExp regex = RegExp(pattern.toString());
75.
            if (!regex.hasMatch(value)) {
76.
              return "Email tidak valid";
77.
            }
78.
            return null;
79.
          },
80.
        );
81.
      }
82.
83.
      //Membuat Textbox password
84.
      Widget passwordTextField() {
85.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Password"),
86.
          keyboardType: TextInputType.text,
87.
88.
          obscureText: true,
89.
          controller: _passwordTextboxController,
90.
          validator: (value) {
91.
            //jika karakter yang dimasukkan kurang dari 6 karakter
92.
            if (value!.length < 6) {</pre>
```

```
93.
              return "Password harus diisi minimal 6 karakter";
94.
            }
95.
            return null;
96.
          },
97.
        );
98.
      }
99.
100.
             //membuat textbox Konfirmasi Password
101.
             Widget _passwordKonfirmasiTextField() {
102.
               return TextFormField(
103.
                 decoration: const InputDecoration(labelText: "Konfirmasi Password"),
104.
                 keyboardType: TextInputType.text,
105.
                 obscureText: true,
106.
                 validator: (value) {
107.
                    //jika inputan tidak sama dengan password
108.
                   if (value != _passwordTextboxController.text) {
                     return "Konfirmasi Password tidak sama";
109.
110.
                   }
111.
                   return null;
112.
                 },
113.
               );
114.
             }
115.
116.
             //Membuat Tombol Registrasi
117.
             Widget _buttonRegistrasi() {
118.
               return ElevatedButton(
                   child: const Text("Registrasi"),
119.
120.
                   onPressed: () {
                     var validate = _formKey.currentState!.validate();
121.
122.
                   });
123.
             }
124.
           }
```

Untuk mencoba halaman **registrasi_page**, buka file **main.dart** kemudian ubah kode menjadi seperti berikut ini

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/ui/registrasi_page.dart';
3.
4. void main() {
5.
      runApp(const MyApp());
6. }
7.
8. class MyApp extends StatelessWidget {
9.
      const MyApp({Key? key}) : super(key: key);
10.
11.
      @override
12.
     Widget build(BuildContext context) {
13.
        return const MaterialApp(
14.
          title: 'Toko Kita',
15.
          debugShowCheckedModeBanner: false,
16.
          home: RegistrasiPage(),
17.
        );
18. }
19.}
```

Dan hasilnya seperti berikut

2:18	LTE 🗹 🗋
Registrasi	
Nama	
Email	
Password	
Konfirmasi Password	
Registrasi	

Login

Buat sebuah file dengan nama login_page.dart pada folder ui dengan kode berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/ui/registrasi_page.dart';
3.
4. class LoginPage extends StatefulWidget {
     const LoginPage({Key? key}) : super(key: key);
5.
6.
7.
     @override
8.
     _LoginPageState createState() => _LoginPageState();
9.}
10.
11. class LoginPageState extends State<LoginPage> {
12. final _formKey = GlobalKey<FormState>();
13.
     bool _isLoading = false;
14.
     final _emailTextboxController = TextEditingController();
15.
16. final passwordTextboxController = TextEditingController();
17.
18.
     @override
19.
     Widget build(BuildContext context) {
20.
       return Scaffold(
21.
         appBar: AppBar(
           title: const Text('Login'),
22.
23.
          ),
24.
         body: SingleChildScrollView(
25.
           child: Padding(
26.
              padding: const EdgeInsets.all(8.0),
27.
              child: Form(
28.
               key: _formKey,
```

```
29.
                child: Column(
                  children: [
30.
                     _emailTextField(),
31.
                    _passwordTextField(),
32.
33.
                    _buttonLogin(),
                    const SizedBox(
34.
                      height: 30,
35.
36.
                    ),
37.
                    _menuRegistrasi()
38.
                  ],
39.
                ),
40.
              ),
41.
            ),
42.
          ),
43.
        );
      }
44.
45.
      //Membuat Textbox email
46.
      Widget _emailTextField() {
47.
48.
        return TextFormField(
49.
          decoration: const InputDecoration(labelText: "Email"),
50.
          keyboardType: TextInputType.emailAddress,
51.
          controller: _emailTextboxController,
          validator: (value) {
52.
53.
            //validasi harus diisi
54.
            if (value!.isEmpty) {
55.
              return 'Email harus diisi';
56.
            }
57.
            return null;
58.
          },
59.
        );
60.
     }
61.
62.
      //Membuat Textbox password
63.
      Widget _passwordTextField() {
64.
        return TextFormField(
65.
          decoration: const InputDecoration(labelText: "Password"),
66.
          keyboardType: TextInputType.text,
67.
          obscureText: true,
68.
          controller: _passwordTextboxController,
          validator: (value) {
69.
70.
            //jika karakter yang dimasukkan kurang dari 6 karakter
71.
            if (value!.isEmpty) {
72.
              return "Password harus diisi";
73.
            }
74.
            return null;
75.
          },
76.
        );
77.
      }
78.
79.
      //Membuat Tombol Login
     Widget _buttonLogin() {
80.
81.
        return ElevatedButton(
            child: const Text("Login"),
82.
            onPressed: () {
83.
84.
              var validate = _formKey.currentState!.validate();
85.
            });
86.
     }
87.
      // Membuat menu untuk membuka halaman registrasi
88.
89.
      Widget menuRegistrasi() {
90.
        return Center(
91.
          child: InkWell(
92.
            child: const Text(
93.
              "Registrasi",
```

```
94.
             style: TextStyle(color: Colors.blue),
95.
            ),
96.
            onTap: () {
              Navigator.push(context,
97.
98.
                 MaterialPageRoute(builder: (context) => const RegistrasiPage()));
99.
            },
100.
                 ),
101.
               );
             }
102.
           }
103.
```

Untuk mencobanya modifikasi file main.dart dimana pada bagian home akan memanggil

LoginPage()

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/ui/login_page.dart';
3.
4. void main() {
5.
     runApp(const MyApp());
6. }
7.
8. class MyApp extends StatelessWidget {
9.
     const MyApp({Key? key}) : super(key: key);
10.
11.
     @override
12.
     Widget build(BuildContext context) {
13.
       return const MaterialApp(
14.
         title: 'Toko Kita',
15.
         debugShowCheckedModeBanner: false,
16.
         home: LoginPage(),
17.
       );
18.
     }
19.}
```

Pada saat dijalankan akan terdapat link untuk membuka halaman registrasi pada bagian bawah

form

2:40 🗂		LTE 🔟 📋
Login		
Email		
Password		
2	Login	
	Registrasi	

Form Produk

Form produk yang akan kita buat berikut ini memiliki 2 fungsi yaitu untuk menambah data produk dan mengubah data produk

Buat sebuah file dengan nama produk_form.dart pada folder ui dengan kode berikut

```
    import 'package:flutter/material.dart';
    import 'package:tokokita/model/produk.dart';

3.
4. class ProdukForm extends StatefulWidget {
5.
     Produk? produk;
6.
      ProdukForm({Key? key, this.produk}) : super(key: key);
7.
8.
9.
     @override
10.
     _ProdukFormState createState() => _ProdukFormState();
11. }
12.
13. class _ProdukFormState extends State<ProdukForm> {
14. final _formKey = GlobalKey<FormState>();
15.
     bool _isLoading = false;
16. String judul = "TAMBAH PRODUK";
     String tombolSubmit = "SIMPAN";
17.
18.
19.
      final _kodeProdukTextboxController = TextEditingController();
20. final _namaProdukTextboxController = TextEditingController();
21.
     final _hargaProdukTextboxController = TextEditingController();
22.
23.
     @override
24. void initState() {
```

```
25.
        super.initState();
       isUpdate();
26.
27.
      }
28.
29.
      isUpdate() {
30.
        if (widget.produk != null) {
          setState(() {
31.
            judul = "UBAH PRODUK";
32.
            tombolSubmit = "UBAH"
33.
34.
            kodeProdukTextboxController.text = widget.produk!.kodeProduk!;
            _namaProdukTextboxController.text = widget.produk!.namaProduk!;
35.
36.
            hargaProdukTextboxController.text =
37.
                widget.produk!.hargaProduk.toString();
38.
          });
39.
        } else {
          judul = "TAMBAH PRODUK";
40.
          tombolSubmit = "SIMPAN";
41.
42.
        }
43.
     }
44.
45.
      @override
46.
     Widget build(BuildContext context) {
47.
        return Scaffold(
48.
          appBar: AppBar(title: Text(judul)),
49.
          body: SingleChildScrollView(
50.
            child: Padding(
51.
              padding: const EdgeInsets.all(8.0),
52.
              child: Form(
53.
                key: _formKey,
54.
                child: Column(
55.
                  children: [
                    _kodeProdukTextField(),
56.
57.
                     _namaProdukTextField(),
58.
                    _hargaProdukTextField(),
59.
                    _buttonSubmit()
60.
                  ],
61.
                ),
62.
              ),
63.
            ),
64.
          ),
        );
65.
66.
     }
67.
68.
      //Membuat Textbox Kode Produk
69.
      Widget _kodeProdukTextField() {
70.
        return TextFormField(
71.
          decoration: const InputDecoration(labelText: "Kode Produk"),
72.
          keyboardType: TextInputType.text,
73.
          controller: _kodeProdukTextboxController,
          validator: (value) {
74.
75.
            if (value!.isEmpty) {
              return "Kode Produk harus diisi";
76.
77.
            }
78.
            return null;
79.
          },
80.
        );
81.
      }
82.
83.
      //Membuat Textbox Nama Produk
     Widget __namaProdukTextField() {
84.
85.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Nama Produk"),
86.
87.
          keyboardType: TextInputType.text,
88.
          controller: _namaProdukTextboxController,
89.
          validator: (value) {
```

```
90.
            if (value!.isEmpty) {
91.
              return "Nama Produk harus diisi";
92.
            }
93.
            return null;
94.
          },
95.
        );
      }
96.
97.
98.
      //Membuat Textbox Harga Produk
99.
      Widget _hargaProdukTextField() {
100.
               return TextFormField(
101.
                 decoration: const InputDecoration(labelText: "Harga"),
102.
                 keyboardType: TextInputType.number,
103.
                 controller: _hargaProdukTextboxController,
104.
                 validator: (value) {
105.
                   if (value!.isEmpty) {
                     return "Harga harus diisi";
106.
107.
                   }
108.
                   return null;
109.
                 },
110.
               );
111.
             }
112.
113.
             //Membuat Tombol Simpan/Ubah
114.
             Widget _buttonSubmit() {
115.
               return OutlinedButton(
116.
                   child: Text(tombolSubmit),
117.
                   onPressed: () {
                     var validate = _formKey.currentState!.validate();
118.
119.
                   });
120.
             }
121.
           }
```

Detail Produk

Buat sebuah file dengan nama produk_detail.dart pada folder ui dengan kode berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/model/produk.dart';
3. import 'package:tokokita/ui/produk_form.dart';
4.
5. class ProdukDetail extends StatefulWidget {
     Produk? produk;
6.
7.
8.
     ProdukDetail({Key? key, this.produk}) : super(key: key);
9.
10.
     @override
11.
     _ProdukDetailState createState() => _ProdukDetailState();
12. }
13.
14. class _ProdukDetailState extends State<ProdukDetail> {
15.
     @override
16.
     Widget build(BuildContext context) {
17.
       return Scaffold(
18.
         appBar: AppBar(
           title: const Text('Detail Produk'),
19.
20.
         ),
21.
         body: Center(
22.
           child: Column(
23.
             children: [
24.
               Text(
```

```
25.
                   "Kode : ${widget.produk!.kodeProduk}",
26.
                  style: const TextStyle(fontSize: 20.0),
27.
                ),
28.
                Text(
29.
                   "Nama : ${widget.produk!.namaProduk}",
30.
                  style: const TextStyle(fontSize: 18.0),
31.
                ),
32.
                Text(
                   "Harga : Rp. ${widget.produk!.hargaProduk.toString()}",
33.
34.
                  style: const TextStyle(fontSize: 18.0),
35.
                ),
                _tombolHapusEdit()
36.
37.
              ],
38.
            ),
39.
          ),
40.
        );
      }
41.
42.
43.
      Widget _tombolHapusEdit() {
44.
        return Row(
45.
          mainAxisSize: MainAxisSize.min,
          children: [
46.
47.
            //Tombol Edit
48.
            OutlinedButton(
49.
                child: const Text("EDIT"),
50.
                onPressed: () {
                  Navigator.push(
51.
52.
                       context,
53.
                       MaterialPageRoute(
                           builder: (context) => ProdukForm(
54.
55.
                                 produk: widget.produk!,
56.
                               )));
57.
                }),
58.
            //Tombol Hapus
59.
            OutlinedButton(
60.
                child: const Text("DELETE"), onPressed: () => confirmHapus()),
61.
          ],
62.
        );
63.
      }
64.
65.
      void confirmHapus() {
66.
        AlertDialog alertDialog = AlertDialog(
67.
          content: const Text("Yakin ingin menghapus data ini?"),
68.
          actions: [
69.
            //tombol hapus
70.
            OutlinedButton(
71.
              child: const Text("Ya"),
72.
              onPressed: () {},
73.
            ),
74.
            //tombol batal
            OutlinedButton(
75.
              child: const Text("Batal"),
76.
77.
              onPressed: () => Navigator.pop(context),
78.
            )
79.
          ],
        );
80.
81.
        showDialog(builder: (context) => alertDialog, context: context);
82.
83.
      }
84.}
```

Tampil List Produk

Buat sebuah file dengan nama produk_page.dart pada folder ui dengan kode berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/model/produk.dart';
3. import 'package:tokokita/ui/produk_detail.dart';
4. import 'package:tokokita/ui/produk_form.dart';
5.
6. class ProdukPage extends StatefulWidget {
7.
      const ProdukPage({Key? key}) : super(key: key);
8.
9.
      @override
     _ProdukPageState createState() => _ProdukPageState();
10.
11. }
12.
13. class _ProdukPageState extends State<ProdukPage> {
14.
     @override
15.
      Widget build(BuildContext context) {
16.
        return Scaffold(
17.
            appBar: AppBar(
18.
              title: const Text('List Produk'),
19.
              actions: [
20.
                Padding(
                    padding: const EdgeInsets.only(right: 20.0),
21.
22.
                    child: GestureDetector(
23.
                      child: const Icon(Icons.add, size: 26.0),
24.
                      onTap: () async {
25.
                        Navigator.push(context,
26.
                            MaterialPageRoute(builder: (context) => ProdukForm()));
27.
                      },
28.
                    ))
29.
              ],
30.
            ),
31.
            drawer: Drawer(
              child: ListView(
32.
33.
                children: [
34.
                  ListTile(
                    title: const Text('Logout'),
35.
36.
                    trailing: const Icon(Icons.logout),
37.
                    onTap: () async {},
38.
                  )
39.
                ],
40.
              ),
41.
            ),
42.
            body: ListView(
43.
              children: [
44.
                ItemProduk(
45.
                    produk: Produk(
46.
                        id: 1,
47.
                        kodeProduk: 'A001',
                        namaProduk: 'Kamera'
48.
49.
                        hargaProduk: 500000)),
50.
                ItemProduk(
51.
                    produk: Produk(
52.
                        id: 2,
                        kodeProduk: 'A002',
53.
                        namaProduk: 'Kulkas'
54.
                        hargaProduk: 2500000)),
55.
                ItemProduk(
56.
57.
                    produk: Produk(
58.
                        id: 3,
                        kodeProduk: 'A003'
59.
                        namaProduk: 'Mesin Cuci',
60.
```
```
61.
                        hargaProduk: 200000)),
62.
             ],
            ));
63.
64.
     }
65.}
66.
67. class ItemProduk extends StatelessWidget {
68.
     final Produk produk;
69.
70.
      const ItemProduk({Key? key, required this.produk}) : super(key: key);
71.
     @override
72.
73.
     Widget build(BuildContext context) {
74.
        return GestureDetector(
75.
          onTap: () {
76.
            Navigator.push(
77.
                context,
78.
                MaterialPageRoute(
79.
                    builder: (context) => ProdukDetail(
80.
                          produk: produk,
81.
                        )));
82.
          },
83.
          child: Card(
84.
            child: ListTile(
              title: Text(produk.namaProduk!),
85.
86.
              subtitle: Text(produk.hargaProduk.toString()),
87.
            ),
88.
          ),
89.
        );
90.
     }
91.}
```

Untuk mencobanya modifikasi file main.dart menjadi seperti berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/ui/produk_page.dart';
3.
4. void main() {
5.
      runApp(const MyApp());
6. }
7.
8. class MyApp extends StatelessWidget {
9.
      const MyApp({Key? key}) : super(key: key);
10.
11.
      @override
12. Widget build(BuildContext context) {
13.
        return const MaterialApp(
14.
          title: 'Toko Kita',
15.
          debugShowCheckedModeBanner: false,
16.
          home: ProdukPage(),
17.
        );
18.
     }
19.}
```

Maka tampilannya akan menjadi seperti berikut



Pada saat tombol tambah diklik maka akan muncul form produk seperti berikut

8:44	LTE 🔟 📋
← Tambah Produk	
Kode Produk	
Nama Produk	
Harga	
Simpan	

Jika salah satu data produk diklik maka akan muncul detail produk

7:24	0			LTE 🔟 🔒
÷	D	etail Pro	oduk	
		Kode Nama Harga : R	: A001 : Kamera p. 5000000	
		EDIT	DELETE	

Ketika tombol EDIT diklik maka akan muncul form produk untuk mengubah data produk



Pada materi selanjutnya akan ada modifikasi pada tampil produk agar dapat menampilkan data dari Rest API serta modifikasi pada Form Produk sehingga dapat berfungsi untuk menyimpan ataupun mengubah data pada Rest API

Membuat Helper Modul

Menambahkan depedencies

Dalam pembuatan aplikasi ini dibutuhkan depedensi untuk menerima dan mengirim request ke Rest API (http) dan depedensi untuk menyimpan token (shared_preferences). Pastikan komputer atau laptop terhubung ke internet, buka file pubspec.yaml kemudian pada bagian dependencies tambahkan kode berikut



 ${\tt Untuk}\ {\tt mengunduh}\ {\tt depedencies}\ {\tt atau}\ {\tt package}\ {\tt yang}\ {\tt telah}\ {\tt ditambahkan},\ {\tt buka}\ {\tt CommandPrompt}$

kemudian masuk ke folder projek dan ketikkan **flutter pub get** kemudian tekan Enter



Membuat Class Token

Buat sebuah folder dengan nama helpers



Kemudian pada folder **helpers** buat sebuah file dengan nama **user_info.dart** dan masukkan kode berikut

```
1. import 'package:shared_preferences/shared_preferences.dart';
2.
3. class UserInfo {
4.
     Future setToken(String value) async {
       final SharedPreferences pref = await SharedPreferences.getInstance();
5.
       return pref.setString("token", value);
6.
7.
     }
8.
     Future<String?> getToken() async {
9.
    final SharedPreferences pref = await SharedPreferences.getInstance();
10.
```

```
11.
       return pref.getString("token");
12. }
13.
14.
     Future setUserID(int value) async {
15.
       final SharedPreferences pref = await SharedPreferences.getInstance();
       return pref.setInt("userID", value);
16.
17.
     }
18.
19.
     Future<int?> getUserID() async {
20. final SharedPreferences pref = await SharedPreferences.getInstance();
21.
       return pref.getInt("userID");
22.
     }
23.
24.
     Future logout() async {
25.
       final SharedPreferences pref = await SharedPreferences.getInstance();
26.
       pref.clear();
27.
     }
28.}
```

Http request

Membuat Modul Error Handling

Buat sebuah file pada folder helpers dengan nama app_exception.dart kemudian

ketikkan kode berikut

```
1. class AppException implements Exception {
2. final _message;
3.
     final prefix;
4.
5.
     AppException([this._message, this._prefix]);
6.
7.
     @override
8.
     String toString() {
9.
       return "$_prefix$_message";
10.
     }
11. }
12.
13. class FetchDataException extends AppException {
14. FetchDataException([String? message])
         : super(message, "Error During Communication: ");
15.
16.}
17.
18. class BadRequestException extends AppException {
     BadRequestException([message]) : super(message, "Invalid Request: ");
19.
20.}
21.
22. class UnauthorisedException extends AppException {
     UnauthorisedException([message]) : super(message, "Unauthorised: ");
23.
24. }
25.
26. class UnprocessableEntityException extends AppException {
     UnprocessableEntityException([message])
27.
        : super(message, "Unprocessable Entity: ");
28.
29.}
30.
31. class InvalidInputException extends AppException {
32. InvalidInputException([String? message]) : super(message, "Invalid Input: ");
33.}
```

Pada file ini berfungsi sebagai penanganan jika terjadi error saat melakukan permintaan atau pengiriman ke Rest API

Membuat Modul Http Request

Agar dapat mengirim atau menerima permintaan ke Rest API, dibuat sebuah fungsi baik

itu method POST, GET dan DELETE.

Buat sebuah file di dalam folder **helpers** dengan nama **api.dart** dan masukkan kode berikut

```
1. import 'dart:io';
2.
3. import 'package:http/http.dart' as http;
4. import 'package:tokokita/helpers/user_info.dart';
5. import 'app_exception.dart';
6.
7. class Api {
8.
     Future<dynamic> post(dynamic url, dynamic data) async {
9.
       var token = await UserInfo().getToken();
10.
       var responseJson;
       try {
11.
12.
         final response = await http.post(Uri.parse(url),
13.
              body: data,
14.
              headers: {HttpHeaders.authorizationHeader: "Bearer $token"});
15.
         responseJson = _returnResponse(response);
16.
        } on SocketException {
         throw FetchDataException('No Internet connection');
17.
18.
       }
19.
       return responseJson;
20.
     }
21.
22.
     Future<dynamic> get(dynamic url) async {
23.
       var token = await UserInfo().getToken();
24.
       var responseJson;
25.
       try {
26.
         final response = await http.get(url,
              headers: {HttpHeaders.authorizationHeader: "Bearer $token"});
27.
28.
         responseJson = _returnResponse(response);
29.
        } on SocketException {
30.
         throw FetchDataException('No Internet connection');
31.
       }
32.
       return responselson;
33.
     }
34.
35.
     Future<dynamic> delete(dynamic url) async {
       var token = await UserInfo().getToken();
36.
37.
       var responseJson;
38.
       try {
39.
         final response = await http.delete(url,
              headers: {HttpHeaders.authorizationHeader: "Bearer $token"});
40.
41.
         responseJson = _returnResponse(response);
42.
        } on SocketException {
43.
         throw FetchDataException('No Internet connection');
44.
       }
45.
       return responseJson;
46.
     }
47.
      dynamic returnResponse(http.Response response) {
48.
49.
        switch (response.statusCode) {
```

```
50. case 200:
51.
           return response:
52.
         case 400:
53.
           throw BadRequestException(response.body.toString());
54.
         case 401:
55.
         case 403:
           throw UnauthorisedException(response.body.toString());
56.
57.
         case 422:
58.
          throw InvalidInputException(response.body.toString());
59.
         case 500:
60.
         default:
61.
           throw FetchDataException(
               'Error occured while Communication with Server with StatusCode
62.
  ${response.statusCode}');
63.
       }
64.
     }
65.}
```

Membuat List API url

Buat sebuah file di dalam folder **helpers** dengan nama **api_url.dart**, dimana fungsi dari file ini adalah menyimpan alamat-alamat API yang telah dibuat sebelumnya (Endpoint dari API Spec)

```
1. class ApiUrl {
2.
     static const String baseUrl = 'http://10.0.2.2/toko-api/public';
3.
     static const String registrasi = baseUrl + '/registrasi';
4.
5.
     static const String login = baseUrl + '/login';
6.
     static const String listProduk = baseUrl + '/produk';
7.
     static const String createProduk = baseUrl + '/produk';
8.
9.
     static String updateProduk(int id) {
10.
      return baseUrl + '/produk/' + id.toString() + '/update';
11.
     }
12.
13.
     static String showProduk(int id) {
14.
     return baseUrl + '/produk/' + id.toString();
15.
     }
16.
17.
     static String deleteProduk(int id) {
18.
       return baseUrl + '/produk/' + id.toString();
19.
     }
20.}
```

Pada baris kedua (baseUrl) merupakan alamat IP dari Rest API, untuk memeriksa apakah terhubung dengan emulator atau tidak, dapat dilakukan dengan memasukkan alamat tersebut pada browser yang ada pada emulator atau handphone android yang terkoneksi dengan laptop



Membuat Bloc

Buat sebuah folder bernama **bloc**. Di dalam folder ini berisis file-file yang berfungsi sebagai controller baik itu untuk melakukan proses login, registrasi dan lain-lain.



Registrasi

Buat sebuah file dengan nama **registrasi_bloc.dart** pada folder **bloc**. Kemudian masukkan kode

berikut

```
    import 'dart:convert';
    import 'package:tokokita/helpers/api.dart';
    import 'package:tokokita/helpers/api_url.dart';
    import 'package:tokokita/model/registrasi.dart';
```

```
6.
7. class RegistrasiBloc {
     static Future<Registrasi> registrasi(
8.
9.
         {String? nama, String? email, String? password}) async {
10.
       String apiUrl = ApiUrl.registrasi;
11.
       var body = {"nama": nama, "email": email, "password": password};
12.
13.
14.
       var response = await Api().post(apiUrl, body);
15.
       var jsonObj = json.decode(response.body);
16.
       return Registrasi.fromJson(jsonObj);
17.
     }
18.}
```

Login

Buat sebuah file dengan nama login_bloc.dart pada folder bloc. Kemudian masukkan kode

berikut

```
1. import 'dart:convert';
2.
3. import 'package:tokokita/helpers/api.dart';
4. import 'package:tokokita/helpers/api_url.dart';
5. import 'package:tokokita/model/login.dart';
6.
7. class LoginBloc {
8. static Future<Login> login({String? email, String? password}) async {
9.
       String apiUrl = ApiUrl.login;
10.
       var body = {"email": email, "password": password};
       var response = await Api().post(apiUrl, body);
11.
12.
      var jsonObj = json.decode(response.body);
13.
       return Login.fromJson(jsonObj);
14. }
15.}
```

Logout

Buat sebuah file dengan nama logout_bloc.dart pada folder bloc. Kemudian masukkan kode

berikut

```
1. import 'package:tokokita/helpers/user_info.dart';
2.
3. class LogoutBloc{
4. static Future logout() async {
5. await UserInfo().logout();
6. }
7. }
```

Produk

Pada bagian ini akan dibuat beberapa fungsi untuk mengambil, mengubah dan menghapus data produk dari Rest API. Buat sebuah file dengan nama **produk_bloc.dart** pada folder **bloc** kemudian masukkan kode berikut

```
1. import 'dart:convert';
2.
3. import 'package:tokokita/helpers/api.dart';
4. import 'package:tokokita/helpers/api_url.dart';
5. import 'package:tokokita/model/produk.dart';
6.
7. class ProdukBloc {
8.
     static Future<List<Produk>> getProduks() async {
9.
       String apiUrl = ApiUrl.listProduk;
10.
       var response = await Api().get(apiUrl);
       var jsonObj = json.decode(response.body);
11.
       List<dynamic> listProduk = (jsonObj as Map<String, dynamic>)['data'];
12.
13.
       List<Produk> produks = [];
14.
       for (int i = 0; i < listProduk.length; i++) {</pre>
15.
          produks.add(Produk.fromJson(listProduk[i]));
16.
       }
17.
       return produks;
18.
     }
19.
20.
      static Future addProduk({Produk? produk}) async {
21.
       String apiUrl = ApiUrl.createProduk;
22.
       var body = {
23.
24.
          "kode_produk": produk!.kodeProduk,
          "nama_produk": produk.namaProduk,
25.
          "harga": produk.hargaProduk.toString()
26.
27.
       };
28.
29.
       var response = await Api().post(apiUrl, body);
30.
       var jsonObj = json.decode(response.body);
31.
       return jsonObj['status'];
32.
     }
33.
34.
      static Future<bool> updateProduk({required Produk produk}) async {
35.
       String apiUrl = ApiUrl.updateProduk(produk.id!);
36.
37.
       var body = {
          "kode_produk": produk.kodeProduk,
38.
          "nama_produk": produk.namaProduk,
39.
40.
          "harga": produk.hargaProduk.toString()
41.
       };
42.
       print("Body : $body");
43.
       var response = await Api().post(apiUrl, body);
44.
       var jsonObj = json.decode(response.body);
45.
       return jsonObj['data'];
46.
     }
47.
      static Future<bool> deleteProduk({int? id}) async {
48.
49.
       String apiUrl = ApiUrl.deleteProduk(id!);
50.
51.
       var response = await Api().delete(apiUrl);
       var jsonObj = json.decode(response.body);
52.
53.
       return (jsonObj as Map<String, dynamic>)['data'];
54.
      }
```

55.}

Menyatukan Fungsionalitas

Membuat Common Dialog Widget

Pada bagian ini akan dibuat dua buah dialog yang nantinya akan digunakan pada tampilan. Buat

sebuah folder dengan nama widget



Kemudian buat sebuah file dengan nama success_dialog.dart dengan kode

```
1. import 'package:flutter/material.dart';
2.
3. class Consts {
4.
   Consts._();
5.
     static const double padding = 16.0;
6.
7.
      static const double avatarRadius = 66.0;
8. }
9.
10. class SuccessDialog extends StatelessWidget {
11. final String? description;
12.
    final VoidCallback? okClick;
13.
14. const SuccessDialog({Key? key, this.description, this.okClick})
```

```
15.
          : super(key: key);
16.
17.
      @override
18.
     Widget build(BuildContext context) {
19.
        return Dialog(
          shape: RoundedRectangleBorder(
20.
              borderRadius: BorderRadius.circular(Consts.padding)),
21.
22.
          elevation: 0.0,
23.
          backgroundColor: Colors.transparent,
24.
          child: dialogContent(context),
25.
        );
26.
      }
27.
28.
      dialogContent(BuildContext context) {
29.
        return Container(
30.
          padding: const EdgeInsets.only(
31.
            top: Consts.padding,
32.
            bottom: Consts.padding,
33.
            left: Consts.padding,
34.
            right: Consts.padding,
35.
          ),
36.
          margin: const EdgeInsets.only(top: Consts.avatarRadius),
37.
          decoration: BoxDecoration(
38.
            color: Colors.white,
            shape: BoxShape.rectangle,
39.
40.
            borderRadius: BorderRadius.circular(Consts.padding),
41.
            boxShadow: const [
42.
              BoxShadow(
43.
                color: Colors.black26,
44.
                blurRadius: 10.0,
45.
                offset: Offset(0.0, 10.0),
46.
              ),
47.
            ],
48.
          ),
49.
          child: Column(
50.
            mainAxisSize: MainAxisSize.min,
51.
            children: [
52.
              const Text(
53.
                "SUKSES",
54.
                style: TextStyle(
55.
                    fontSize: 24.0,
56.
                    fontWeight: FontWeight.w700,
57.
                    color: Colors.green),
58.
              ),
59.
              const SizedBox(height: 16.0),
60.
              Text(
61.
                description!,
62.
                textAlign: TextAlign.center,
63.
                style: const TextStyle(
64.
                  fontSize: 16.0,
65.
                ),
              ),
66.
67.
              const SizedBox(height: 24.0),
68.
              Align(
69.
                alignment: Alignment.bottomRight,
70.
                child: OutlinedButton(
71.
                  onPressed: () {
72.
                    Navigator.of(context).pop(); // To close the dialog
73.
                    okClick!();
74.
                  },
75.
                  child: const Text("OK"),
76.
                ),
77.
              )
78.
            ],
79.
          ),
```

80.); 81. } 82.}

Kemudian buat file dengan nama warning_dialog.dart dengan kode

```
1. import 'package:flutter/material.dart';
2.
3. class Consts {
4.
     Consts._();
5.
6.
     static const double padding = 16.0;
7.
     static const double avatarRadius = 66.0;
8. }
9.
10. class WarningDialog extends StatelessWidget {
11.
     final String? description;
12.
     final VoidCallback? okClick;
13.
14.
     const WarningDialog({Key? key, this.description, this.okClick})
          : super(key: key);
15.
16.
17.
     @override
18.
     Widget build(BuildContext context) {
19.
        return Dialog(
20.
          shape: RoundedRectangleBorder(
21.
              borderRadius: BorderRadius.circular(Consts.padding)),
22.
          elevation: 0.0,
23.
          backgroundColor: Colors.transparent,
24.
          child: dialogContent(context),
25.
        );
26.
     }
27.
28.
     dialogContent(BuildContext context) {
29.
        return Container(
          padding: const EdgeInsets.only(
30.
31.
            top: Consts.padding,
32.
            bottom: Consts.padding,
33.
            left: Consts.padding,
34.
            right: Consts.padding,
35.
          ),
36.
          margin: const EdgeInsets.only(top: Consts.avatarRadius),
37.
          decoration: BoxDecoration(
38.
            color: Colors.white,
39.
            shape: BoxShape.rectangle,
40.
            borderRadius: BorderRadius.circular(Consts.padding),
41.
            boxShadow: const [
42.
              BoxShadow(
43.
                color: Colors.black26,
44.
                blurRadius: 10.0,
45.
                offset: Offset(0.0, 10.0),
46.
              ),
47.
            ],
          ),
48.
49.
          child: Column(
50.
            mainAxisSize: MainAxisSize.min,
            children: [
51.
52.
              const Text(
                "GAGAL",
53.
54.
                style: TextStyle(
55.
                    fontSize: 24.0, fontWeight: FontWeight.w700, color: Colors.red),
56.
              ),
57.
              const SizedBox(height: 16.0),
58.
              Text(
```

```
59.
                description!,
60.
                textAlign: TextAlign.center,
                style: const TextStyle(
61.
62.
                  fontSize: 16.0,
63.
                ),
64.
              ),
65.
              const SizedBox(height: 24.0),
66.
              Align(
67.
                alignment: Alignment.bottomRight,
68.
                child: ElevatedButton(
69.
                  onPressed: () {
70.
                    Navigator.of(context).pop(); // To close the dialog
71.
                    okClick!();
72.
                  },
73.
                  child: const Text("OK"),
74.
                ),
75.
              )
76.
            ],
          ),
77.
78.
        );
79.
      }
80.}
```

Modifikasi main.dart

Buka kembali file **main.dart** kita akan memodifikasi file tersebut dengan kondisi jika belum login maka akan membuka halaman login, namun jika sudah login maka akan membuka halaman list produk

1. import 'package:flutter/material.dart'; 2. import 'package:tokokita/helpers/user_info.dart'; 3. import 'package:tokokita/ui/login_page.dart'; 4. import 'package:tokokita/ui/produk_page.dart'; 5. 6. void main() { 7. runApp(const MyApp()); 8. } 9. 10. class MyApp extends StatefulWidget { 11. const MyApp({Key? key}) : super(key: key); 12. 13. @override _MyAppState createState() => _MyAppState(); 14. 15.} 16. 17. class _MyAppState extends State<MyApp> { Widget page = const CircularProgressIndicator(); 18. 19. 20. @override 21. void initState() { 22. super.initState(); 23. isLogin(); 24. } 25. void isLogin() async { 26. 27. var token = await UserInfo().getToken(); if (token != null) { 28. 29. setState(() { page = const ProdukPage(); 30. 31. }); 32. } **else** { 33. setState(() {

34.	<pre>page = const LoginPage();</pre>
35.	});
36.	}
37.	}
38.	
39.	@override
40.	<pre>Widget build(BuildContext context) {</pre>
41.	<pre>return MaterialApp(</pre>
42.	title: 'Toko Kita',
43.	<pre>debugShowCheckedModeBanner: false</pre>
44.	home: page,
45.);
46.	}
47.	}

Modifikasi registrasi_page.dart

Buka file **registrasi_page.dart** pada folder **ui** kemudian modifikasi fungsi _buttonRegistrasi dan

tambahkan fungsi dengan nama _submit seperti dibawah

119.	//Membuat Tombol Registrasi
120.	Widget _buttonRegistrasi() {
121.	return ElevatedButton(
122.	<pre>child: const Text("Registrasi"),</pre>
123.	onPressed: () {
124.	<pre>var validate = _formKey.currentState!.validate();</pre>
125.	<pre>if (validate) {</pre>
126.	<pre>if (!_isLoading) _submit();</pre>
127.	}
128.	});
129.	}
130.	
131.	<pre>void _submit() {</pre>
132.	_formKey.currentState!.save();
133.	setState(() {
134.	_isLoading = true;
135.	});
136.	RegistrasiBloc.registrasi(
137.	<pre>nama: _namaTextboxController.text,</pre>
138.	<pre>email: _emailTextboxController.text,</pre>
139.	<pre>password: _passwordTextboxController.text)</pre>
140.	.then((value) {
141.	showDialog(
142.	context: context,
143.	barrierDismissible: false,
144.	builder: (BuildContext context) => SuccessDialog(
145.	description: "Registrasi berhasil, silahkan login",
146.	okClick: () {
147.	Navigator.pop(context);
148.	},
149.));
150.	}, onError: (error) {
151.	showDialog(
152.	context: context,
153.	barrierDismissible: false,
154.	builder: (BuildContext context) =>
155.	description: "Registrasi gagal, silahkan coba lagi",
156.));
157.	});
158.	<pre>setState(() {</pre>
159.	_isLoading = false;
160.	});
161.	}

Sehingga keseluruhan kode pada registrasi_page.dart menjadi seperti berikut

```
1. import 'package:flutter/material.dart';

    import 'package:tokokita/bloc/registrasi_bloc.dart';
    import 'package:tokokita/widget/success_dialog.dart';

4. import 'package:tokokita/widget/warning_dialog.dart';
5.
6. class RegistrasiPage extends StatefulWidget {
7.
      const RegistrasiPage({Key? key}) : super(key: key);
8.
      @override
9.
      _RegistrasiPageState createState() => _RegistrasiPageState();
10.
11. }
12.
13. class _RegistrasiPageState extends State<RegistrasiPage> {
14.
      final _formKey = GlobalKey<FormState>();
15.
      bool _isLoading = false;
16.
      final _namaTextboxController = TextEditingController();
17.
             emailTextboxController = TextEditingController();
18.
      final
19.
      final passwordTextboxController = TextEditingController();
20.
      @override
21.
      Widget build(BuildContext context) {
22.
23.
        return Scaffold(
24.
          appBar: AppBar(
25.
            title: const Text("Registrasi"),
26.
          ),
27.
          body: SingleChildScrollView(
28.
            child: Padding(
29.
              padding: const EdgeInsets.all(8.0),
30.
               child: Form(
31.
                 key: _formKey,
32.
                 child: Column(
33.
                   mainAxisAlignment: MainAxisAlignment.center,
34.
                   children: [
                     _namaTextField(),
35.
36.
                     _emailTextField(),
                     _passwordTextField(),
37.
                     _passwordKonfirmasiTextField(),
38.
39.
                     _buttonRegistrasi()
40.
                   ],
41.
                 ),
              ),
42.
43.
            ),
44.
          ),
45.
        );
46.
      }
47.
48.
      //Membuat Textbox Nama
49.
      Widget _namaTextField() {
50.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Nama"),
51.
52.
          keyboardType: TextInputType.text,
53.
          controller: _namaTextboxController,
          validator: (value) {
54.
55.
            if (value!.length < 3) {</pre>
              return "Nama harus diisi minimal 3 karakter";
56.
57.
            }
58.
            return null;
59.
          },
60.
        );
```

```
}
61.
62.
      //Membuat Textbox email
63.
64.
     Widget _emailTextField() {
65.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Email"),
66.
          keyboardType: TextInputType.emailAddress,
67.
68.
          controller: _emailTextboxController,
69.
          validator: (value) {
70.
            //validasi harus diisi
            if (value!.isEmpty) {
71.
72.
              return 'Email harus diisi';
73.
            }
74.
            //validasi email
75.
            Pattern pattern =
76.
                r'^{(([^{<>})]^{,;:}s@^{]}+(.[^{<>})[^{]}),.;:s@^{]})@(([0)
    -9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-
   Z]{2,}))$';
77.
            RegExp regex = RegExp(pattern.toString());
78.
            if (!regex.hasMatch(value)) {
79.
              return "Email tidak valid";
80.
            }
81.
            return null;
82.
          },
83.
        );
84.
     }
85.
     //Membuat Textbox password
86.
87.
     Widget _passwordTextField() {
88.
        return TextFormField(
89.
          decoration: const InputDecoration(labelText: "Password"),
90.
          keyboardType: TextInputType.text,
91.
          obscureText: true,
92.
          controller: _passwordTextboxController,
          validator: (value) {
93.
94.
            //jika karakter yang dimasukkan kurang dari 6 karakter
95.
            if (value!.length < 6) {</pre>
96.
              return "Password harus diisi minimal 6 karakter";
97.
            }
98.
            return null;
99.
          },
100.
               );
101.
             }
102.
103.
             //membuat textbox Konfirmasi Password
104.
             Widget _passwordKonfirmasiTextField() {
105.
               return TextFormField(
106.
                 decoration: const InputDecoration(labelText: "Konfirmasi Password"),
107.
                 keyboardType: TextInputType.text,
108.
                 obscureText: true,
109.
                 validator: (value) {
                   //jika inputan tidak sama dengan password
110.
111.
                   if (value != _passwordTextboxController.text) {
112.
                     return "Konfirmasi Password tidak sama";
113.
                   }
114.
                   return null;
115.
                 },
116.
               );
117.
             }
118.
119.
             //Membuat Tombol Registrasi
             Widget _buttonRegistrasi() {
120.
               return ElevatedButton(
121.
122.
                   child: const Text("Registrasi"),
123.
                   onPressed: () {
```

124.	<pre>var validate = formKey.currentState!.validate();</pre>
125.	<pre>if (validate) {</pre>
126.	<pre>if (! isLoading) submit();</pre>
127.	}
128.	});
129.	}
130.	
131.	<pre>void _submit() {</pre>
132.	_formKey.currentState!.save();
133.	<pre>setState(() {</pre>
134.	_isLoading = true;
135.	});
136.	RegistrasiBloc.registrasi(
137.	<pre>nama: _namaTextboxController.text,</pre>
138.	<pre>email: _emailTextboxController.text,</pre>
139.	<pre>password: _passwordTextboxController.text)</pre>
140.	<pre>.then((value) {</pre>
141.	showDialog(
142.	context: context,
143.	barrierDismissible: <mark>false</mark> ,
144.	<pre>builder: (BuildContext context) => SuccessDialog(</pre>
145.	description: "Registrasi berhasil, silahkan login",
146.	okClick: () {
147.	Navigator.pop(context);
148.	},
149.));
150.	}, onError: (error) {
151.	showDialog(
152.	context: context,
153.	barrierDismissible: false,
154.	<pre>builder: (BuildContext context) => const WarningDialog(</pre>
155.	description: "Registrasi gagal, silahkan coba lagi",
156.));
157.	});
158.	<pre>setState(() {</pre>
159.	_isLoading = false;
160.	});
161.	}
162.	}



Modifikasi login_page.dart (fungsi login)

Buka file **login_page.dart** pada folder **ui** kemudian modifikasi fungsi _buttonLogin dan tambahkan fungsi dengan nama _submit seperti dibawah

```
83. //Membuat Tombol Login
84. Widget _buttonLogin() {
85.
       return ElevatedButton(
          child: const Text("Login"),
86.
87.
           onPressed: () {
             var validate = _formKey.currentState!.validate();
88.
89.
             if (validate) {
90.
              if (!_isLoading) _submit();
91.
             }
92.
           });
93. }
94.
95. void _submit() {
96.
      _formKey.currentState!.save();
97.
       setState(() {
98.
         _isLoading = true;
99.
       });
100.
              LoginBloc.login(
                      email: _emailTextboxController.text,
101.
102.
                      password: _passwordTextboxController.text)
103.
                  .then((value) async {
                await UserInfo().setToken(value.token.toString());
104.
                await UserInfo().setUserID(int.parse(value.userID.toString()));
105.
106.
                Navigator.pushReplacement(
                    context, MaterialPageRoute(builder: (context) => const
107.
   ProdukPage()));
              }, onError: (error) {
108.
                print(error);
109.
110.
                showDialog(
```

```
111.
                     context: context,
                     barrierDismissible: false,
112.
113.
                     builder: (BuildContext context) => const WarningDialog(
114.
                           description: "Login gagal, silahkan coba lagi",
115.
                         ));
116.
               });
117.
               setState(() {
118.
                 isLoading = false;
119.
               });
120.
```

Adapun kode secara keseluruhan menjadi seperti berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/login_bloc.dart';
3. import 'package:tokokita/helpers/user_info.dart';
4. import 'package:tokokita/ui/produk_page.dart';
5. import 'package:tokokita/ui/registrasi_page.dart';
6. import 'package:tokokita/widget/warning_dialog.dart';
7.
8. class LoginPage extends StatefulWidget {
9.
      const LoginPage({Key? key}) : super(key: key);
10.
11.
      @override
      _LoginPageState createState() => _LoginPageState();
12.
13. }
14.
15. class _LoginPageState extends State<LoginPage> {
      final _formKey = GlobalKey<FormState>();
16.
      bool isLoading = false;
17.
18.
19.
      final _emailTextboxController = TextEditingController();
20.
      final passwordTextboxController = TextEditingController();
21.
22.
      @override
23.
      Widget build(BuildContext context) {
24.
        return Scaffold(
25.
          appBar: AppBar(
26.
            title: const Text('Login'),
27.
          ),
28.
          body: SingleChildScrollView(
29.
            child: Padding(
30.
              padding: const EdgeInsets.all(8.0),
31.
               child: Form(
32.
                key: _formKey,
                child: Column(
33.
34.
                   children: [
35.
                     _emailTextField(),
36.
                     _passwordTextField(),
                     _buttonLogin(),
37.
38.
                     const SizedBox(
39.
                       height: 30,
40.
                     ),
41.
                     _menuRegistrasi()
42.
                  ],
43.
                ),
44.
              ),
45.
            ),
46.
          ),
47.
        );
      }
48.
49.
50.
      //Membuat Textbox email
51.
      Widget _emailTextField() {
```

```
52.
        return TextFormField(
53.
          decoration: const InputDecoration(labelText: "Email"),
          keyboardType: TextInputType.emailAddress,
54.
55.
          controller: _emailTextboxController,
56.
          validator: (value) {
            //validasi harus diisi
57.
58.
            if (value!.isEmpty) {
59.
              return 'Email harus diisi';
60.
            }
61.
            return null;
62.
          },
63.
        );
64.
      }
65.
66.
      //Membuat Textbox password
      Widget _passwordTextField() {
67.
68.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Password"),
69.
70.
          keyboardType: TextInputType.text,
71.
          obscureText: true,
72.
          controller: _passwordTextboxController,
73.
          validator: (value) {
74.
            //jika karakter yang dimasukkan kurang dari 6 karakter
            if (value!.isEmpty) {
75.
76.
              return "Password harus diisi";
77.
            }
78.
            return null;
79.
          },
80.
        );
81.
      }
82.
83.
      //Membuat Tombol Login
     Widget _buttonLogin() {
84.
85.
        return ElevatedButton(
86.
            child: const Text("Login"),
87.
            onPressed: () {
88.
              var validate = _formKey.currentState!.validate();
89.
              if (validate) {
90.
                if (!_isLoading) _submit();
91.
              }
92.
            });
93.
      }
94.
95.
      void _submit() {
96.
        _formKey.currentState!.save();
97.
        setState(() {
98.
          _isLoading = true;
99.
        });
100.
               LoginBloc.login(
                        email: _emailTextboxController.text,
101.
102.
                        password: _passwordTextboxController.text)
103.
                    .then((value) async {
104.
                 await UserInfo().setToken(value.token.toString());
105.
                 await UserInfo().setUserID(int.parse(value.userID.toString()));
106.
                 Navigator.pushReplacement(
107.
                      context,
                                  MaterialPageRoute(builder:
                                                                  (context)
                                                                                =>
                                                                                      const
   ProdukPage()));
108.
               }, onError: (error) {
109.
                 print(error);
                 showDialog(
110.
111.
                      context: context,
112.
                      barrierDismissible: false,
113.
                      builder: (BuildContext context) => const WarningDialog(
                            description: "Login gagal, silahkan coba lagi",
114.
115.
                          ));
```

```
116.
               });
117.
               setState(() {
118.
                 _isLoading = false;
119.
               });
120.
             }
121.
122.
             // Membuat menu untuk membuka halaman registrasi
123.
             Widget _menuRegistrasi() {
124.
               return Center(
                 child: InkWell(
125.
126.
                   child: const Text(
                      "Registrasi",
127.
                     style: TextStyle(color: Colors.blue),
128.
129.
                   ),
130.
                   onTap: () {
                     Navigator.push(context,
131.
132.
                         MaterialPageRoute(builder:
                                                            (context)
                                                                            =>
                                                                                      const
   RegistrasiPage()));
133.
                   },
134.
                 ),
135.
               );
136.
             }
137.
           }
```

Jika Gagal akan muncul pesan seperti berikut



Jika berhasil akan menuju ke halaman produk_page.dart

7:51	LTE 🔬 📋
≡ List Produk	÷
Kamera 5000000	
Kulkas 2500000	
Mesin Cuci 2000000	

Modifikasi produk_page.dart

Menambahkan fungsi logout pada drawer

Agar link logout dapat berfungsi, akan ditambahkan kode pada drawer logout, seperti

berikut

33.	drawer:	Drawer(
34.		child: ListView(
35.		children: [
36.		ListTile(
37.		<pre>title: const Text('Logout'),</pre>
38.		<pre>trailing: const Icon(Icons.logout),</pre>
39.		onTap: () async {
40.		<pre>await LogoutBloc.logout().then((value) => {</pre>
41.		Navigator.pushReplacement(
42.		context,
43.		MaterialPageRoute(
44.		<pre>builder: (context) => LoginPage()))</pre>
45.		});
46.		},
47.)
48.],
49.),
50.),
51.		body: ListView(

Secara keseluruhan kode pada produk_page.dart menjadi seperti berikut

1. import 'package:flutter/material.dart';

```
2. import 'package:tokokita/bloc/logout_bloc.dart';
3. import 'package:tokokita/model/produk.dart';
4. import 'package:tokokita/ui/login_page.dart';
5. import 'package:tokokita/ui/produk_detail.dart';
6. import 'package:tokokita/ui/produk form.dart';
7.
8. class ProdukPage extends StatefulWidget {
9.
      const ProdukPage({Key? key}) : super(key: key);
10.
11.
      @override
      ProdukPageState createState() => ProdukPageState();
12.
13. }
14.
15. class ProdukPageState extends State<ProdukPage> {
16.
      @override
17.
      Widget build(BuildContext context) {
18.
        return Scaffold(
19.
            appBar: AppBar(
20.
              title: const Text('List Produk'),
21.
              actions: [
22.
                Padding(
23.
                    padding: const EdgeInsets.only(right: 20.0),
24.
                    child: GestureDetector(
25.
                      child: const Icon(Icons.add, size: 26.0),
26.
                      onTap: () async {
27.
                        Navigator.push(context,
                            MaterialPageRoute(builder: (context) => ProdukForm()));
28.
29.
                      },
30.
                    ))
31.
              ],
32.
            ),
33.
            drawer: Drawer(
              child: ListView(
34.
35.
                children: [
36.
                  ListTile(
37.
                    title: const Text('Logout'),
38.
                    trailing: const Icon(Icons.logout),
39.
                    onTap: () async {
40.
                      await LogoutBloc.logout().then((value) => {
                             Navigator.pushReplacement(
41.
42.
                                 context,
43.
                                 MaterialPageRoute(
44.
                                     builder: (context) => LoginPage()))
45.
                          });
46.
                    },
47.
                  )
48.
                ],
49.
              ),
50.
            ),
51.
            body: ListView(
              children: [
52.
53.
                ItemProduk(
54.
                    produk: Produk(
55.
                        id: 1,
56.
                        kodeProduk: 'A001',
                        namaProduk: 'Kamera'
57.
58.
                        hargaProduk: 500000)),
59.
                ItemProduk(
                    produk: Produk(
60.
                        id: 2,
61.
                        kodeProduk: 'A002',
62.
63.
                        namaProduk: 'Kulkas'
64.
                        hargaProduk: 2500000)),
                ItemProduk(
65.
66.
                    produk: Produk(
```

```
67.
                         id: 3,
68.
                         kodeProduk: 'A003',
                         namaProduk: 'Mesin Cuci',
69.
                         hargaProduk: 2000000)),
70.
71.
              ],
            ));
72.
73.
     }
74.}
75.
76. class ItemProduk extends StatelessWidget {
      final Produk produk;
77.
78.
79.
      const ItemProduk({Key? key, required this.produk}) : super(key: key);
80.
81.
      @override
82.
     Widget build(BuildContext context) {
83.
        return GestureDetector(
84.
          onTap: () {
85.
            Navigator.push(
86.
                context,
87.
                MaterialPageRoute(
88.
                    builder: (context) => ProdukDetail(
89.
                           produk: produk,
90.
                         )));
91.
          },
92.
          child: Card(
93.
            child: ListTile(
94.
              title: Text(produk.namaProduk!),
95.
              subtitle: Text(produk.hargaProduk.toString()),
96.
           ),
97.
          ),
98.
        );
99.
      }
100.
           }
```

Menampilkan Data Produk dari Rest API

Pada bagian ini akan dimodifikasi file produk_page.dart sehingga dapat menampilkan data

dari Rest API. Berikut kode keseluruhan

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/logout_bloc.dart';
3. import 'package:tokokita/bloc/produk_bloc.dart';
4. import 'package:tokokita/model/produk.dart';
5. import 'package:tokokita/ui/login_page.dart';
6. import 'package:tokokita/ui/produk_detail.dart';
7. import 'package:tokokita/ui/produk_form.dart';
8.
9. class ProdukPage extends StatefulWidget {
10. const ProdukPage({Key? key}) : super(key: key);
11.
12.
      @override
      _ProdukPageState createState() => _ProdukPageState();
13.
14. }
15.
16. class _ProdukPageState extends State<ProdukPage> {
17.
      @override
18.
      Widget build(BuildContext context) {
         return Scaffold(
19.
20.
        appBar: AppBar(
```

```
21.
            title: const Text('List Produk'),
22.
            actions: [
23.
              Padding(
                  padding: const EdgeInsets.only(right: 20.0),
24.
25.
                  child: GestureDetector(
26.
                    child: const Icon(Icons.add, size: 26.0),
27.
                    onTap: () async {
28.
                      Navigator.push(context,
29.
                           MaterialPageRoute(builder: (context) => ProdukForm()));
30.
                    },
                  ))
31.
32.
            ],
33.
          ),
34.
          drawer: Drawer(
35.
            child: ListView(
36.
              children: [
37.
                ListTile(
                  title: const Text('Logout'),
38.
39.
                  trailing: const Icon(Icons.logout),
40.
                  onTap: () async {
41.
                    await LogoutBloc.logout().then((value) => {
42.
                           Navigator.pushReplacement(context,
43.
                               MaterialPageRoute(builder: (context) => LoginPage()))
44.
                        });
45.
                  },
46.
               )
              ],
47.
            ),
48.
49.
          ),
50.
          body: FutureBuilder<List>(
51.
            future: ProdukBloc.getProduks(),
52.
            builder: (context, snapshot) {
53.
              if (snapshot.hasError) print(snapshot.error);
54.
              return snapshot.hasData
55.
                  ? ListProduk(
56.
                      list: snapshot.data,
57.
                    )
58.
                  : const Center(
59.
                      child: CircularProgressIndicator(),
60.
                    );
61.
            },
62.
          ),
63.
        );
64.
     }
65.}
66.
67. class ListProduk extends StatelessWidget {
68.
     final List? list;
69.
70.
      const ListProduk({Key? key, this.list}) : super(key: key);
71.
      @override
72.
73.
      Widget build(BuildContext context) {
74.
        return ListView.builder(
75.
            itemCount: list == null ? 0 : list!.length,
            itemBuilder: (context, i) {
76.
77.
              return ItemProduk(
78.
                produk: list![i],
79.
              );
80.
            });
81.
     }
82.}
83.
84. class ItemProduk extends StatelessWidget {
85.
      final Produk produk;
```

```
86.
87.
      const ItemProduk({Key? key, required this.produk}) : super(key: key);
88.
89.
      @override
90.
     Widget build(BuildContext context) {
91.
        return GestureDetector(
92.
          onTap: () {
93.
            Navigator.push(
94.
                context,
95.
                MaterialPageRoute(
96.
                    builder: (context) => ProdukDetail(
97.
                           produk: produk,
98.
                         )));
99.
          },
100.
                  child: Card(
101.
                   child: ListTile(
                      title: Text(produk.namaProduk!),
102.
103.
                      subtitle: Text(produk.hargaProduk.toString()),
104.
                   ),
105.
                 ),
106.
               );
107.
             }
           }
108.
```

Adapun perubahan yang dilakukan adalah penambahan sebuah class bernama ListProduk

dengan kode

```
67. class ListProduk extends StatelessWidget {
68. final List? list;
69.
70.
     const ListProduk({Key? key, this.list}) : super(key: key);
71.
72.
     @override
     Widget build(BuildContext context) {
73.
74.
       return ListView.builder(
75.
            itemCount: list == null ? 0 : list!.length,
            itemBuilder: (context, i) {
76.
77.
              return ItemProduk(
78.
                produk: list![i],
79.
              );
80.
            });
81.
     }
82.}
```

Kemudian perubahan pada bagian body menjadi

```
50. body: FutureBuilder<List>(
51.
            future: ProdukBloc.getProduks(),
52.
            builder: (context, snapshot) {
53.
              if (snapshot.hasError) print(snapshot.error);
54.
              return snapshot.hasData
55.
                  ? ListProduk(
56.
                      list: snapshot.data,
57.
                    )
58.
                  : const Center(
59.
                      child: CircularProgressIndicator(),
60.
                    );
61.
            },
62.
          ),
```

Serta memasukkan beberapa package yang diperlukan

- 1. import 'package:flutter/material.dart'; import 'package:tokokita/bloc/logout_bloc.dart';
 import 'package:tokokita/bloc/produk_bloc.dart';
 import 'package:tokokita/model/produk.dart';

- 5. import 'package:tokokita/ui/login_page.dart';
- import 'package:tokokita/ui/produk_detail.dart';
- 7. import 'package:tokokita/ui/produk_form.dart';



Memodifikasi Form Produk (produk form.dart)

Membuat fungsi simpan

Agar tombol simpan dapat berfungsi diperlukan kode fungsi untuk menyimpan data dengan memanggil bloc produk_bloc yang telah dibuat sebelumnya, kita akan menambahkan sebuah fungsi dengan nama simpan dan memodifikasi fungsi _buttonSubmit

```
116.
              //Membuat Tombol Simpan/Ubah
117.
              Widget _buttonSubmit() {
                return OutlinedButton(
118.
                    child: Text(tombolSubmit),
119.
                    onPressed: () {
120.
                       var validate = _formKey.currentState!.validate();
if (validate) {
121.
122.
                         if (!_isLoading) {
123.
                           if (widget.produk != null) {
124.
```

```
125.
                            //kondisi update produk
126.
127.
                          } else {
128.
                            //kondisi tambah produk
129.
                            simpan();
130.
                          }
131.
                       }
132.
                      }
                   });
133.
134.
             }
135.
136.
             simpan() {
137.
               setState(() {
138.
                  _isLoading = true;
139.
               });
140.
               Produk createProduk = Produk(id: null);
               createProduk.kodeProduk = _kodeProdukTextboxController.text;
141.
               createProduk.namaProduk = _namaProdukTextboxController.text;
142.
               createProduk.hargaProduk =
143.
   int.parse(_hargaProdukTextboxController.text);
144.
               ProdukBloc.addProduk(produk: createProduk).then((value) {
145.
                 Navigator.of(context).push(MaterialPageRoute(
146.
                      builder: (BuildContext context) => const ProdukPage()));
               }, onError: (error) {
147.
148.
                 showDialog(
149.
                     context: context,
150.
                     builder: (BuildContext context) => const WarningDialog(
151.
                            description: "Simpan gagal, silahkan coba lagi",
152.
                          ));
153.
               });
154.
               setState(() {
                 _isLoading = false;
155.
156.
               });
157.
             }
```

Dengan keseluruhan kode menjadi

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/produk_bloc.dart';
3. import 'package:tokokita/model/produk.dart';
4. import 'package:tokokita/ui/produk_page.dart';
5. import 'package:tokokita/widget/warning_dialog.dart';
6.
7. class ProdukForm extends StatefulWidget {
8.
     Produk? produk;
9.
10.
     ProdukForm({Key? key, this.produk}) : super(key: key);
11.
12.
     @override
     _ProdukFormState createState() => _ProdukFormState();
13.
14. }
15.
16. class _ProdukFormState extends State<ProdukForm> {
17.
     final formKey = GlobalKey<FormState>();
18.
     bool isLoading = false;
     String judul = "TAMBAH PRODUK";
19.
20.
     String tombolSubmit = "SIMPAN";
21.
     final _kodeProdukTextboxController = TextEditingController();
22.
23.
     final
            _namaProdukTextboxController = TextEditingController();
24.
     final _hargaProdukTextboxController = TextEditingController();
25.
26.
     @override
27.
     void initState() {
```

```
28.
        super.initState();
        isUpdate();
29.
30.
     }
31.
32.
      isUpdate() {
33.
        if (widget.produk != null) {
          setState(() {
34.
            judul = "UBAH PRODUK";
35.
            tombolSubmit = "UBAH";
36.
37.
            _kodeProdukTextboxController.text = widget.produk!.kodeProduk!;
38.
            namaProdukTextboxController.text = widget.produk!.namaProduk!;
            _hargaProdukTextboxController.text =
39.
40.
                widget.produk!.hargaProduk.toString();
41.
          });
          else {
42.
        }
          judul = "TAMBAH PRODUK";
43.
          tombolSubmit = "SIMPAN";
44.
45.
        }
46.
     }
47.
48.
      @override
49.
      Widget build(BuildContext context) {
        return Scaffold(
50.
          appBar: AppBar(title: Text(judul)),
51.
52.
          body: SingleChildScrollView(
53.
            child: Padding(
54.
              padding: const EdgeInsets.all(8.0),
55.
              child: Form(
56.
                key: _formKey,
                child: Column(
57.
                  children: [
58.
                    _kodeProdukTextField(),
59.
60.
                     _namaProdukTextField(),
                     _hargaProdukTextField(),
61.
                    _buttonSubmit()
62.
63.
                  ],
64.
                ),
65.
              ),
66.
            ),
67.
          ),
68.
       );
69.
      }
70.
71.
      //Membuat Textbox Kode Produk
72.
     Widget _kodeProdukTextField() {
73.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Kode Produk"),
74.
75.
          keyboardType: TextInputType.text,
76.
          controller: _kodeProdukTextboxController,
          validator: (value) {
77.
            if (value!.isEmpty) {
78.
79.
              return "Kode Produk harus diisi";
80.
            }
81.
            return null;
82.
          },
83.
        );
      }
84.
85.
86.
      //Membuat Textbox Nama Produk
      Widget _namaProdukTextField() {
87.
88.
        return TextFormField(
89.
          decoration: const InputDecoration(labelText: "Nama Produk"),
90.
          keyboardType: TextInputType.text,
          controller: _namaProdukTextboxController,
91.
92.
          validator: (value) {
```

```
93
            if (value!.isEmpty) {
94.
              return "Nama Produk harus diisi";
95.
            }
96.
            return null;
97.
          },
98.
        );
99.
      }
100.
101.
              //Membuat Textbox Harga Produk
              Widget _hargaProdukTextField() {
102.
103.
                return TextFormField(
104.
                  decoration: const InputDecoration(labelText: "Harga"),
105.
                  keyboardType: TextInputType.number,
106.
                  controller: _hargaProdukTextboxController,
107.
                  validator: (value) {
108.
                    if (value!.isEmpty) {
                      return "Harga harus diisi";
109.
110.
                    }
111.
                    return null;
112.
                  },
113.
                );
114.
              }
115.
116.
              //Membuat Tombol Simpan/Ubah
117.
              Widget _buttonSubmit() {
118.
                return OutlinedButton(
119.
                    child: Text(tombolSubmit),
120.
                    onPressed: () {
                      var validate = _formKey.currentState!.validate();
121.
                      if (validate) {
122.
123.
                        if (!_isLoading) {
124.
                           if (widget.produk != null) {
125.
                             //kondisi update produk
126.
127.
                           } else {
128.
                             //kondisi tambah produk
129.
                             simpan();
130.
                           }
131.
                        }
                      }
132.
133.
                    });
134.
135.
136.
              simpan() {
137.
                setState(() {
138.
                  _isLoading = true;
139.
                });
140.
                Produk createProduk = Produk(id: null);
                createProduk.kodeProduk = _kodeProdukTextboxController.text;
createProduk.namaProduk = _namaProdukTextboxController.text;
141.
142.
                createProduk.hargaProduk = int.parse(_hargaProdukTextboxController.text);
143.
                ProdukBloc.addProduk(produk: createProduk).then((value) {
144.
145.
                  Navigator.of(context).push(MaterialPageRoute(
146.
                      builder: (BuildContext context) => const ProdukPage()));
147.
                }, onError: (error) {
                  showDialog(
148.
149.
                      context: context,
150.
                      builder: (BuildContext context) => const WarningDialog(
                             description: "Simpan gagal, silahkan coba lagi",
151.
152.
                           ));
153.
                });
154.
                setState(() {
155.
                   _isLoading = false;
156.
                });
157.
              }
```

158. }

Membuat fungsi ubah

Sama halnya dengan simpan, kita buat sebuah fungsi ubah kemudian kita sertakan pada

fungsi _buttonSubmit

Dengan fungsi ubah sebagai berikut

159. u	pah() {
160.	<pre>setState(() {</pre>
161.	_isLoading = true;
162.	});
163.	Produk updateProduk = Produk(id: null);
164.	updateProduk.id = widget.produk!.id;
165.	updateProduk.kodeProduk = _kodeProdukTextboxController.text;
166.	updateProduk.namaProduk = _namaProdukTextboxController.text;
167.	updateProduk.hargaProduk =
<pre>int.parse</pre>	e(_hargaProdukTextboxController.text);
168.	<pre>ProdukBloc.updateProduk(produk: updateProduk).then((value) {</pre>
169.	Navigator.of(context).push(MaterialPageRoute(
170.	<pre>builder: (BuildContext context) => const ProdukPage()));</pre>
171.	}, onError: (error) {
172.	showDialog(
173.	context: context,
174.	<pre>builder: (BuildContext context) => const WarningDialog(</pre>
175.	description: "Permintaan ubah data gagal, silahkan coba
lagi",	
176.));
177.	});
178.	<pre>setState(() {</pre>
179.	_isLoading = false;
180.	});
181.	}

Kemudian kita tambahkan pada fungsi _buttonSubmit

```
116.
           //Membuat Tombol Simpan/Ubah
117.
             Widget _buttonSubmit() {
118.
               return OutlinedButton(
                 child: Text(tombolSubmit),
119.
                   onPressed: () {
120.
121.
                     var validate = _formKey.currentState!.validate();
122.
                     if (validate) {
123.
                       if (!_isLoading) {
124.
                         if (widget.produk != null) {
125.
                           //kondisi update produk
126.
                           ubah();
127.
                         } else {
128.
                           //kondisi tambah produk
129.
                           simpan();
130.
                         }
                      }
131.
132.
                     }
133.
                   });
             }
134.
```

Dengan kode keseluruhan menjadi seperti berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/produk_bloc.dart';
3. import 'package:tokokita/model/produk.dart';

    import 'package:tokokita/ui/produk_page.dart';

5. import 'package:tokokita/widget/warning_dialog.dart';
6.
7. class ProdukForm extends StatefulWidget {
8.
      Produk? produk;
9.
10.
      ProdukForm({Key? key, this.produk}) : super(key: key);
11.
12.
      @override
      _ProdukFormState createState() => _ProdukFormState();
13.
14. }
15.
16. class _ProdukFormState extends State<ProdukForm> {
      final _formKey = GlobalKey<FormState>();
bool _isLoading = false;
String judul = "TAMBAH PRODUK";
17.
18.
19.
20.
      String tombolSubmit = "SIMPAN";
21.
22.
      final _kodeProdukTextboxController = TextEditingController();
23.
      final namaProdukTextboxController = TextEditingController();
24.
      final _hargaProdukTextboxController = TextEditingController();
25.
26.
      @override
27.
      void initState() {
28.
      super.initState();
29.
        isUpdate();
30. }
31.
32.
      isUpdate() {
        if (widget.produk != null) {
33.
          setState(() {
34.
35.
            judul = "UBAH PRODUK";
36.
            tombolSubmit = "UBAH";
37.
            _kodeProdukTextboxController.text = widget.produk!.kodeProduk!;
38.
            __namaProdukTextboxController.text = widget.produk!.namaProduk!;
            _hargaProdukTextboxController.text =
39.
40.
                widget.produk!.hargaProduk.toString();
41.
          });
        } else {
42.
          judul = "TAMBAH PRODUK";
43.
          tombolSubmit = "SIMPAN";
44.
45.
        }
      }
46.
47.
48.
      @override
49.
      Widget build(BuildContext context) {
50.
        return Scaffold(
          appBar: AppBar(title: Text(judul)),
51.
52.
          body: SingleChildScrollView(
53.
            child: Padding(
54.
              padding: const EdgeInsets.all(8.0),
55.
               child: Form(
56.
                 key: formKey,
57.
                 child: Column(
58.
                   children: [
                     _kodeProdukTextField(),
59.
60.
                     namaProdukTextField(),
                     _hargaProdukTextField(),
61.
62.
                     buttonSubmit()
```

```
],
),
63.
64.
65.
              ),
66.
           ),
67.
          ),
68.
        );
69.
     }
70.
      //Membuat Textbox Kode Produk
71.
     Widget kodeProdukTextField() {
72.
73.
        return TextFormField(
          decoration: const InputDecoration(labelText: "Kode Produk"),
74.
75.
          keyboardType: TextInputType.text,
76.
          controller: _kodeProdukTextboxController,
77.
          validator: (value) {
78.
            if (value!.isEmpty) {
79.
              return "Kode Produk harus diisi";
80.
            }
81.
            return null;
82.
          },
83.
        );
84.
     }
85.
86.
      //Membuat Textbox Nama Produk
87.
     Widget _namaProdukTextField() {
88.
        return TextFormField(
89.
          decoration: const InputDecoration(labelText: "Nama Produk"),
90.
          keyboardType: TextInputType.text,
          controller: _namaProdukTextboxController,
91.
          validator: (value) {
92.
93.
            if (value!.isEmpty) {
94.
              return "Nama Produk harus diisi";
95.
            }
96.
           return null;
97.
          },
98.
        );
99.
     }
100.
101.
             //Membuat Textbox Harga Produk
             Widget hargaProdukTextField() {
102.
103.
               return TextFormField(
104.
                 decoration: const InputDecoration(labelText: "Harga"),
105.
                 keyboardType: TextInputType.number,
106.
                 controller: _hargaProdukTextboxController,
107.
                 validator: (value) {
108.
                   if (value!.isEmpty) {
109.
                     return "Harga harus diisi";
110.
                   }
111.
                   return null;
112.
                 },
113.
               );
114.
             }
115.
116.
             //Membuat Tombol Simpan/Ubah
117.
             Widget _buttonSubmit() {
               return OutlinedButton(
118.
119.
                   child: Text(tombolSubmit),
120.
                   onPressed: () {
121.
                     var validate = _formKey.currentState!.validate();
122.
                     if (validate) {
123.
                       if (! isLoading) {
124.
                          if (widget.produk != null) {
125.
                            //kondisi update produk
126.
                            ubah();
127.
                          } else {
```
```
128.
                            //kondisi tambah produk
129.
                            simpan();
130.
                          }
131.
                       }
                     }
132.
133.
                   });
134.
             }
135.
             simpan() {
136.
137.
               setState(() {
                  isLoading = true;
138.
139.
               });
               Produk createProduk = Produk(id: null);
140.
141.
               createProduk.kodeProduk = _kodeProdukTextboxController.text;
142.
               createProduk.namaProduk = _namaProdukTextboxController.text;
143.
               createProduk.hargaProduk = int.parse(_hargaProdukTextboxController.text);
144.
               ProdukBloc.addProduk(produk: createProduk).then((value) {
145.
                 Navigator.of(context).push(MaterialPageRoute(
146.
                     builder: (BuildContext context) => const ProdukPage()));
147.
               }, onError: (error) {
148.
                 showDialog(
149.
                      context: context,
150.
                     builder: (BuildContext context) => const WarningDialog(
151.
                            description: "Simpan gagal, silahkan coba lagi",
152.
                          ));
153.
               });
               setState(() {
154.
155.
                  _isLoading = false;
156.
               });
157.
             }
158.
159.
             ubah() {
160.
               setState(() {
161.
                  _isLoading = true;
162.
               });
               Produk updateProduk = Produk(id: null);
163.
164.
               updateProduk.id = widget.produk!.id;
               updateProduk.kodeProduk = _kodeProdukTextboxController.text;
165.
               updateProduk.namaProduk = _namaProdukTextboxController.text;
166.
167.
               updateProduk.hargaProduk = int.parse( hargaProdukTextboxController.text);
168.
               ProdukBloc.updateProduk(produk: updateProduk).then((value) {
169.
                 Navigator.of(context).push(MaterialPageRoute(
170.
                     builder: (BuildContext context) => const ProdukPage()));
171.
               }, onError: (error) {
172.
                 showDialog(
173.
                      context: context,
174.
                     builder: (BuildContext context) => const WarningDialog(
175.
                            description: "Permintaan ubah data gagal, silahkan coba lagi",
176.
                          ));
177.
               });
               setState(() {
178.
179.
                  _isLoading = false;
180.
               });
181.
             }
182.
           }
```

Menambahkan fungsi hapus pada Detail Produk (produk_detail.dart)

Buka file produk_detail.dart pada folder ui, kemudian kita modifikasi pada fungsi confirmHapus

menjadi seperti berikut

65. void confirmHapus() {
66. AlertDialog alertDialog = AlertDialog(

```
content: const Text("Yakin ingin menghapus data ini?"),
67.
68.
          actions: [
69.
            //tombol hapus
70.
            OutlinedButton(
              child: const Text("Ya"),
71.
72.
              onPressed: () {
73.
                Navigator.push(
74.
                     context,
75.
                     MaterialPageRoute(
76.
                         builder: (context) => ProdukForm(
77.
                               produk: widget.produk!,
78.
                             )));
79.
              },
80.
            ),
81.
            //tombol batal
82.
            OutlinedButton(
              child: const Text("Batal"),
83.
              onPressed: () => Navigator.pop(context),
84.
85.
            )
86.
          ],
87.
        );
88.
89.
        showDialog(builder: (context) => alertDialog, context: context);
90.
      }
```

Dengan kode keseluruhan menjadi

```
    import 'package:flutter/material.dart';

import 'package:tokokita/model/produk.dart';
3. import 'package:tokokita/ui/produk form.dart';
4.
5. class ProdukDetail extends StatefulWidget {
     Produk? produk;
6.
7.
8.
     ProdukDetail({Key? key, this.produk}) : super(key: key);
9.
10.
     @override
11.
      _ProdukDetailState createState() => _ProdukDetailState();
12. }
13.
14. class _ProdukDetailState extends State<ProdukDetail> {
15.
     @override
16.
     Widget build(BuildContext context) {
17.
       return Scaffold(
18.
          appBar: AppBar(
19.
            title: const Text('Detail Produk'),
20.
          ),
21.
          body: Center(
            child: Column(
22.
23.
              children: [
24.
                Text(
25.
                  "Kode : ${widget.produk!.kodeProduk}",
                  style: const TextStyle(fontSize: 20.0),
26.
27.
                ),
28.
                Text(
29.
                  "Nama : ${widget.produk!.namaProduk}",
30.
                  style: const TextStyle(fontSize: 18.0),
31.
                ),
32.
                Text(
                  "Harga : Rp. ${widget.produk!.hargaProduk.toString()}",
33.
34.
                  style: const TextStyle(fontSize: 18.0),
35.
                ),
                _tombolHapusEdit()
36.
              ],
37.
```

```
38.
          ),
39.
          ),
40.
       );
      }
41.
42.
43.
     Widget _tombolHapusEdit() {
44.
        return Row(
45.
          mainAxisSize: MainAxisSize.min,
46.
          children: [
47.
            //Tombol Edit
48.
            OutlinedButton(
49.
                child: const Text("EDIT"),
50.
                onPressed: () {
51.
                  Navigator.push(
52.
                      context,
53.
                      MaterialPageRoute(
54.
                           builder: (context) => ProdukForm(
55.
                                 produk: widget.produk!,
56.
                               )));
57.
                }),
58.
            //Tombol Hapus
59.
            OutlinedButton(
                child: const Text("DELETE"), onPressed: () => confirmHapus()),
60.
61.
          ],
62.
        );
63.
      }
64.
65.
      void confirmHapus() {
        AlertDialog alertDialog = AlertDialog(
66.
67.
          content: const Text("Yakin ingin menghapus data ini?"),
          actions: [
68.
69.
            //tombol hapus
70.
            OutlinedButton(
              child: const Text("Ya"),
71.
72.
              onPressed: () {
73.
                Navigator.push(
74.
                    context,
75.
                    MaterialPageRoute(
76.
                        builder: (context) => ProdukForm(
77.
                               produk: widget.produk!,
78.
                             )));
79.
              },
80.
            ),
81.
            //tombol batal
82.
            OutlinedButton(
83.
              child: const Text("Batal"),
84.
              onPressed: () => Navigator.pop(context),
85.
            )
86.
          ],
        );
87.
88.
89.
        showDialog(builder: (context) => alertDialog, context: context);
90.
      }
91.}
```