BAB 2
LANDASAN TEORI

2.1. Aplikasi Mobile

Aplikasi merupakan suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Aplikasi dibuat untuk memudahkan pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan, dan penambahan data yang dibutuhkan [3]. Aplikasi *mobile* adalah aplikasi yang telah dirancang khusus untuk platform *mobile* (misalnya iOS, android, atau windows phone). Dalam banyak kasus, aplikasi *mobile* memiliki *user interface* dengan mekanisme interaksi unik yang disediakan oleh platform *mobile*, interoperabilitas dengan sumber daya berbasis web yang menyediakan akses ke beragam informasi yang relevan dengan aplikasi, dan kemampuan pemrosesan lokal untuk pengumpulan, analisis, dan format informasi dengan cara yang paling cocok untuk platform *mobile*. Selain itu aplikasi *mobile* menyediakan kemampuan penyimpanan persisten dalam platform. [4]

2.2. Chatting

Chatting adalah percakapan dua orang atau lebih, secara *real-time* melalui jaringan internet. Layanan-layanan untuk *chatting* di internet antara lain; Yahoo Messanger, Mirc, Pidgin, Skype, MSN Messenger, Windows Live Messenger, Trilian, Camfrog, ICQ dan sebagainya. Dengan adanya layanan *Chat*, memungkinkan kita untuk dapat halaman *login* sampai *login* berhasil dilakukan. Setelah itu *user* akan dapat mengakses halaman utama dari aplikasi *chatting* tersebut. Jika ingin mengirim pesan, maka *user* harus masuk ke halaman *chatting*. Jika sudah selesai melakukan *chatting*, maka *user* dapat kembali ke halaman utama. [5]

2.3. Website

Website adalah suatu media yang terdiri dari beberapa halaman yang saling berkaitan satu sama lain, dan berfungsi sebagai media untuk menampilkan suatu informasi, baik berbentuk gambar, video, teks, suara, ataupun gabungan dari semuanya. *Website* bersifat *multi- platform* yang artinya dapat dibuka dari segala perangkat atau *device* yang terhubung dengan jaringan internet. Walaupun teknologi ini sudah cukup lama digunakan, namun saat ini masih banyak sekali perusahaan-perusahaan yang masih menggunakan *website* dalam menampilkan profil perusahaan *(company profile)*, menjual produk, ataupun sebagai sistem yang dapat digunakan oleh pelanggan. [6]

2.4. Android

Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan platform terbuka bagi para pengembang untuk membuat aplikasi mereka sendiri. Pada awalnya dikembangkan oleh Android Inc, sebuah perusahaan pendatang baru yang membuat perangkat lunak untuk ponsel yang kemudian dibeli oleh Google Inc. Untuk pengembangannya, dibentuklah Open Handset Alliance (OHA), konsorsium dari 34 perusahaan perangkat keras, perangkat lunak, dan telekomunikasi termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia. [7]

2.5. Kotlin

Kotlin adalah bahasa pemrograman berbasis *Java Virtual Machine (JVM)*. *Kotlin* merupakan bahasa pemrograman yang pragmatis untuk android yang mengkombinasikan *object oriented (OO)* dan bahasa fungsional. *Kotlin* juga bahasa pemrograman yang interoperabilitas yang membuat bahasa ini dapat digabungkan dalam satu *project* dengan bahasa pemrograman Java. Bahasa pemrograman ini juga dapat digunakan untuk pengembangan aplikasi berbasis desktop, web dan *backend* [8]. *Kotlin* awalnya dikembangkan oleh JetBrains, perusahaan dibalik IntelliJ IDEA. Setelah melalui banyak perkembangan, JetBrains merilis *kotlin* secara *open source* dan kini

perkembangannya semakin maju. Google mendukung penuh *kotlin* untuk pengembang aplikasi Android. [9]

2.6. Application Programming Interface (API)

Application Programming Interface (API) adalah antarmuka yang digunakan untuk mengakses aplikasi atau layanan dari sebuah program. API memungkinkan pengembang untuk memakai fungsi yang sudah ada dari aplikasi lain sehingga tidak perlu membuat ulang dari awal. Pada konteks *website*, API merupakan pemanggilan fungsi melalui *Hyper Text Transfer Protocol* (HTTP) dan mendapatkan respon berupa *Extensible Markup Language* (XML) *atau JavaScript Object Notation* (JSON). [10]

Tujuan penggunaan dari API adalah untuk saling berbagi data antar aplikasi yang berbeda, Tujuan penggunaan API lainnya yaitu untuk mempercepat proses pengembangan aplikasi dengan cara menyediakan sebuah *function* yang terpisah sehingga developer tidak perlu lagi merancang fitur yang serupa. API yang bekerja pada tingkat sistem operasi membantu aplikasi berkomunikasi dengan layer dasar dan satu sama lain mengikuti serangkaian protokol dan spesifikasi yang telah disesuaikan. [10]

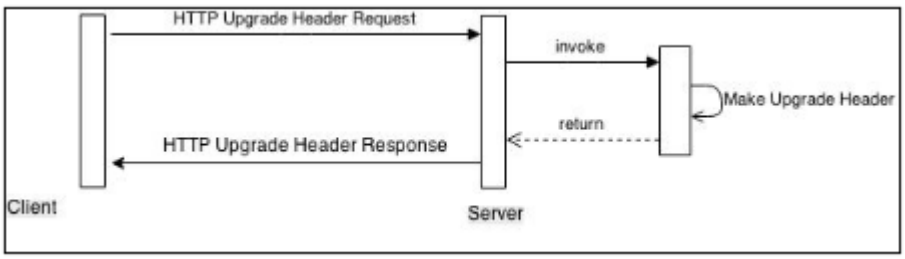
2.7. Representational State Transfer (REST)

REST merupakan gaya arsitektur dalam mendesain sebuah *web service* di mana desain REST memiliki *resource* yang dapat diakses melalui sebuah alamat HTTP URL yang *unique*. REST juga memungkinkan klien dapat melakukan *request* melalui protokol HTTP dengan mudah menggunakan URI. Masing-masing alamat URL mengacu kepada kumpulan program yang akan dieksekusi dan akan mengembalikan pesan kepada pengirim perintah. REST mengirimkan perintah yang akan dikerjakan oleh *server* menggunakan metode-metode HTTP *request method* yang disebut *verb*. Terdapat 8 (delapan) HTTP *request method*, yaitu: *GET, POST, PUT, DELETE, OPTIONS, HEAD, TRACE,* dan *CONNECT.* Dalam penggunaan REST API hanya menggunakan 4 (empat) dari metode-metode tersebut, yaitu: *GET, POST, PUT* dan *DELETE.* [11]

2.8. WebSocket

WebSocket adalah protokol komunikasi komputer yang menyediakan kanal komunikasi *full-duplex* melalui satu jalur koneksi TCP [12]. Protokol Websocket telah distandarkan oleh IETF pada RFC 6455 di 2011. WebSocket berbeda dari HTTP. Kedua protokol terletak di *layer* 7 dalam model OSI dan bergantung pada TCP pada *layer* 4. Meskipun berbeda, RFC 6455 menyatakan bahwa WebSocket "dirancang untuk bekerja melalui *port* HTTP 443 dan 80 serta untuk mendukung proksi dan perantara HTTP", sehingga membuatnya kompatibel dengan HTTP. Untuk mencapai kompatibilitas, jabat tangan WebSocket menggunakan *header* HTTP *Upgrade* untuk mengubah dari protokol HTTP ke protokol WebSocket. [13]

WebSocket berkomunikasi dengan *layer* TCP. Koneksi dibuat melalui HTTP melalui mekanisme *handshake* antara klien dan server. Setelah melakukan *handshake*, koneksi di-*upgrade* menjadi TCP [14]. Berikut adalah diagram alur cara kerja WebSocket:



Gambar 2.1 Cara Kerja WebSocket [14]

Berikut adalah penjelasan untuk setiap langkah dari diagram alur di atas:

1. Pertama, klien memulai sebuah panggilan HTTP dengan menambahkan *header* khusus untuk menandakan bahwa *request* adalah suatu panggilan WebSocket.

```
GET /socket HTTP/1.1
Host: example.org
Connection: Upgrade
Upgrade: websocket
```

2. Setelah server mengecek kebenaran *request* tersebut, sebuah *response* dikirim kembali untuk klien.

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
```

3. Selama koneksi masih terbuka, klien dan server dapat mengirim data satu sama lain.
4. Pertukaran data menggunakan *frame* yang dapat berupa teks atau *binary* melalui protokol TCP. [14]

2.9. JavaScript Object Notation (JSON)

JSON adalah sebuah format untuk berbagi data. Sesuai dengan namanya, JSON diturunkan dari bahasa pemrograman Javascript, akan tetapi format ini tersedia bagi banyak bahasa lain termasuk Python, Ruby, PHP, dan Java. JSON biasanya dilafalkan seperti nama "Jason." JSON menggunakan ekstensi .json saat ia berdiri sendiri. Saat didefinisikan di dalam format *file* lain (seperti di dalam .html), ia dapat tampil di dalam tanda petik sebagai JSON *string*, atau ia dapat dimasukkan ke dalam sebuah variabel. Format ini sangat mudah untuk ditransfer antar server web dengan *client* atau *browser*. [10]

2.10. Database

Database adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. *Database* adalah representasi kumpulan fakta yang saling berhubungan disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan. *Database* merupakan sekumpulan informasi yang saling berkaitan pada suatu subjek tertentu pada tujuan tertentu pula. *Database* adalah susunan *record* data operasional lengkap dari suatu organisasi atau perusahaan, yang diorganisir dan disimpan secara terintegrasi dengan menggunakan metode tertentu dalam

komputer sehingga mampu memenuhi informasi yang optimal yang dibutuhkan oleh para pengguna. [15]

2.11. SQLite

SQLite adalah suatu *library* yang menerapkan mesin *database self-contained, serverless, zero-configuration,* dan *transactional*. *Self-contained* berati SQLite membutuhkan sedikit sekali dukungan dari *library* eksternal atau dari sistem operasi. *Serverless* berati SQLite dalam mengakses *database* baik itu *read* atau *write* dapat secara langsung dari *file database* tanpa melalui proses server dan tidak mendukung pengaksesan secara *remote* (artinya *database* SQLite bisa dikendalikan dari jarak jauh dengan adanya jaringan komputer *("Computer Network")*, baik melalui jaringan lokal (intranet) atau internet), di mana kebanyakan mesin SQL *database* diterapkan sebagai proses server yang terpisah. *Zero-configuration* menunjukkan SQLite tidak membutuhkan instalasi sebelum penggunaannya. *Transactional* SQLite merupakan suatu transaksional *database* di mana dalam melakukan perubahan proses *query menerapkan Atomic, Consistent, Isoalated, and Durable (ACID).* [16]

2.12. Shopee Chat API

Shopee API adalah layanan *Application Programming Interface (API)* yang disediakan oleh platform *e-commerce* Shopee untuk para *developer* dalam mempermudah para penjual Shopee untuk meningkatkan penjualannya dengan memanfaatkan *tools* dan layanan untuk kebutuhan seperti mengelola produk, menganalisis pendapatan, dan mengelola percakapan atau *chat*. Shopee Chat API adalah bagian dari Shopee API yang berfokus untuk mengelola percakapan atau *chat* pada akun toko penjual Shopee. Dalam upaya untuk mendapatkan pesan secara *real-time*, Shopee Chat API menggunakan teknologi WebSocket dan REST untuk keperluan umum lainnya. [17]

Berikut adalah daftar *endpoint* dari Shopee API yang digunakan pada penelitian ini:

1. *Endpoint Login*

Tabel 2.1 *Endpoint Login*

Path	https://seller.shopee.co.id/webchat/api/coreapi/v1.2/mini/login		
Penggunaan	Untuk melakukan login akun toko Shopee.		
Protocol	HTTP (REST)		
Method	POST		
Headers	**Key**	**Tipe Data**	**Keterangan**
	Cookie	String	Data *login* yang didapatkan dari *WebView*.

Berikut adalah HTTP *request* untuk *endpoint login:*

```
POST
https://seller.shopee.co.id/webchat/api/coreapi/v1.2/mini/login
HTTP/1.1
Host: seller.shopee.co.id
Cookie: <cookie>
```

2. *Endpoint* untuk mendapatkan daftar percakapan

Tabel 2.2 *Endpoint* untuk mendapatkan daftar percakapan

Path	https://seller.shopee.co.id/webchat/api/coreapi/v1.2/mini /conversations		
Penggunaan	Untuk mendapatkan daftar percakapan.		
Protocol	HTTP (REST)		
Method	GET		
Headers	**Key**	**Tipe Data**	**Keterangan**

	Authorization	String	*Access token* yang didapatkan dari *endpoint login.*

Berikut adalah HTTP *request* untuk *endpoint* mendapatkan daftar percakapan:

```
GET
https://seller.shopee.co.id/webchat/api/coreapi/v1.2/mini/conversations HTTP/1.1
Host: seller.shopee.co.id
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5XXXX
```

3. *Endpoint* untuk mendapatkan daftar pesan

Tabel 2.3 *Endpoint* untuk mendapatkan daftar pesan

Path	https://seller.shopee.co.id/webchat/api/v1.2/mini /conversations/{conversation_id}/messages		
Penggunaan	Untuk mendapatkan daftar pesan dari suatu percakapan.		
Protocol	HTTP (REST)		
Method	GET		
Headers	**Key**	**Tipe Data**	**Keterangan**
	Authorization	String	*Access token* yang didapatkan dari *endpoint login.*
Parameters	**Key**	**Tipe Data**	**Keterangan**
(Path)	conversation_id	Long	ID dari suatu percakapan.

Berikut adalah HTTP *request* untuk *endpoint* mendapatkan daftar pesan:

```
GET
https://seller.shopee.co.id/webchat/api/v1.2/mini/conversations/{conversation_id}/messages HTTP/1.1
Host: seller.shopee.co.id
```

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5XXXX
```

4. *Endpoint* untuk membalas pesan

Tabel 2.4 *Endpoint* untuk membalas pesan

Path	https://seller.shopee.co.id/webchat/api/v1.2/messages		
Penggunaan	Untuk membalas pesan dari suatu percakapan.		
Protocol	HTTP (REST)		
Method	POST		
Headers	**Key**	**Tipe Data**	**Keterangan**
	Authorization	String	*Access token* yang didapatkan dari *endpoint login.*
	Content-type	String	application/json
Body	ChatData::class		

Berikut adalah HTTP *request* untuk *endpoint* membalas pesan:

```
POST https://seller.shopee.co.id/webchat/api/v1.2/messages
HTTP/1.1
Host: seller.shopee.co.id
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5XXXX
Content-type: application/json

{
  "request_id": "<uuid>",
  "to_id": "<destination>",
  "type": "text",
  "content": {
    "text": "<content>"
  },
  "shop_id": "<shop_id>",
  "chat_send_option": {
    "force_send_cancel_order_warning": false,
    "comply_cancel_order_warning": false
  },
```

```
  "device_id": "<uuid>",
  "conversation_id": "<conversation_id>",
  "re_policy": {
    "dfp_access_f": ""
  }
}
```

5. *Endpoint* untuk websocket

Tabel 2.5 *Endpoint* untuk websocket

Path	wss://seller-push-ws.shopee.co.id/socket.io		
Penggunaan	Untuk mendapatkan notifikasi pesan baru secara real-time.		
Protocol	WebSocket		
Method	GET		
Headers	**Key**	**Tipe Data**	**Keterangan**
	Connection	String	Upgrade
	Upgrade	String	websocket
Body	Payload::class		

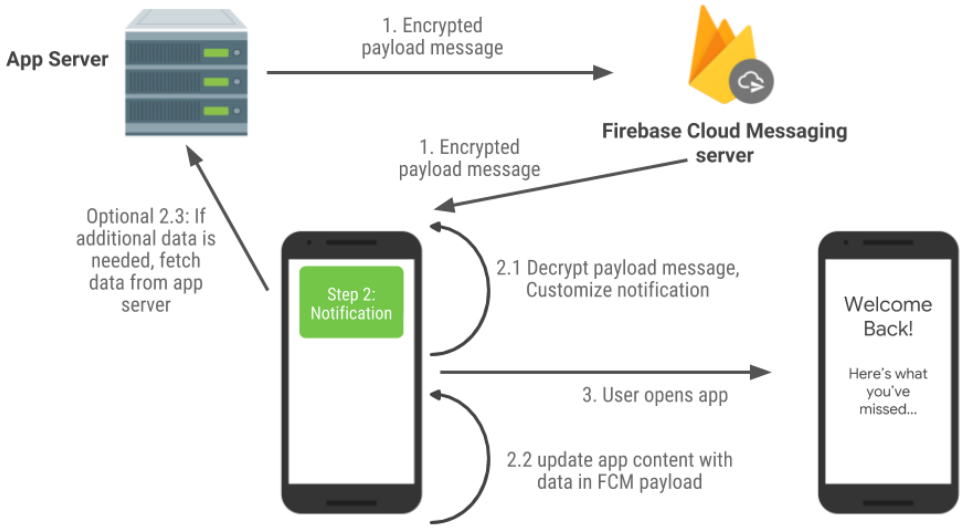
Berikut adalah HTTP *request* untuk *endpoint* websocket:

```
GET https://seller-push-ws.shopee.co.id/socket.io HTTP/1.1
Host: seller-push-ws.shopee.co.id
Connection: Upgrade
Upgrade: websocket
```

2.13. Firebase Cloud Messaging (FCM)

Firebase Cloud Messaging (FCM) merupakan layanan *cross-platform* untuk berkirim pesan yang disediakan oleh Google secara gratis. FCM juga menyediakan fungsi untuk melakukan *push notification*, yaitu notifikasi yang muncul di bagian atas layar *smartphone* dan dapat diseret ke bawah, untuk mengakses pesan lengkapnya pengguna cukup menekan pesan yang tampil pada notifikasinya.

Penggunaan fitur *push notification* dengan FCM sangat membantu karena FCM akan mengirimkan notifikasi secara *realtime*. [18] Berikut adalah cara kerja dari Firebase Cloud Messaging (FCM):



Gambar 2.2 Cara Kerja Firebase Cloud Messaging (FCM) [19]

Berikut adalah penjelasan dari diagram di atas:

1. Pesan yang terenskripsi dari *app* server akan dikirimkan ke *Firebase Cloud Messaging Server* melalui Firebase SDK atau REST API. Lalu Firebase mengirimkan pesan kepada aplikasi tujuan.
2. Pesan yang berhasil diterima oleh aplikasi untuk di-*decrypt* lalu ditampilkan.
   2.1. Pesan dapat terlebih dahulu dikostumisasi sebelum ditampilkan.
   2.2. *Update* konten aplikasi dengan data yang berada di dalam *payload* FCM.
   2.3. Opsional, jika data tambahan diperlukan, ambil data dari *app server*.
3. *User* membuka aplikasi melalui notifikasi yang telah dikirimkan sebelumnya.

2.14. Android Studio

Android Studio adalah *Integrated Development Environment (IDE)* resmi untuk pengembangan aplikasi Android yang dikembangkan oleh Google Inc. Berbasis editor kode dan alat developer yang andal dari IntelliJ IDEA, Android

Studio menawarkan lebih banyak fitur yang mampu meningkatkan produktivitas Anda saat mem-*build* aplikasi Android, seperti:

1. Sistem *build* berbasis Gradle yang fleksibel.
2. Emulator yang cepat dan kaya fitur.
3. Lingkungan terpadu tempat Anda bisa mengembangkan aplikasi untuk semua perangkat Android.
4. Terapkan Perubahan untuk melakukan *push* pada perubahan kode dan *resource* ke aplikasi yang sedang berjalan tanpa memulai ulang aplikasi
5. *Template* kode dan integrasi GitHub untuk membantu Anda membuat fitur aplikasi umum dan mengimpor kode sampel.
6. *Framework* dan alat pengujian yang lengkap.
7. Alat *lint* untuk merekam performa, kegunaan, kompatibilitas versi, dan masalah lainnya.
8. Dukungan C++ dan NDK.
9. Dukungan bawaan untuk Google Cloud Platform, yang memudahkan integrasi Google Cloud Messaging dan App Engine. [20]

2.15. IntelliJ IDEA

IntelliJ IDEA adalah *Integrated Development Environment (IDE)* pilihan utama untuk *development* Java dan Kotlin yang dikembangkan oleh JetBrains r.s.o. IntelliJ IDEA mempunyai fitur-fitur yang memudahkan proses *development* seperti: *intelligent coding assistance*, *refactoring* yang dapat diandalkan, navigasi kode instan, *built-in developer tools*, dukungan *development* web dan perusahaan, dan lain-lain. [21]

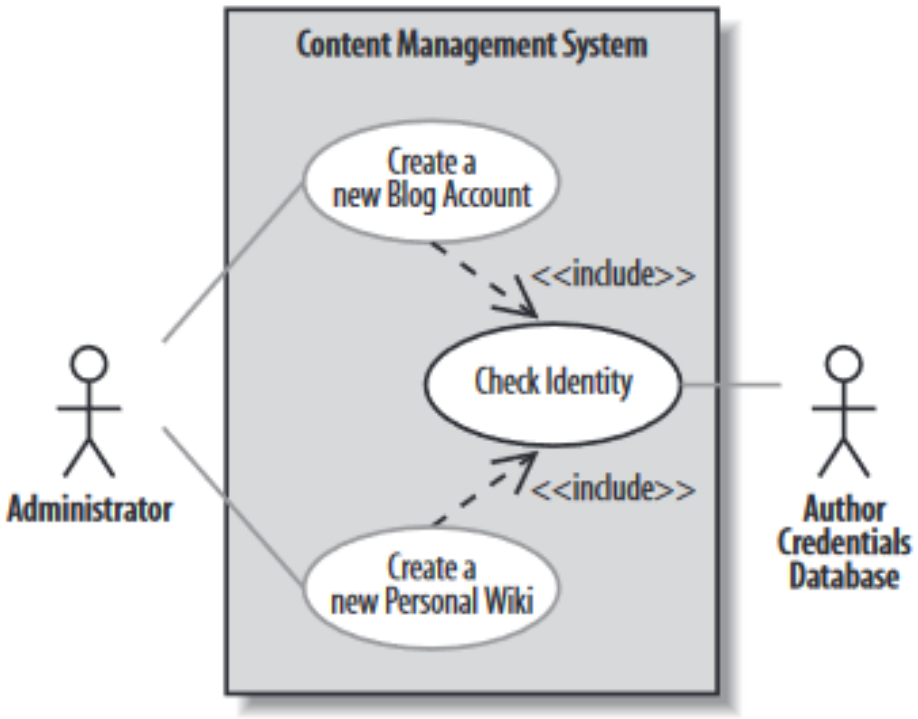
2.16. Unified Model Language (UML)

UML (Unifed Modeling Language) adalah salah satu standar Bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. Ada beberapa diagram yang digunakan proses pembuatan perangkat lunak berorientasi objek di antaranya, *use case diagram, activity diagram, class diagram* dan *sequence diagram.* [22]

2.16.1. Use Case Diagram

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut *actor* dan *use case*.

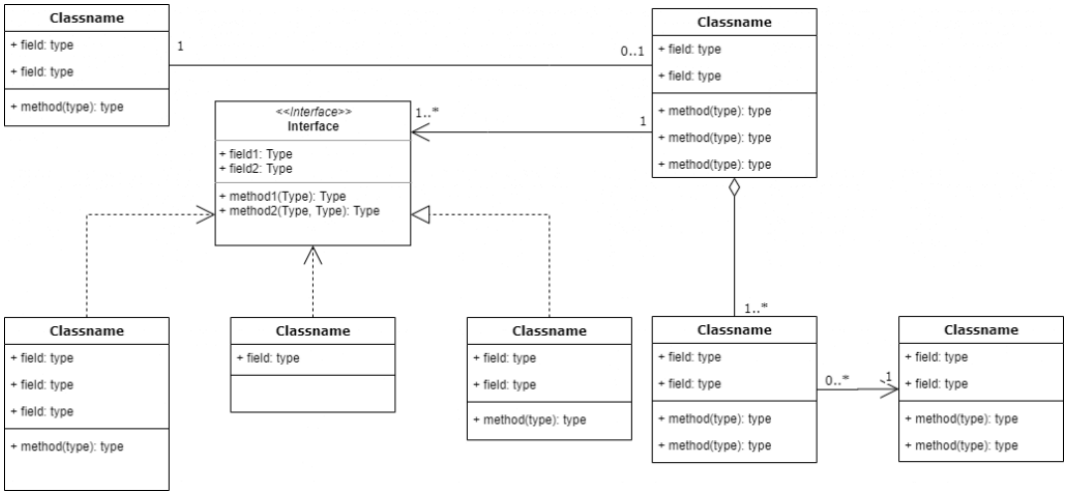
1. *Actor* merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibangun.
2. *Use Case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. [22]



Gambar 2.3 Contoh *Use Case Diagram* [23]

2.16.2. Class Diagram

Class diagram atau diagram kelas atau menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. [22]



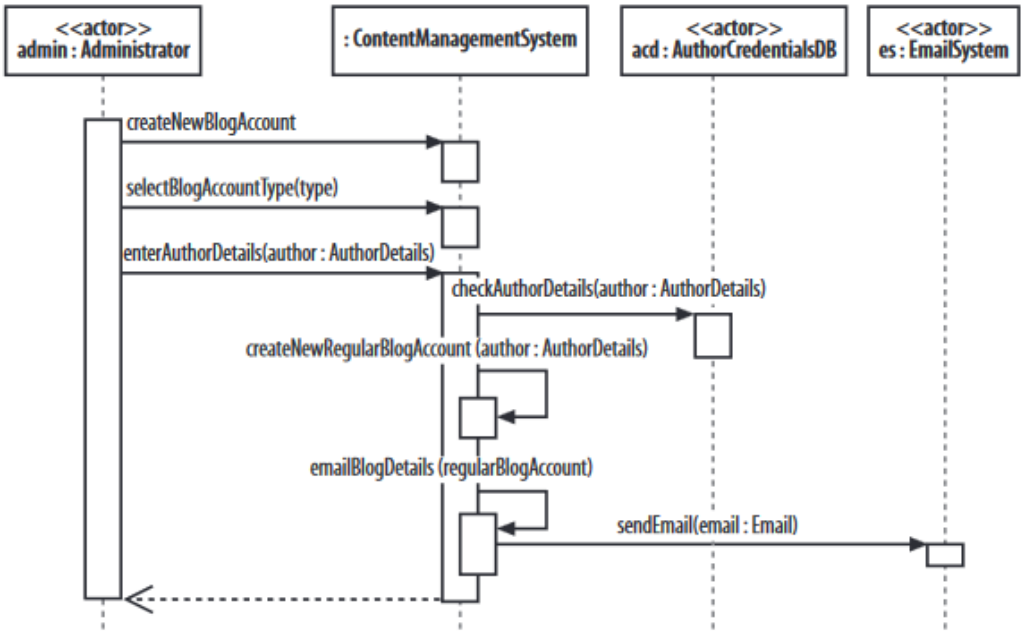
Gambar 2.4 Contoh *Class Diagram* [23]

2.16.3. Sequence Diagram

Sequence diagram atau diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. [22]

Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram *sekuen* yang harus dibuat juga semakin banyak. Penomoran pesan berdasarkan urutan

interaksi pesan. Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu. [22]

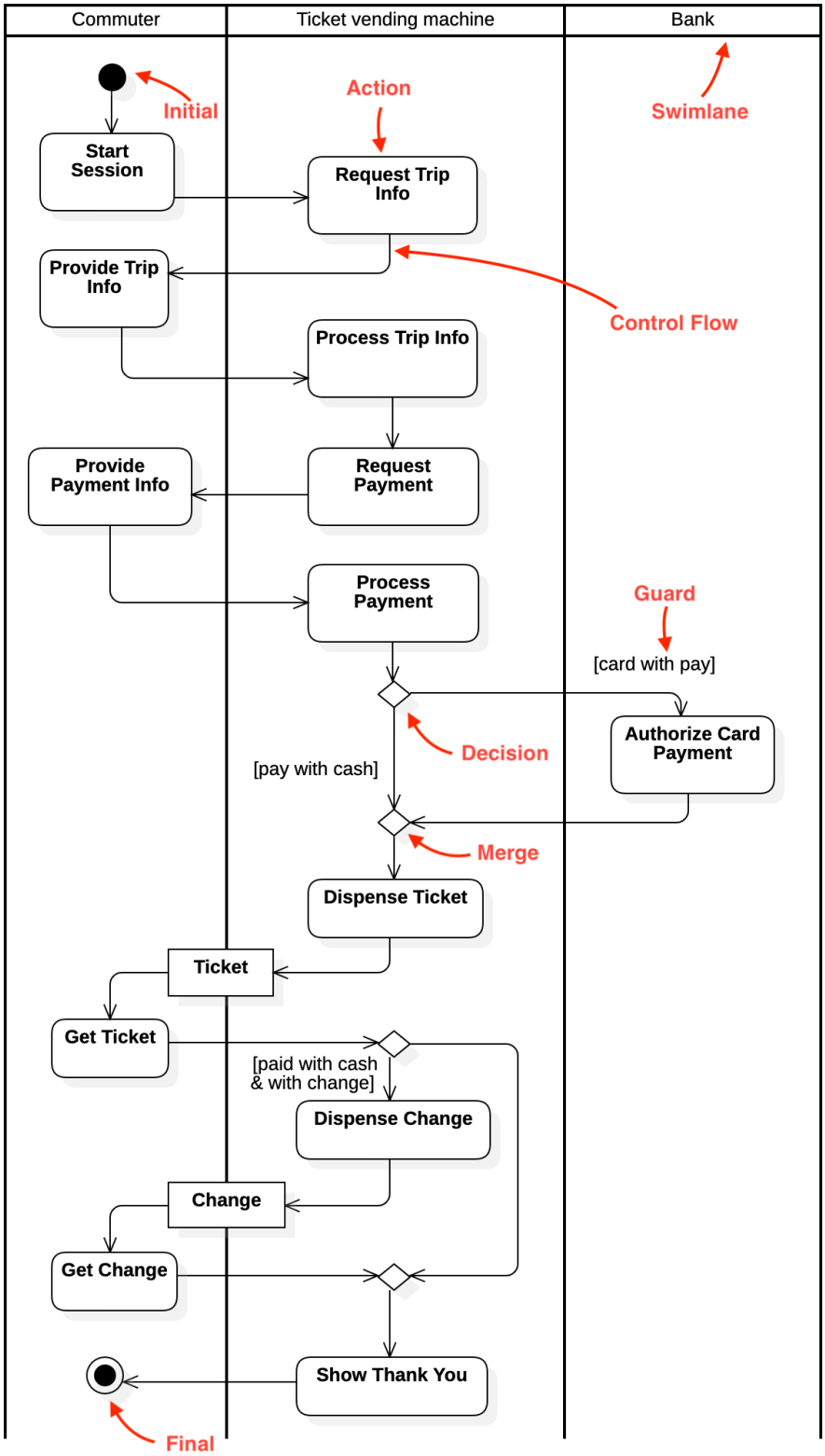


Gambar 2.5 Contoh *Sequence Diagram* [23]

2.16.4. Activity Diagram

Activity diagram atau diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

1. Rancangan proses bisnis di mana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem/*user interface* di mana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian di mana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak. [22]



Gambar 2.6 Contoh *Activity Diagram* [23]

2.17. Skala Likert

Skala likert digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Dalam penelitian, fenomena sosial ini telah ditetapkan secara spesifik oleh peneliti, yang selanjutnya disebut sebagai variabel penelitian. Dengan skala Likert, maka variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyusun item-item instrumen yang dapat berupa pernyataan atau pertanyaan. Jawaban setiap item instrumen yang menggunakan skala Likert mempunyai gradasi dari sangat positif sampai sangat negatif, yang dapat berupa kata-kata antara lain: [24]

a. Sangat setuju
b. Setuju
c. Ragu-ragu
d. Tidak setuju
e. Sangat tidak setuju

a. Selalu
b. Sering
c. Kadang-kadang
d. Tidak pernah

a. Sangat positif
b. Positif
c. Negatif
d. Sangat negatif

a. Sangat baik
b. Baik
c. Tidak baik
d. Sangat tidak baik

Untuk keperluan analisis kuantitatif, maka jawaban itu dapat diberi skor, misalnya:

1. Setuju/selalu/sangat positif diberi skor 5.
2. Setuju/sering/positif diberi skor 4.
3. Ragu-ragu/kadang-kadang/netral diberi skor 3.
4. Tidak setuju/hampir tidak pernah/negatif diberi skor 2.
5. Sangat tidak setuju/tidak pernah diberi skor 1. [24]