BRIEFING
PENGENALAN PRAKTIKUM MCS

Pada bab *briefing*, praktikan akan diperkenalkan mengenai hal-hal mendasar yang berkaitan dengan proses praktikum *Mobile Computing System* (MCS), seperti materi, bahasa pemrograman, *framework*, dan *tools* yang digunakan selama praktikum berlangsung. Selain itu, pada pertemuan ini, praktikan juga akan diajarkan bagaimana caranya meng*install* berbagai keperluan yang dibutuhkan selama praktikum.

B.1 Tujuan Praktikum

Tujuan	Penjelasan
Memahami sistem operasi android dan sejarah perkembangannya	Pada bab *briefing*, praktikan akan diberikan penjelasan singkat mengenai sistem operasi android, seperti apa itu android, asal mula android, dan perkembangan dari sistem operasi android
Memahami dasar bahasa pemrograman dart dan *framework* Flutter dalam mengembangkan aplikasi berbasis *mobile*	Pada bab *briefing*, praktikan akan diberikan penjelasan singkat mengenai bahasa pemrograman dart mulai dari apa itu dart, asal mula, kelebihan, kekurangan dan beberapa *syntax* yang ada pada dart. Selain itu, terdapat juga penjelasan mengenai *framework* Flutter.
Mengenalkan seluruh tools yang akan digunakan selama praktikum	Pada bab *briefing*, praktikan akan dikenalkan terhadap beberapa tools yang akan digunakan, seperti android studio dan *visual studio code.*
Memahami proses instalasi dart SDK, android *studio*, dan *visual studio code*	Pada bab *briefing* ini terdapat langkah demi langkah dalam melakukan instalasi terhadap dart SDK dan android studio, mulai dari proses *download, setup* pada *hardware* hingga

	konfigurasi pada *software* android *studio* dan *visual studio code*
Memahami cara pembuatan Flutter *project* pada android *studio*	Pada bab *briefing*, praktikan akan diajarkan bagaimana caranya membuat *project* Flutter pada android studio
Memahami beberapa widget dasar yang ada pada flutter	Pada bab *briefing*, praktikan akan diperkenalkan dengan beberapa widget dasar yang ada pada Flutter

B.2 Persyaratan Praktikum

Disarankan praktikan menggunakan *hardware* dan *software* sesuai pada dokumentasi ini. Apabila terdapat versi yang lumayan lampau dari versi yang direkomendasikan atau *hardware* yang lawas maka sebaiknya bertanya kepada Asisten mengajar *shift*.

***HARDWARE* YANG DIBUTUHKAN PRAKTIKUM**	**JENIS**
PC / Laptop CPU	**≥ 4 Cores**
PC / Laptop RAM	**≥ 8 GB**
PC / Laptop *Storage*	**≥ 10 GB**

***SOFTWARE* YANG DIBUTUHKAN PRAKTIKUM**
Android *Studio* / *Visual Studio Code*

B.3 Materi Praktikum

B.3.1 Perangkat Android

Android merupakan salah satu jenis *operating system mobile* berbasis linux yang dikembangkan oleh Android Inc pada tahun 2003 di California, Amerika Serikat. Android Inc didirikan oleh 4 orang yang ahli dibidang IT, yakni Andy Rubin, Rich Minner, Nick Sears dan Chris White. Pada bulan Agustus 2005, perusahaan besar ternama, yakni Google melakukan proses akuisisi terhadap

perusahaan Android Inc. Pada tanggal 23 September 2008, sistem operasi Android berhasil dirilis ke publik bersamaan dengan pembentukan sebuah konsorsium *Open Handset Alliance* (OHA) yang terdiri dari beberapa perusahaan *hardware*, *software*, dan telekomunikasi. Selain itu, beberapa perusahaan ternama seperti Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, 21 dan Nvidia juga ikut tergabung ke dalam konsorsium tersebut. Sistem android bersifat *open-source* yang membuat seluruh vendor pengembang *smartphone* diberikan akses untuk melakukan *custom* terhadap android versi mereka sendiri

Sejak perilisannya pada tahun 2008, sistem operasi Android telah mengalami perkembangan yang begitu pesan hingga sampai saat ini. Tercatat bahwa sudah ada sekitar 23 jenis sistem Android yang telah dikembangkan, antara lain:

Tabel B.1 Perkembangan Sistem Operasi Android

NO	NAMA SISTEM	TAHUN PERILISAN
1	Android 1.0 (Alpha)	23 September 2008
2	Android 1.1 (Beta)	9 Februari 2009
3	Android 1.5 (Cupcake)	27 April 2009
4	Android 1.6 (Donut)	15 September 2009
5	Android 2.0 (Éclair)	26 Oktober 2009
6	Android 2.2 (Froyo)	20 Mei 2010
7	Android 2.3 (Gingerbread)	6 Desember 2010
8	Android 3.0 (Honeycomb)	22 Februari 2011
9	Android 4.0 (Ice Cream Sandwich)	18 Oktober 2011
10	Android 4.1 (Jelly Bean)	9 Juli 2012
11	Android 4.4 (KitKat)	31 Oktober 2013
12	Android 5.0 (Lollipop)	4 November 2014
13	Android 6.0 (Marshmallow)	29 September 2015
14	Android 7.0 - 7.1 (Nougat)	22 Agustus 2016
15	Android 8.0 - 8.1 (Oreo)	21 Agustus 2017
16	Android 9 (Pie)	6 Agustus 2018

NO	NAMA SISTEM	TAHUN PERILISAN
17	Android 10 (Android Q)	3 September 2019
18	Android 11 (Red Velvet Cake)	8 Spetember 2020
19	Android 12 (Snow Cone)	4 Oktober 2021
20	Android 13 (Tiramisu)	15 Agustus 2022
21	Android 14 (Upside Down Cake)	4 Oktober 2023
22	Android 15 (Vanilla Ice Cream)	3 September 2024
23	Android 16 (Baklava)	10 Juni 2025

B.3.2 Dart

Dart merupakan bahasa pemrograman *open source* berorientasi objek yang dikembangkan oleh Lars Bark dan Kasper Lund di Google pada tahun 2011 yang kemudian diresmikan pada November 2013. Bahasa pemrograman ini awalnya dirancang khusus untuk pengembangan aplikasi *website*. Namun, seiring dengan berkembangnya zaman bahasa pemrograman ini telah dapat digunakan untuk mengembangkan berbagai aplikasi, seperti perangkat *mobile*, *Internet of Things* (IoT), *game, server, desktop*, dan lain-lain.

Bahasa pemrograman ini mengikuti gaya penulisan C-*style* yang membuat sintaks dari bahasa ini mirip dengan sintaks yang ada pada beberapa bahasa pemrograman yang berorientasi objek lainnya, seperti Java, C#, JavaScript, Kotlin, dan lain-lain (Swathiga, 2022). Terdapat beberapa fitur yang disediakan oleh bahasa pemrograman ini, yang membuat bahasa pemrograman ini menjadi populer dan banyak digunakan, antara lain:

1. ***Open source***

   Bahasa pemrograman ini bersifat *open-source* yang membuat bahasa ini dapat digunakan oleh siapa pun secara gratis.

2. ***Object oriented***

   Bahasa pemrograman ini mendukung seluruh konsep dasar dari pemrograman objek, seperti *class*, *inheritence*, *abstract*, *encaptulation*, dan *polymorprhism*.

3. ***Simple syntax***

   Bahasa pemrograman dart memiliki *syntax* yang sederhana dan mudah untuk dipelajari.

4. ***Cross platform***

   Bahasa pemrograman ini dapat dijalankan diberbagai sistem operasi berbeda, seperti windows, Linux, Unix, MacOS, dan sistem operasi lainnya.

5. ***Multiplatform***

   Bahasa pemrograman ini mendukung pengembangan aplikasi *multiplatform*, sehingga *programmer* dapat membangun aplikasi untuk Android dan iOS secara bersamaan dengan menggunakan kode yang sama

6. ***Garbage collection***

   Dart menyediakan sebuah sistem yang dapat mengelola memori secara otomatis dan dapat digunakan untuk mencegah terjadinya kebocoran memori.

7. ***Asynchronus***

   Bahasa dart menyediakan konsep *asynchronus* dengan fitur *async* dan *await* yang dapat meningkatkan kinerja *programmer*. Hal ini memungkinkan dart untuk menjalankan berbagai tugas secara bersamaan.

8. ***Extensive libraries***

   Dart menyediakan berbagai *library* bawaan yang dapat digunakan, seperti *Software Development Kit* (SDK), *core*, *math*, *asynchronous*, *convert*, html, *input-output* (IO), dan masih banyak lagi *library* yang dapat digunakan.

B.3.2.1 Tipe Data

Sama seperti bahasa pemrograman pada umumnya, dart memiliki beberapa tipe data yang digunakan untuk menentukan jenis dari nilai yang tersimpan dalam sebuah variabel. Berikut merupakan beberapa tipe data yang dapat digunakan dalam bahasa pemrograman dart:

1 ***String***

String merupakan tipe data yang paling umum digunakan disetiap bahasa pemrograman. Tipe data ini akan membaca *value* yang diterima sebagai sebuah kalimat. Tipe data ini ditandai dengan penggunaan *keyword* ***string*** atau *value* yang diapit dengan petik 2 (“…”).

2 ***Integer***

Tipe data *integer* merupakan salah satu tipe data numerik yang akan merepresentasikan nilai bilangan bulat, seperti 1, 3, 10, -30, dan sebagainya. Tipe data ini ditandai dengan penggunaan *keyword* ***int***.

3 ***Double***

Double merupakan jenis lain dari tipe data numerik yang akan menerima nilai bilangan desmial atau *float*, seperti 3.14, 5.7, 11.8, dan sebagainya. Tipe ini ditandai dengan penggunaan *keyword* ***double***.

4 ***Number***

Tipe data *number* merupakan gabungan dari tipe data *integer* dan *double*. Tipe data ini dapat merepresentasikan bilangan bulat maupun bilangan desimal. Penggunaan tipe data ini ditandai dengan *keyword* ***num*** yang digunakan.

5 ***Boolean***

Boolean merupakan tipe data yang hanya menyimpan nilai ***true*** atau ***false*** terhadap sebuah nilai. Tipe data ini dapat digunakan dengan memanggil *keyword* ***bool***.

6 ***Dynamic***

Dynamic merupakan tipe data yang fleksibel yang dimiliki oleh bahasa pemrograman dart. Tipe data ini dapat berupa *string*, *integer*, *double*, *number* dan lain sebagainya, yang dapat berubah-ubah sesuai dengan *value* yang ditentukan. Hal tersebut dapat terjadi, karena tipe data ini akan menentukan jenisnya ketika sudah ada nilai yang diterima. Tipe data ini ditandai dengan penggunaan *keyword* ***dynamic***.

7 ***List***

List merupakan tipe data yang dapat menampung berbagai data ke dalam suatu objek, seperti *string*, *integer*, *double*, *number*, dan *boolean*. Penggunaan tipe data ini ditandai dengan *keyword* ***list*** dan penggunaan kurung siku ([⋯]) yang mengapit data-data.

8 ***Map***

Map merupakan tipe data yang akan menyimpan sekumpulan data dalam format ***key:value***. Dalam tipe data ini, seluruh data akan disimpan ke dalam sebuah *key* dan untuk mengakses data tersebut, pengguna dapat mengakses *key* yang menyimpan datanya. Tipe data ini ditandai dengan penggunaan format *key:value* dalam menyimpan datanya dan data tersebut diapit oleh penggunaan kurung kurawal ({⋯}).

B.3.2.2 Control Flow

Bahasa pemrograman dart memiliki 4 sistem percabangan yang dapat digunakan, antara lain:

1. ***If statement***

Statement if merupakan salah satu sistem percabangan sederhana yang akan menjalankan sebuah blok kode program jika suatu kondisi terpenuhi atau saat kondisi tersebut bernilai *true*. Berikut merupakan *syntax* yang digunakan dalam *if statement:*

```
if (condition) {
  // body of if
}
```

Pada *statement if,* sistem akan memeriksa *boolean expression* yang dihasilkan dari kondisi yang diamati. Jika kondisi tersebut menghasilkan nilai *true,* maka sistem akan menjalankan kode program yang ada di dalam *statement if* tersebut. Namun, jika kondisi tersebut bernilai *false* maka sistem tidak akan menjalankan blok kode program apa pun.

2. ***If-else statement***

Statement ini akan menjalankan 2 kondisi yang berbeda, bergantung kepada nilai *boolean* yang diterima. Berikut merupakan *syntax* yang digunakan dalam *statement if-else:*

```
if (condition) {
  // body of if
} else {
  // body of else
}
```

Dalam *statement* ini sistem akan memeriksa terlebih dahulu *boolean* expression yang akan dihasilkan dari kondisi yang diamati. Jika kondisi tersebut menghasilkan nilai *true*, maka sistem akan menjalankan kode program di dalamnya. Namun, jika kondisi yang didapati adalah *false*, maka program tetap akan menjalankan sebuah kode yang berada pada *statement else.*

3. ***Else-if statement***

Statement else-if merupakan perkembangan dari *statement if-else.* Dalam *statement* ini, programmer dapat memasukkan beberapa kondisi yang akan dicek oleh sistem, sehingga *output* yang dihasilkan lebih bervariasi. Berikut adalah potongan kode dari *statement else-if:*

```
if (condition1) {
  // body of if
} else if (condition2) {
  // body of if
} else {
  // statement
}
```

Dalam kode program tersebut, sistem akan melakukan pengecekkan terlebih dahulu terhadap kondisi pada *statement if.* Jika kondisi tersebut terpenuhi, maka sistem akan menjalankan blok kode program yang ada di dalamnya. Namun, jika kondisi tersebut tidak terpenuhi maka sistem akan

menjalankan perintah *else if* yang memiliki sebuah kondisi. Jika pada *statement else if* kondisi tersebut terpenuhi, maka sistem akan menjalankan kode program di dalamnya. Namun, jika kondisi tidak terpenuhi maka sistem akan memeriksa kondisi *else if* lainnya. Jika kondisi *else if* yang lain juga tidak terpenuhi, maka sistem akan menjalankan blok program yang ada pada *statement else.*

4. ***Switch-case statement***

Switch-case statement merupakan satu-satunya *control flow* yang dimiliki oleh dart dengan mekanisme yang berbeda. Dalam *statement* ini, seluruh *case* atau kondisi akan diperiksa secara bersamaan. Jika salah satu *case* terpenuhi, maka sistem akan menjalankan blok program tersebut tanpa menjalankan blok program yang lain. Namun, jika dari beberapa *case* yang telah didefinisikan tidak ada yang sesuai, maka sistem akan menjalankan blok program pada *statement default.* Berikut merupakan *syntax* pada *statement switch-case:*

```
switch(variable_expression) {
   case constant_expr1:
      // statements;
      break;
   case constant_expr2:
      //statements;
      break;
   default:
      //statements;
      break;
}
```

B.3.3 Golang

Golang atau Go merupakan bahasa pemrograman yang dikembangkan oleh Google pada tahun 2007 dan bersifat *open source.* Golang mulai dikeperkenalkan ke publik pada tahun 2009. Sama seperti bahasa pemrograman pada umumnya, Golang memiliki sebuah *framework* yang digunakan untuk membangun berbagai aplikasi, seperti Goji, Revel, Martini, Gocraft, Buffalo, Echo

dan Gin. Golang dapat digunakan untuk mengembangkan *website*, *cloud* dan jaringan. Bahasa pemrograman ini memiliki beberapa keunggulan dan kekurangannya sendiri. Keunggulan yang ditawarkan oleh bahasa pemrograman ini, antara lain:

1. **Kecepatan**

   Golang merupakan bahasa pemrograman yang dikompilasi artinya kode yang ditulis bisa langsung diterjemahkan dengan format yang dapat dimengerti oleh prosesor.

2. **Mudah dipelajari**

   Golang termasuk ke dalam bahasa pemrograman yang mudah untuk dipelajari mulai dari proses instalasi hingga penggunaanya. Sehingga pengguna tidak membutuhkan waktu lama untuk mempelajari Golang.

3. **Punya banyak dukungan**

   Golang dapat digunakan di berbagai OS dan juga terdapat beberapa *Integrated Development Environment* (IDE) dan *text editor* yang mendukung penulisan Go seperti VSCode, Atom, Eclipse, Sublime, InteliJ.

4. **Banyak digunakan *industry***

   Banyak industri menggunakan Golang untuk membangun *environment* perusahaan. Mempelajari Golang memungkinkan menunjang profesi sebagai *programmer*.

Meskipun bahasa ini menawarkan beberapa keunggulan yang menarik, nyatanya terdapat beberapa kekurangan yang dimiliki oleh bahasa pemrograman ini, antara lain:

1. **Memerlukan banyak waktu untuk melakukan tugas**

   Kesederhanaan Golang justru membuat bahasa pemrograman ini kurang deskriptif dibandingkan dengan pemrograman yang lain. Jika *developer* menulis suatu perintah pada bahasa pemrograman lain mungkin *developer* akan lebih banyak menulis baris code yang dibutuhkan di bahasa Go.

2. **Tergolong bahasa pemrograman yang baru**

   Sebagai bahasa pemrograman yang tergolong baru kelemahannya adalah sulit diimplementasikan ke *platform* yang tidak terintegrasi dengan Golang. *Developer* akan merasa sedikit kesulitan dengan *library* yang ada.

3. **Tidak mendukung *generic function***

   Bahasa pemrograman lain mendukung *generic function* sehingga *developer* dapat menggunakan kembali kode atau *function* yang sebelumnya sudah dibuat. Kurangnya dukungan Go terhadap *generic function* menjadikan pengembangan tidak efisien.

Bahasa pemrograman ini telah diimplementasikan ke dalam beberapa aplikasi besar, seperti:

1. **Uber**. Golang diimplementasikan ke dalam *maps* yang berfungsi sebagai penuntun dalam melakukan sebuah perjalanan yang lebih cepat.
2. ***Slack***. Golang digunakan di beberapa fitur, seperti posting pesan, notifikasi, kalender dan lainnya.
3. ***Dropbox***. Golang digunakan untuk mengelola layanan *cloud-storage sharing*.
4. **Riot Games**. Golang digunakan untuk membangun *environment* dan penulisan kode dengan cepat.

B.3.4 Flutter

Flutter merupakan sebuah *framework open-source* yang dikembangkan oleh perusahaan ternama bernama Google pada tahun 2015. Flutter dirancang khusus bagi para *developer* yang ingin mengembangkan aplikasi *mobile* berbasis Android atau iOS. Selain itu, Flutter juga dapat digunakan untuk mengembangkan aplikasi berbasis *website* hingga *desktop*. *Framework* ini dapat digunakan dengan mudah oleh programmer yang memiliki pengetahuan terkait bahasa pemrograman dart serta konsep widget dimana seluruh elemen yang terlihat pada aplikasi, seperti *text*, *button*, *image*, dan elemen-elemen lainnya merupakan sebuah widget.

Selain konsep dasar widget, Flutter juga memiliki konsep *state* dalam membangun sebuah halaman aplikasi dimana terdapat 2 jenis *state* widget yang harus dimengerti oleh *programmer*, yakni ***stateless widget*** dan ***stateful widget***. *Stateless widget* merupakan jenis *state* widget yang cocok digunakan untuk membuat halaman yang statis atau tidak memerlukan perubahan secara teratur. Sedangkan *stateful widget* merupakan jenis *state* widget yang cocok digunakan untuk membuat halaman yang dinamis atau isinya dapat berubah sepanjang waktu.

Framework Flutter memiliki beberapa keunggulan dibandingkan dengan *framework* pengembangan aplikasi *mobile* lainnya yang membuat *framework* ini menjadi populer dan disenangi oleh para *developer*. *Framework* ini memiliki kemampuan untuk mengembangkan aplikasi *mobile* yang ***multiplatform*** dengan menggunakan kode yang sama, sehingga *programmer* dapat membangun dua aplikasi dengan sistem operasi yang berbeda sekaligus dengan cepat tanpa membedakan kode program yang digunakan. Flutter juga memiliki fitur ***hot reload*** yang dapat membantu *developer* dalam melihat perubahan yang telah diberikan kepada aplikasi tanpa perlu menjalankan kembali kode program dari awal.

B.3.5 *Firebase*

Firebase merupakan sebuah layanan *backend* yang disediakan oleh Google yang dapat membantu *developer* dalam mengembangkan aplikasi miliknya. *Firebase* merupakan platform yang bersifat *realtime*, artinya ketika terjadi perubahan data pada *firebase*, maka informasi yang ditampilkan pada setiap perangkat yang terhubung dengan *firebase* akan di*update* pada saat perubahan data terjadi.

Firebase didirikan oleh Andrew Lee dan James Tamplin pada tahun 2011. Pada saat itu *firebase* hanya menyediakan satu layanan yang dapat digunakan oleh para pengguna, yakni *realtime database*. Layanan tersebut merupakan layanan *database* NoSQL yang seluruh datanya disimpan ke dalam format JSON dan datanya akan disesuaikan secara *realtime* ke setiap user yang terhubung dengan *firebase*. Kemudian pada Oktober 2014, *firebase* diakuisisi oleh Google dan berkembang menjadi layanan pengembangan aplikasi. Saat ini *firebase* telah

memiliki beberapa layanan yang dapat digunakan dalam mengembangkan aplikasi, seperti *realtime database*, *cloud firestore*, *cloud storage*, *firebase authentication*, *firebase analytics*, *firebase hosting*, dan masih banyak lagi layanan yang disediakan oleh *firebase*.

B.3.6 Android *Studio*

Android *studio* merupakan *software Integrated Development Environment* (IDE) resmi yang digunakan dalam pengembangan aplikasi Android. *Software* ini hadi menggantikan Eclipse sebagai IDE pengembangan aplikasi android sejak perilisannya pada tahun 2014. *Software* ini bersifat *open source* dan dapat diunduh di berbagai sistem operasi yang ada, seperti windows, Mac OS, dan linux.

Software ini dilengkapi dengan *intellegent code editor* yang mampu mengolah dan menganalisis kode secara lengkap yang menjadikan *developer* semakin produktif. Selain itu, Android Studio dilengkapi dengan *code templates* dan Github *integration* yang memudahkan *develover* Android dalam mengembangkan aplikasi mereka dari sample-sample kode yang telah disediakan ataupun meng*import* dari Github. Android Studio dilengkapi dengan *emulator* yang mencakup semua devices, baik ukuran maupun bentuk. Keunggulan tersebut memudahkan *developer* untuk melihat hasil *project* yang telah dikembangkan dari berbagai *device* yang ada.

Software ini juga memiliki beberapa kekurangan yang dapat menjadi bahan pertimbangan oleh *developer* untuk menggunakannya dalam pengembangan suatu aplikasi Android. Salah satu kelemahan yang dimiliki oleh *software* ini adalah ukuran file dari *software* tersebut yang mencapai hampir 1 GB yang seringkali menjadi masalah saat mendownload *software* tersebut.

B.3.7 *Visual Studio Code*

Visual studio code atau yang biasa dikenal dengan vscode merupakan salah satu *text editor* yang ringan, gratis, dan fleksibel yang dapat digunakan oleh seorang *programmer* dalam menuliskan suatu kode program yang digunakan dalam proses pengembangan sistem. Vscode merupakan *text editor multiplatform* yang

dikembangkan oleh Microsoft dan dapat dijalankan diberbagai sistem operasi, seperti windows, MacOS, dan linux. *Visual studio code* memiliki beberapa keunggulan yang ditawarkan kepada para programmer dibandingkan dengan *text editor* lainnya, seperti:

1. **Fitur lengkap**

   Visual studio code memiliki *extension marketplace* yang membuat *programmer* dapat mencari dan menambahkan berbagai jenis *extension* yang ingin digunakan secara gratis.

2. **Ringan dan cepat**

   Microsoft selaku pengembang *visual studio code* merancang *text editor* ini agar tetap ringan dan cepat ketika sedang menggunakannya, meskipun terdapat berbagai fitur yang disediakan. *Text editor* ini dapat berjalan dengan baik dan lancar di perangkat manapun, baik yang memiliki sumber daya yang luas maupun terbatas.

3. **Mendukung banyak bahasa pemrograman**

   Visual studio code memungkinkan *programmer* untuk menuliskan kode program dengan menggunakan berbagai jenis bahasa pemrograman yang ada. Hal tersebut dapat dilakukan karena *visual studio code* memiliki *extension marketplace* yang di dalamnya terdapat berbagai jenis bahasa pemrograman yang dapat diunduh dan digunakan, seperti Java, JavaScript, Python, Golang, C++, C#, Dart, dan bahasa pemrograman lainnya.

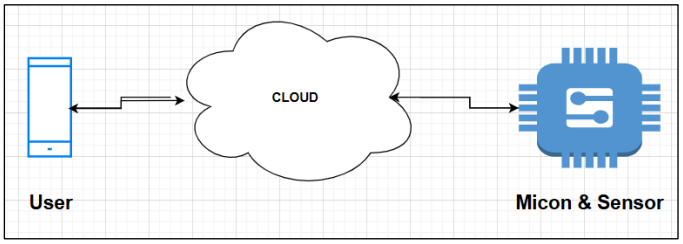
4. ***Multiplatform***

   Meskipun vscode dikembangkan oleh Microsoft. Namun, *text editor* ini dapat digunakan dan dijalankan diberbagai perangkat OS diluar Microsoft, seperti linux, Mac OS, dan sistem operasi lainnya.

B.3.8 *Internet of Things*

Internet of Things (IoT) merupakan sebuah konsep yang memungkinkan terhubungnya perangkat dengan lainnya melalui internet untuk melakukan aktivitas tertentu. Cara kerja IoT pada dasarnya membutuhkan 3 komponen, yaitu sensor, *gateway* dan *cloud* sehingga melibatkan pencarian, pengolahan dan pengiriman.

Awalnya sensor mengambil dan mengumpulkan data yang kemudian akan ditransmisikan ke *cloud* menggunakan *gateway*. *Gateway* dari arah lain juga dapat menjadi *trigger* bagi *output* perangkat IoT itu sendiri, seperti menghidupkan atau mematikan. Contoh implementasi dari cara kerja IoT dapat dilihat pada Gambar B.1.



Gambar B.1 Ilustrasi *Internet of Things*

Dari arah *microcontroller* dan sensor terdapat ESP32, sensor DHT11 dan juga lampu, Sensor DHT11 akan menangkap suhu dan dibuat kondisi tertentu apabila suhu di bawah 19°C maka lampu akan menyala otomatis. Keadaan ini yang akan dikirimkan ke *cloud*. Di dalam *cloud* berisi *database* terhadap perangkat IoT dan dari pihak *user* akan membaca informasi dari *cloud*. *Cloud* yang digunakan bisa berupa *service firebase*, *thingspeak* atau bahkan *cloud* yang dibangun sendiri (*back end*). Hal ini juga berlaku dari pihak *user* melakukan *trigger* terhadap IoT dengan memanipulasi *database*.

B.3.9 Postman

Postman adalah sebuah *platform* yang digunakan untuk pengujian dan pengelolaan API (*Application Programming Interface*). Alat ini sangat populer di kalangan *developer* karena memungkinkan *developer* untuk membuat, mengirim, dan menganalisis permintaan API tanpa perlu menulis kode secara manual. Postman menyediakan antarmuka yang intuitif untuk mengeksplorasi berbagai *endpoint*, mengatur parameter, menambahkan headers, dan melihat *respons* dari server secara *real-time*. Selain itu, Postman juga mendukung fitur dokumentasi, otomasi pengujian, dan kolaborasi tim dalam satu lingkungan kerja.

Postman pertama kali dikembangkan pada tahun 2012 oleh Abhinav Asthana sebagai ekstensi Chrome, dengan tujuan menyederhanakan proses

pengujian API yang saat itu cukup rumit. Proyek kecil ini kemudian berkembang pesat dan menjadi perusahaan global bernama Postman Inc., yang kini berpusat di San Francisco, California.

Terdapat beberapa fungsi utama yang disediakan oleh postman sebagai *platform* pengujian API, antara lain:

1. **API *Client***

   API *Client* merupakan fitur yang memungkinkan *developer* untuk mengirim permintaan HTTP ke berbagai *endpoint*. *Developer* dapat mengatur metode HTTP yang ingin digunakan, seperti GET, POST, PUT, hingga DELETE. Fitur ini juga mendukung pengaturan *headers*, *authentication*, dan *body* sesuai kebutuhan. Cocok untuk eksplorasi dan pengujian API secara manual.

2. ***Collection***

   Collection merupakan tempat untuk menyimpan dan mengelola kumpulan permintaan API. Dengan struktur folder yang rapi, *developer* dapat mengelompokkan permintaan berdasarkan fitur atau modul. *Collection* juga bisa digunakan dalam pengujian otomatis dan dokumentasi. Ini sangat membantu menjaga keteraturan dalam proyek besar.

3. ***Environment* & *Variables***

   Fitur *environment* memungkinkan *developer* membuat konfigurasi yang berbeda untuk berbagai tahap, seperti *development*, *testing*, atau *production*. *Developer* dapaat menggunakan variabel agar tidak perlu mengedit permintaan satu per satu saat berpindah lingkungan. Misalnya, URL dasar dan token API dapat diatur dalam variabel. Ini membuat *workflow* lebih fleksibel dan efisien.

4. ***Test Script***

   Postman mendukung penulisan skrip uji berbasis JavaScript untuk memvalidasi hasil *respons* API. *Developer* dapat menguji status kode, nilai dalam JSON, dan performa API. Hasil pengujian dapat ditampilkan secara langsung di antarmuka Postman. Hal tersebut dapat membantu *developer* dalam menemukan *bug* lebih cepat dan menjaga kualitas API.

5. ***Mock Server***

Dengan *mock server*, *developer* dapat membuat simulasi *respons* dari API yang belum selesai dibuat. Ini memungkinkan tim *frontend* tetap bekerja meskipun *backend* belum siap. *Developer* dapat mengatur *respons* sesuai dengan skenario tertentu untuk tahap pengujian. Fitur ini sangat berguna dalam proses *prototyping* dan integrasi awal.

6. **API *Documentation***

Postman dapat menghasilkan dokumentasi API secara otomatis dari koleksi yang telah dibuat. Dokumentasi tersebut bersifat interaktif dan mudah untuk dipahami, bahkan oleh orang non-teknis.

7. ***Collaboration* & *Workspaces***

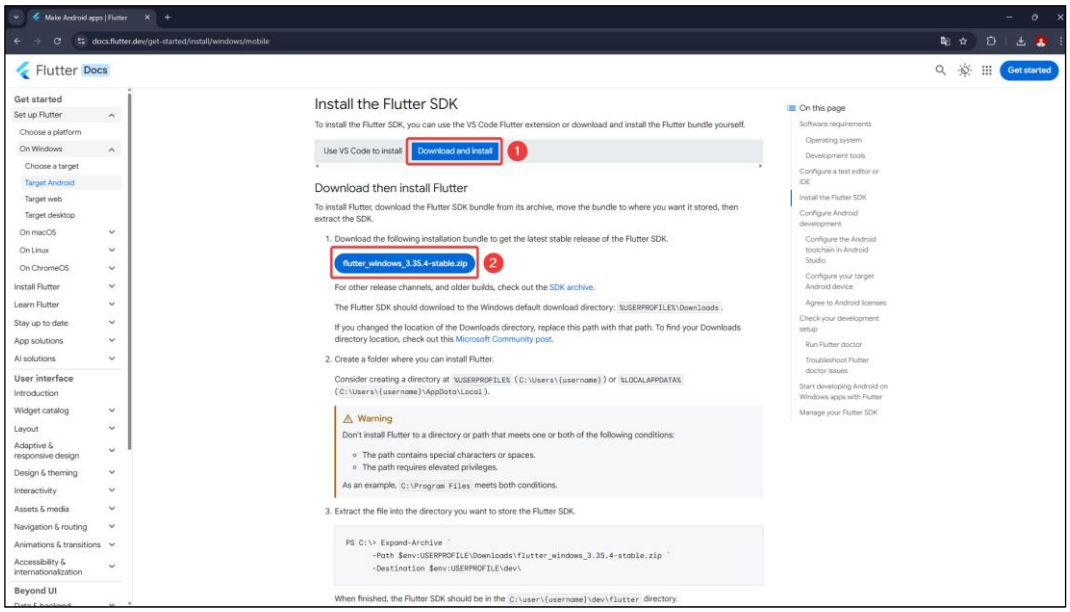
Postman menyediakan fitur *workspaces* untuk berkolaborasi dengan tim secara *online*. Tim pengembang dapat berbagi koleksi, variabel, hasil uji, dan dokumentasi dengan anggota tim yang lain secara langsung. Setiap perubahan dapat dilacak dan disinkronkan yang membuat kerja tim menjadi lebih terorganisir dan efisien.

B.4 Instalasi dan *Setup Software* Praktikum

B.4.1 Instalasi dan *Setup* Flutter SDK

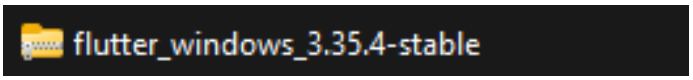
Langkah pertama yang harus dilakukan agar *framework* Flutter dapat digunakan untuk mengembangkan sebuah *software* adalah melakukan instalasi terhadap Flutter SDK. Berikut merupakan langkah-langkah dalam proses instalasi Flutter SDK:

1. Bukalah link berikut untuk men*download* Flutter SDK:
   https://docs.flutter.dev/get-started/install/windows/mobile
2. Carilah *section* **"*Install* the Flutter SDK"** dengan cara melakukan *scroll* kebawah.
3. Pilihlah menu ***download and install*** yang ada pada *section* tersebut.
4. Pilihlah **flutter_windows_3.35.4-stable.zip** untuk meng*install* flutter SDK yang direkomendasikan **(Note: Versi dapat berubah-ubah)**.



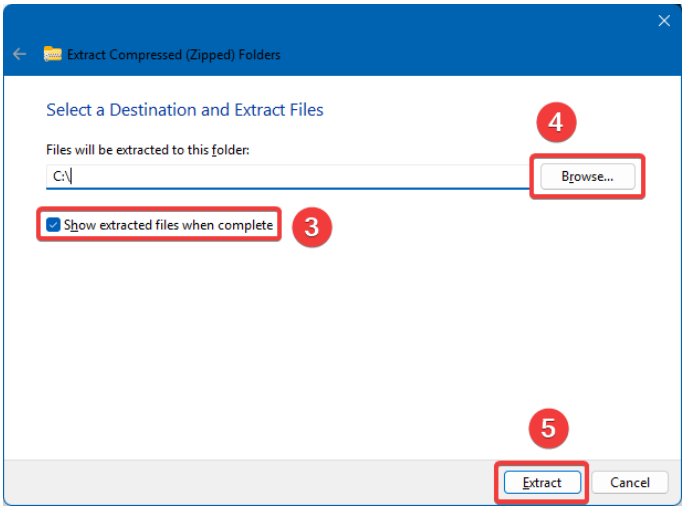
Gambar B.2 Halaman *Website* untuk Men*download* Flutter *SDK*

5. Tunggulah sampai proses *download* Flutter SDK selesai.



Gambar B.3 File Flutter SDK yang Telah di*download*

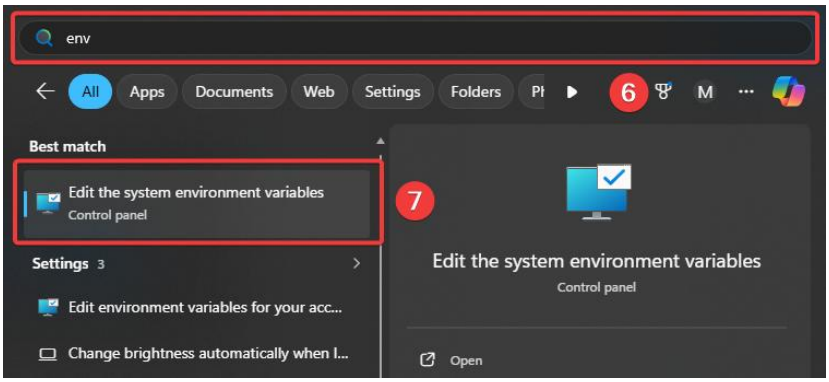
6. Setelah file Flutter SDK telah selesai di*download*, lakukanlah proses *extract* pada file tersebut dengan cara menekan tombol kanan pada *mouse* dan pilihlah opsi ***extract all***.
7. Centanglah opsi **"*Show extracted files when complete*"** yang disediakan.
8. Pilihlah destinasi penyimpanan untuk menyimpan hasil *extract* dari file tersebut dengan menekan menu ***browse*** yang disediakan **(Note: *Destination Storage* untuk menyimpan file hasil ekstraksi dapat disimpan dimana saja)**.
9. Tekanlah tombol ***extract*** untuk melanjutkan proses ekstraksi file dan tunggulah hingga proses ekstraksi file selesai dilakukan.



Gambar B.4 Proses *Extract* File Flutter SDK

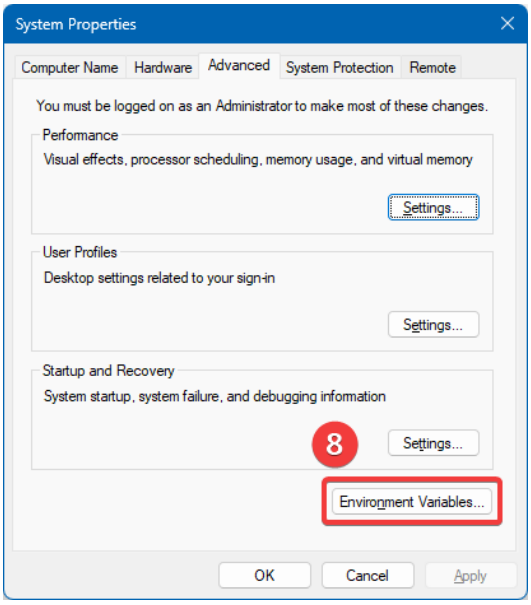
10. Carilah *folder* bernama **Flutter** pada direktori tersebut. Kemudian carilah *folder* bernama **bin** di dalam *folder* tersebut **([Direktori_asal]:\flutter\bin)**

11. Salinlah *path* tersebut dan bukalah *environment variables* dengan cara **masuk ke dalam *search windows* atau dengan menekan logo *windows* pada *keyboard***. Setelah muncul kolom *search*, ketiklah **"env"** pada bagian *search* dan pilihlah menu **"*Edit the system environment variabels*"**.

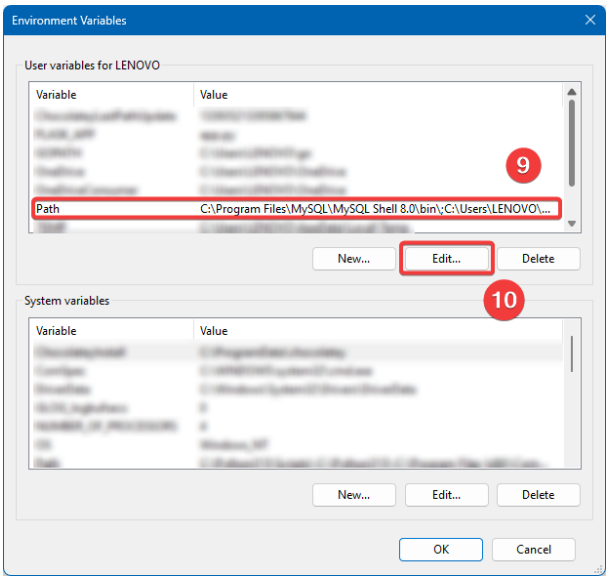


Gambar B.5 Menu *Search* Pada Windows

12. Masuklah ke dalam menu ***environment variables*** kemudian pilihlah menu **path** pada bagian atas dan klik menu **"Edit…"**

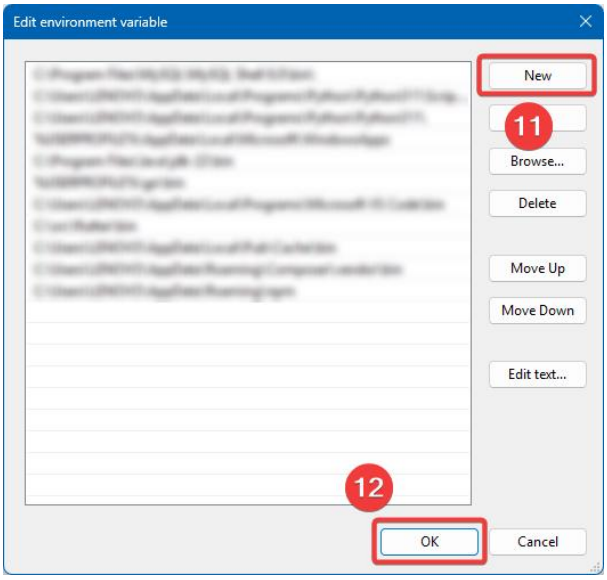


Gambar B.6 Menu *System Properties*



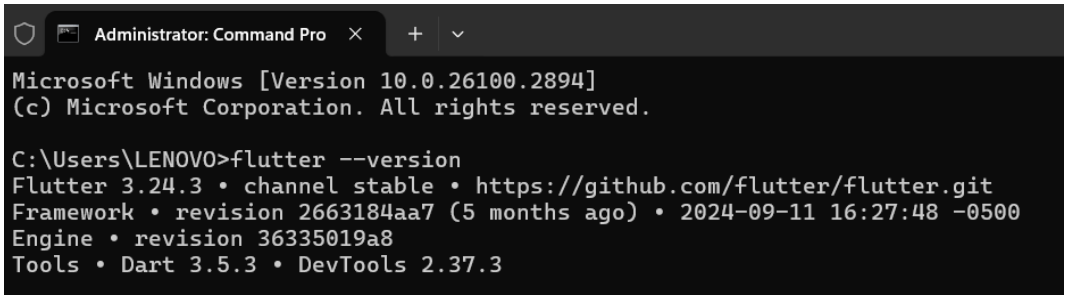
Gambar B.7 Menu *Environment Variables*

13. Pilihlah menu ***new*** pada menu *edit environment variable* dan lakukanlah *paste* terhadap *path* yang telah disalin.
14. Setelah *path* flutter SDK berhasil terpasang, tekanlah tombol **OK** untuk menyimpan konfigurasi *path*.



Gambar B.8 Menu *Edit Environment Variables*

15. Bukalah ***command prompt*** pada perangkat dan ketik perintah **flutter --version.** Proses instalasi flutter SDK dinyatakan selesai, jika *output* pada *command prompt* terlihat, seperti pada gambar berikut **(Note: Versi dapat berbeda-beda)**.



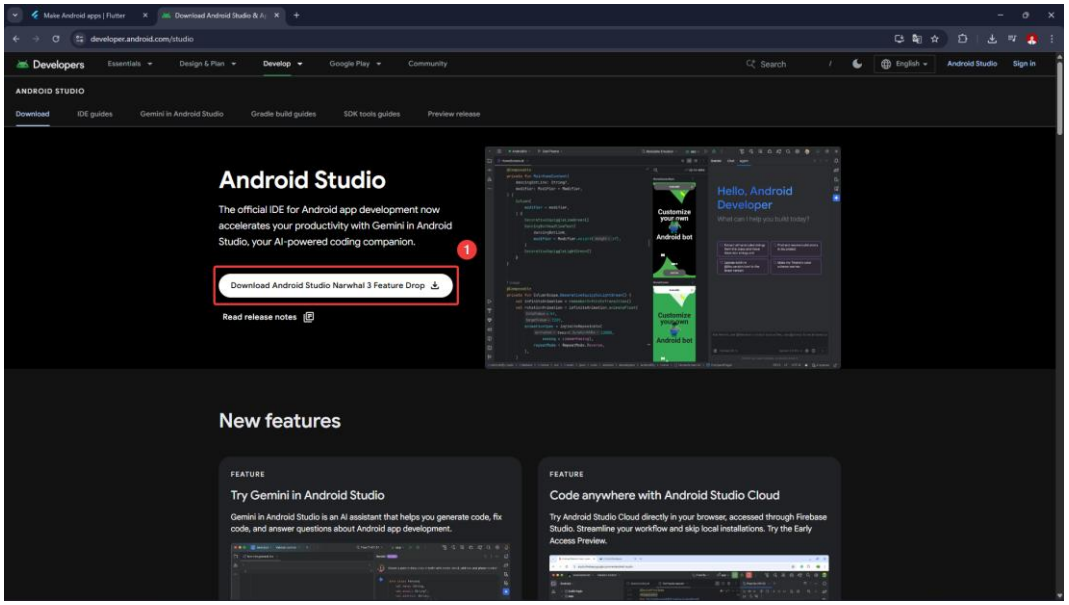
Gambar B.9 *Output* pada *Command Prompt*

B.4.2 Instalasi dan *Setup* Android *Studio*

Setelah proses instalasi Flutter SDK berhasil dilakukan, langkah berikutnya yang harus dilakukan adalah meng*install software* **android *studio*** yang nantinya akan digunakan selama proses praktikum. Berikut merupakan langkah-langkah dalam meng*install software* android *studio*:

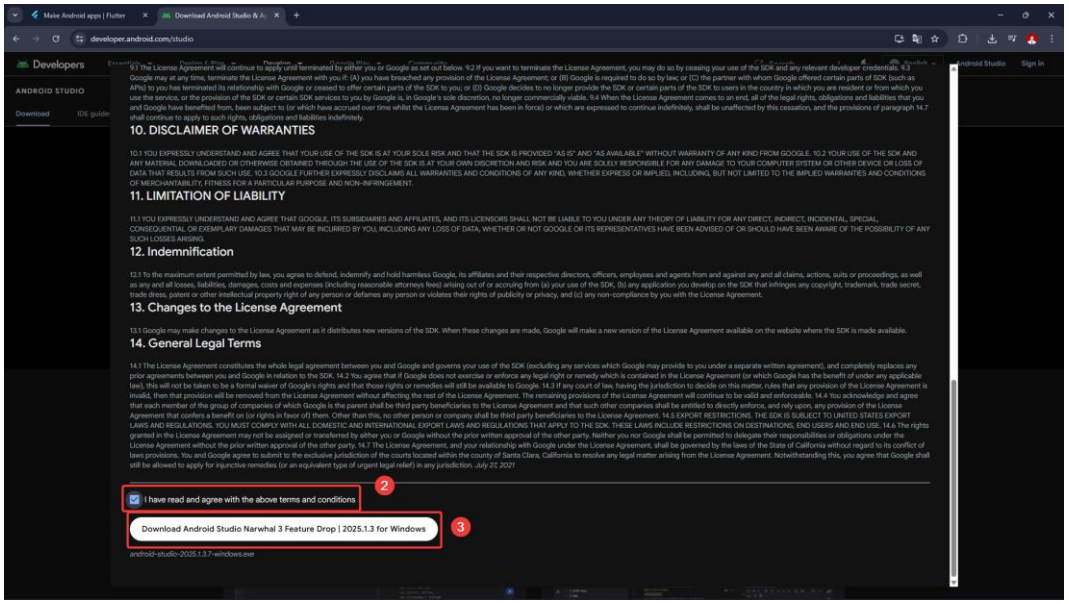
1. Bukalah link berikut untuk men*download software android studio*: **https://developer.android.com/studio**

2. Tekanlah tombol **Download Android Studio Narwhal 3 Feature Drop** (**Note: Versi Android Studio dapat berubah-ubah**).



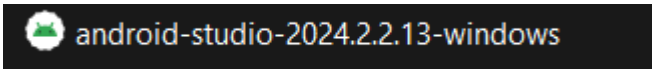
Gambar B.10 Halaman *Website* untuk Men*download* Android *Studio*

3. Setelah menekan tombol tersebut, maka *website* akan menampilkan menu **"Terms and Conditions."** Lakukanlah *scroll* hingga mencapai batas bawah dari menu tersebut dan centanglah bagian **"Saya telah membaca dan menyetujui persyaratan dan ketentuan di atas."** Kemudian tekan kembali tombol download yang telah disediakan.



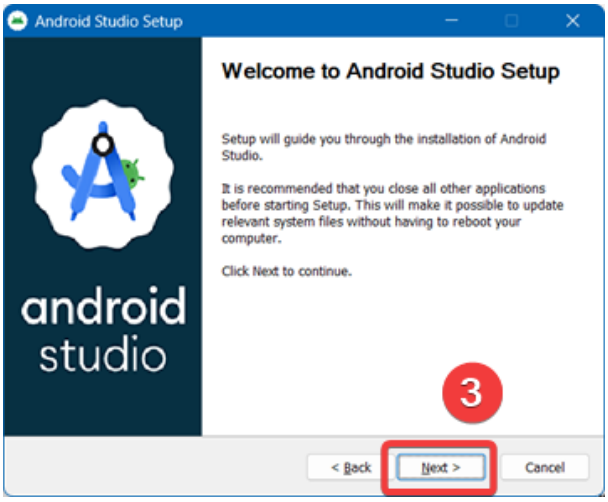
Gambar B.11 Tampilan Menu *Terms and Conditions*

4. Tunggulah sampai proses download android studio selesai.
5. Bukalah file android studio yang telah selesai didownload.



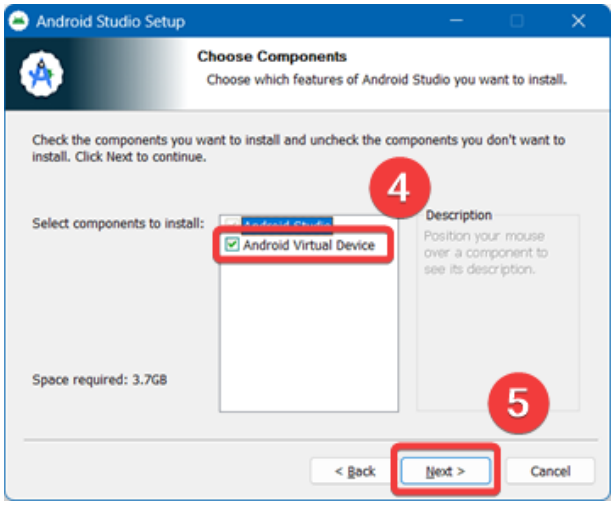
Gambar B.12 File Android *Studio* yang telah di*download*

6. Setelah masuk ke dalam menu *setup* untuk android studio, pilihlah tombol ***next*** untuk melanjutkan ke langkah berikutnya.



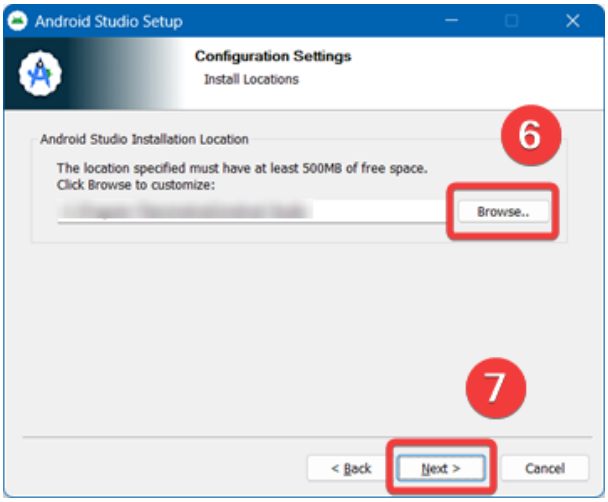
Gambar B.13 Menu *Setup* Android *Studio*

7. Centanglah opsi **Android *Virtual Device*** agar android *studio* menyediakan opsi *virtual device* pada saat proses *running project.* Kemudian tekanlah tombol ***next*** untuk melanjutkan ketahap berikutnya.



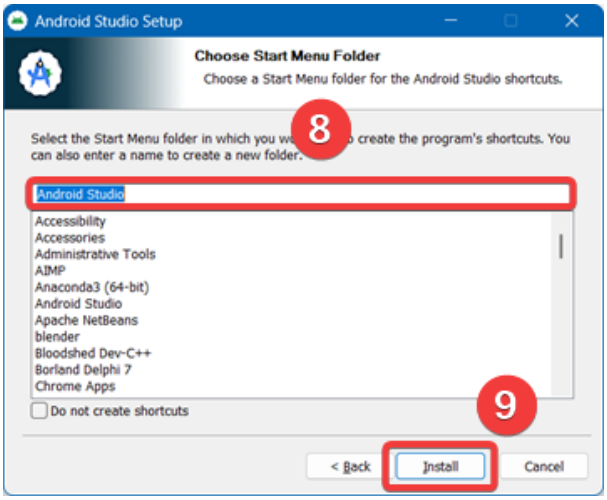
Gambar B.14 Menu *Install Components* Android *Studio*

8. Pilihlah lokasi direktori untuk menyimpan *software* android *studio* pada perangkat dan tekanlah tombol ***next*** untuk melanjutkan ketahap berikutnya.



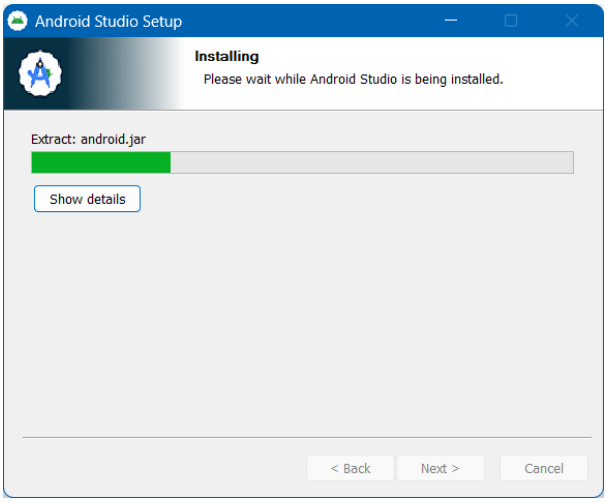
Gambar B.15 Menu *Configuration Location* Android *Studio*

9. Pada menu *choose start menu folder*, pastikan untuk memilih **android studio** kemudian tekanlah tombol ***install*** untuk melakukan proses instalasi terhadap android *studio*.



Gambar B.16 Menu untuk Membuat *Shortcut* Aplikasi Android *Studio*

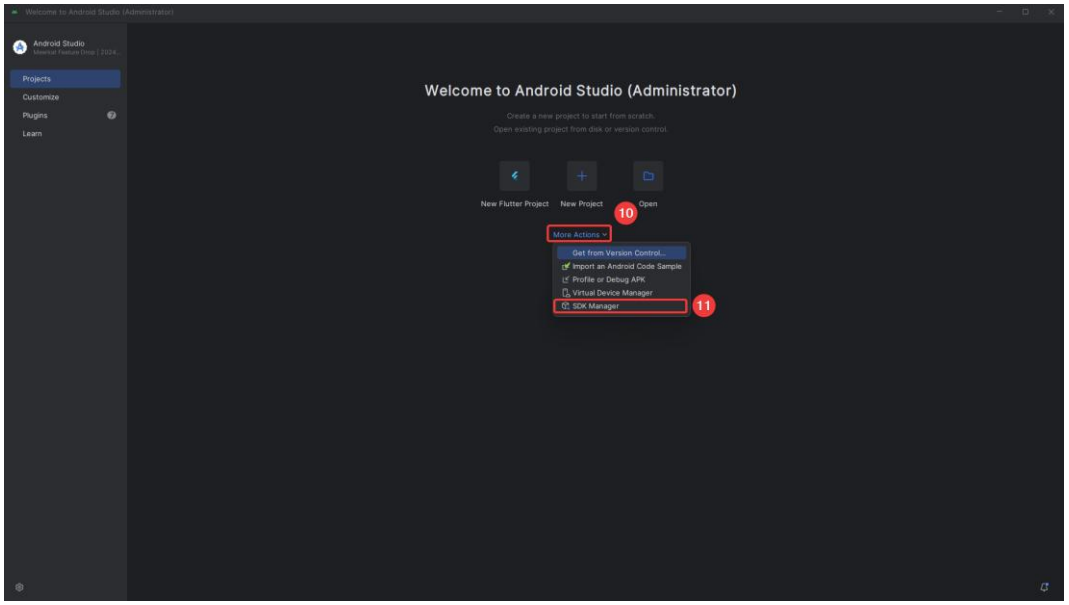
10. Tunggulah sampai proses instalasi android *studio* selesai.



Gambar B.17 Proses Instalasi Android *Studio*

11. Setelah proses instalasi android *studio* selesai, bukalah aplikasi android *studio* yang telah ter*install*.

12. Jika belum pernah membuat *project* dengan menggunakan *software* android *studio*, maka tampilan awalnya akan terlihat, seperti pada Gambar B.18. Bukalah menu **"*More Actions*"** yang telah disediakan oleh aplikasi dan pilihlah *option* **SDK *Manager***.

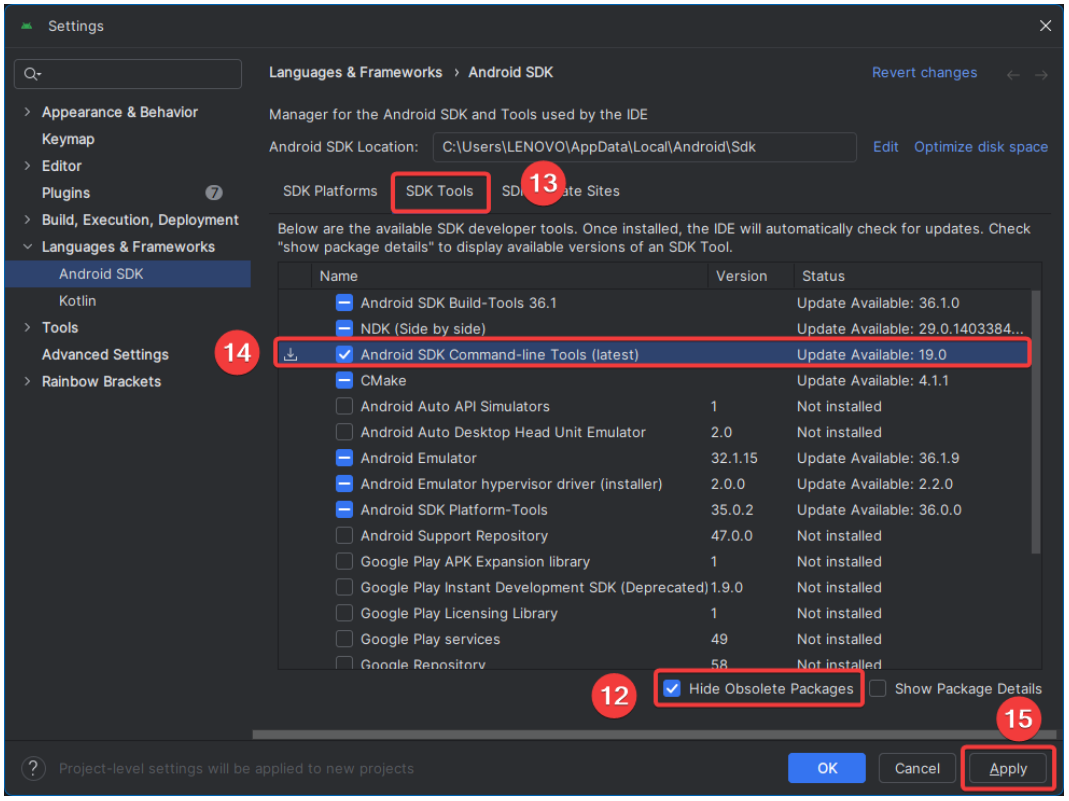


Gambar B.18 Tampilan Aplikasi Android *Studio*

13. Pada menu SDK Manager, centanglah *option* ***Hide Obsolete Packages*** yang terletak di bagian bawah

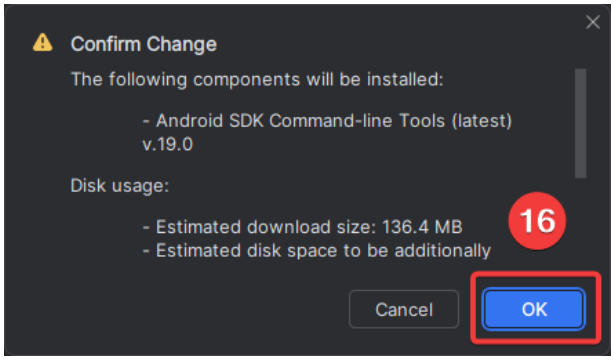
14. Selanjutnya, masuklah ke dalam menu **SDK Tools** kemudian pilihlah **"Android SDK Command-line Tools (latest)"**.

15. Tekanlah tombol **apply** yang berada pada bagian bawah.



Gambar B.19 Menu SDK *Tools* pada Android *Studio*

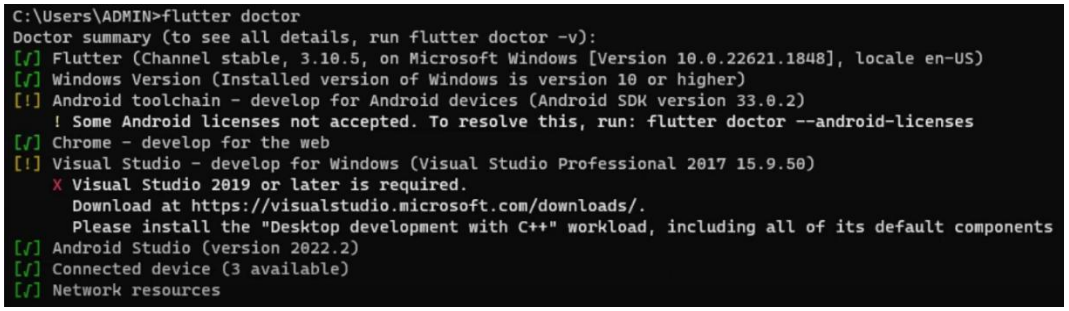
16. Aplikasi android *studio* akan menampilkan jendela baru yang berisikan pesan konfirmasi, seperti pada Gambar B.20. Pada bagian ini tekanlah tombol **OK** untuk menlanjutkan ketahap berikutnya



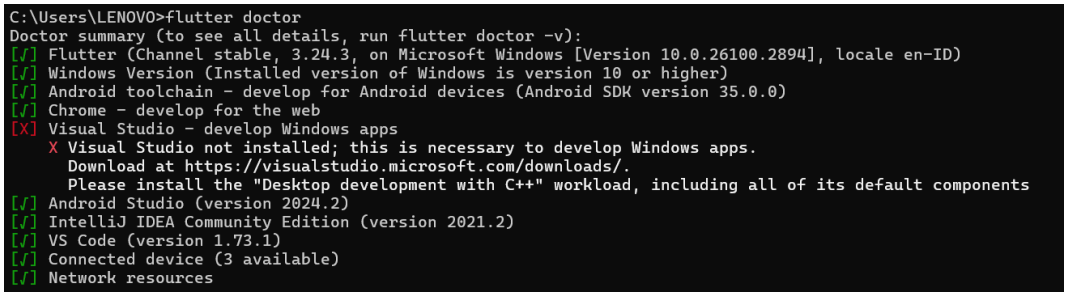
Gambar B.20 Jendela Pesan Konfirmasi

17. Tunggulah sampai proses *download* Android SDK CLT selesai.

18. Setelah selesai, bukalah ***command prompt*** pada perangkat dan ketik perintah **flutter doctor**. Jika *output* yang dihasilkan pada saat menjalankan perintah tersebut terlihat, seperti pada Gambar B.21, maka ketiklah perintah **flutter doctor --android-licenses** ketika proses analisa telah selesai. Namun, jika *output* yang ditampilkan terlihat seperti pada Gambar B.22, maka proses *setup* android *studio* telah selesai.



Gambar B.21 *Output* Ketika Android *Licenses* Belum diambil



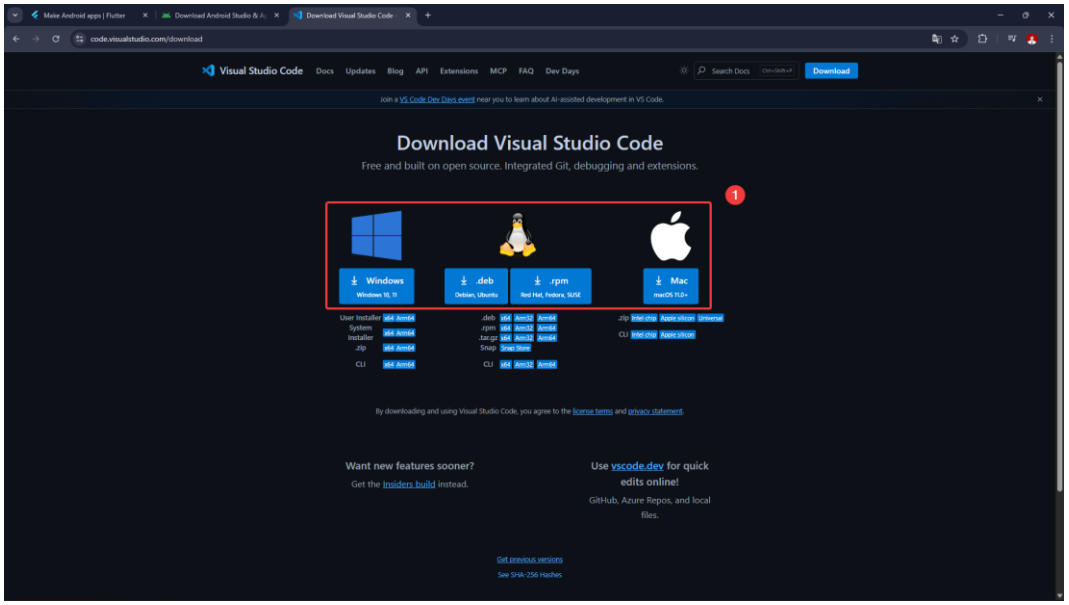
Gambar B.22 *Output* Ketika Android *Licenses* Telah diambil

B.4.3 Instalasi dan *Setup Visual Studio Code*

Setelah proses instalasi Flutter SDK dan *software* android *studio* berhasil dilakukan, langkah berikutnya yang harus dilakukan adalah meng*install software* ***visual studio code*** yang nantinya akan digunakan pada proses praktikum. Berikut merupakan langkah-langkah dalam meng*install software* android studio:

1. Bukalah link berikut untuk men*download software visual studio code*: https://code.visualstudio.com/download#

2. Tekanlah menu *download* yang telah disediakan (**Note: Harap menekan salah satu menu *download* yang disesuaikan dengan sistem operasi perangkat**).

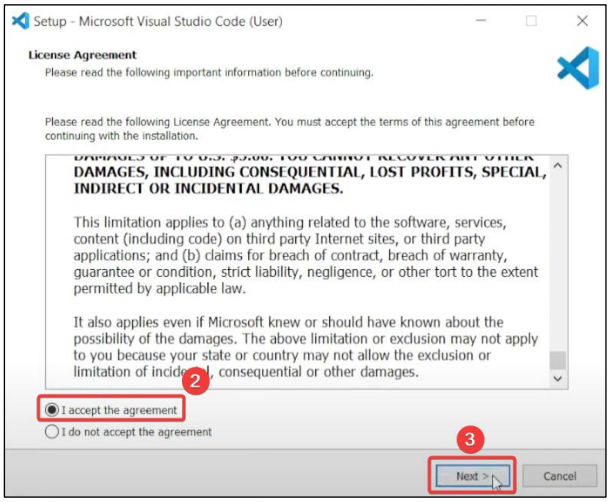


Gambar B.23 Halaman *Website* untuk Men*download Visual Studio Code*

3. Tunggulah sampai proses *download visual studio code* selesai.
4. Bukalah file *visual studio code* yang telah selesai di*download*.

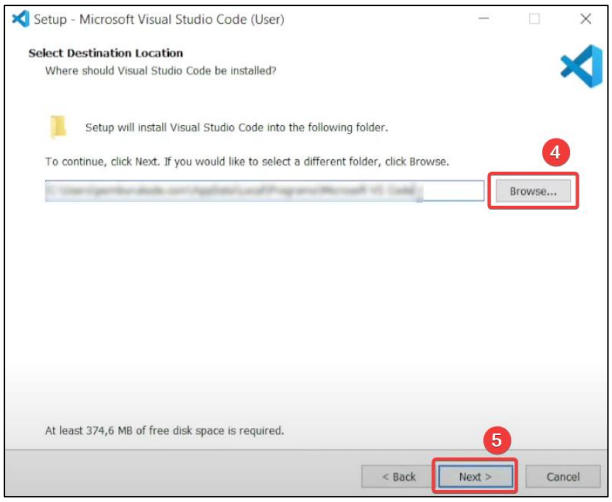
Gambar B.24 File *Visual Studio Code* yang telah di*download*

5. Tekanlah opsi **"I accept the aggrement"** dan tombol ***next*** untuk melanjutkan proses ke tahap berikutnya.



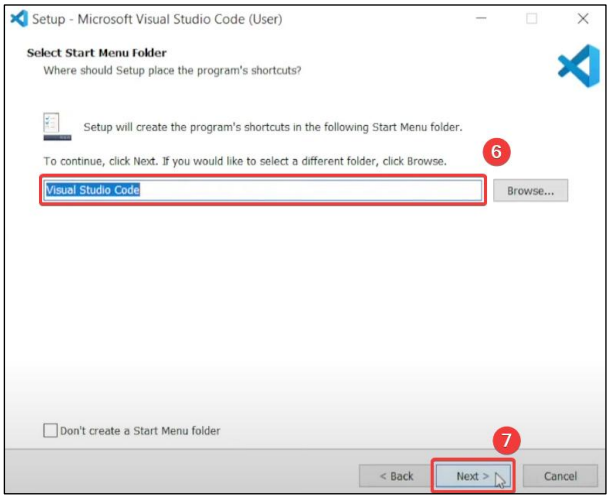
Gambar B.25 Menu *Setup Visual Studio Code*

6. Pilihlah lokasi direktori untuk menyimpan *software visual studio code* pada perangkat dan tekanlah tombol ***next*** untuk melanjutkan ketahap berikutnya.



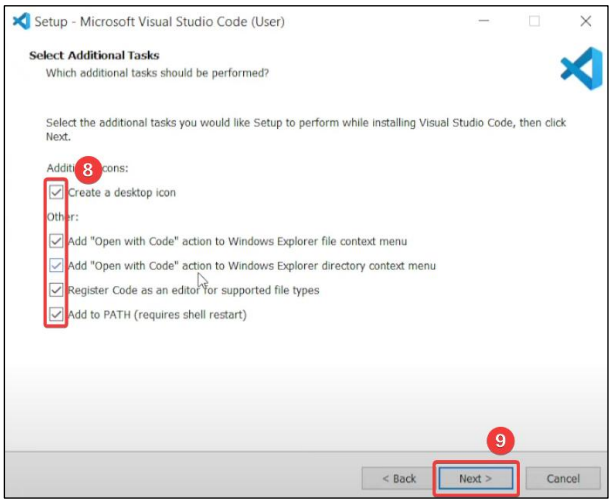
Gambar B.26 Menu *Configuration Location Visual Studio Code*

7. Pada menu **"Select Start Menu Folder"** pastikan nama pada *field* adalah "Visual Studio Code" kemudian tekanlah tombol ***next***.



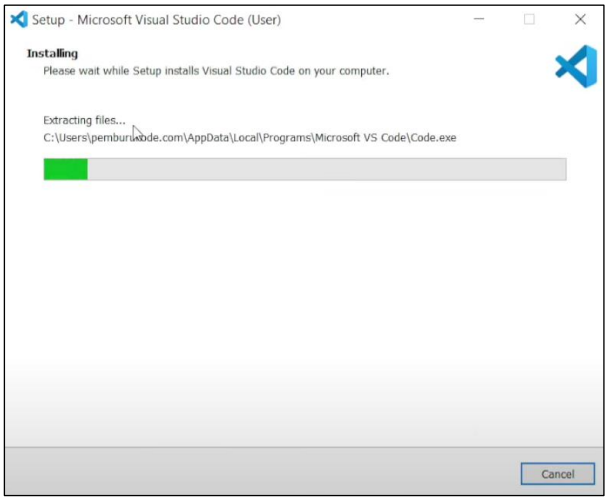
Gambar B.27 Menu untuk Membuat *Shortcut* Aplikasi *Visual Studio Code*

8. Centanglah seluruh opsi yang ada pada menu **"Select Additional Tasks"** dan tekanlah tombol ***next***.



Gambar B.28 Menu *Select Additional Task*

9. Tunggulah sampai proses instalasi *visual studio code* selesai.



Gambar B.29 Proses Instalasi *Visual Studio Code*

Setelah proses instalasi selesai, centahlah opsi **"Launch Visual Studio Code"** dan tekanlah tombol ***finish*** untuk langsung membuka aplikasi *visual studio code*.