

## BAB III. LANDASAN TEORI

### 3.1. *E-Learning*

Perkembangan teknologi digital saat ini mulai berdampak pada bidang pendidikan. Salah satunya dengan munculnya metode atau konsep pembelajaran *e-learning* atau pembelajaran elektronik dengan sistem *online*. *E-learning* merupakan sebuah sistem atau konsep pembelajaran yang memanfaatkan teknologi informasi dalam proses dalam belajar antara peserta didik dan pengajar [10]. Pembelajaran secara *e-learning* memungkinkan proses belajar mengajar bisa dilakukan oleh siapapun, dimanapun, dan kapanpun. Penerapan *e-learning* saat ini sudah semakin populer dan mulai diterapkan di beberapa institusi dan perusahaan.

Dalam penerapannya, ada beberapa hal yang menjadi karakteristik utama dari *e-learning*. Pertama yaitu dari pengertian dari *e-learning* sendiri yang berarti pembelajaran elektronik secara *online*, maka bisa diartikan metode ini memanfaatkan peran jasa teknologi dan digital. Kedua, yaitu bahan ajar dari *e-learning* yang bersifat mandiri dalam bentuk digital. Bahan tersebut disimpan dan dikelola dalam sebuah sistem komputasi. Hal ini memungkinkan pengajar dan peserta pelatihan bisa mengakses bahan ajar kapan saja dan dimana saja. Karakteristik *e-learning* juga dapat membuat kurikulum dan sistem administrasi pendidikan yang bisa diakses melalui jaringan komputer setiap saat.

Penerapan *e-learning* memiliki manfaat yang dapat dirasakan oleh semua pihak. Diantaranya adalah menunjang proses pembelajaran bagi para peserta pelatihan dalam mendapatkan materi yang optimal dan bagi pengelola pembelajaran dapat memantau proses dan perkembangan pembelajaran peserta dengan mudah dan cepat. Kemudian *e-learning* memberikan kemudahan dalam hal akses. Peserta dan pengajar dapat melakukan interaksi secara intens kapan saja dan dimana saja. Dengan adanya *e-learning*, maka peserta dapat mengatur waktu belajar mereka sendiri dan dapat mempelajari kembali materi pembelajaran yang belum dipahami dengan baik [11].

### 3.2. Low Code Platform

Pada saat ini telah tersedia banyak *platform* pengembangan dan bahasa pemrograman yang digunakan perusahaan atau sebuah tim dalam mengerjakan proyek pengembangan perangkat lunak. Pengembangan perangkat lunak dapat dibagi berdasarkan pada *platform* atau tempat untuk menjalankan perangkat lunak. Pada perangkat android, *platform* pengembangan yang digunakan yaitu Android Studio dengan bahasa pemrograman Java. Pengembangan pada perangkat iOS dapat menggunakan Xcode sebagai *platform* pengembangan dengan bahasa pemrograman Swift. Untuk mengembangkan sebuah *website*, *platform* pengembangan yang bisa dipakai adalah Visual Studio Code dan dan notepad++ dengan php sebagai bahasa pemrograman, serta masih banyak terdapat *platform* pengembangan dan bahasa pemrograman lainnya.

*Low-code platform* merupakan salah satu alternatif pengembangan perangkat lunak. *Low-code platform* memungkinkan pengembang untuk meminimalkan pengkodean secara manual dengan bahasa pemrograman tertentu. Penggunaan fitur *drag and drop* merupakan salah satu fitur yang penting dan dapat membuat proses pengembangan menjadi lebih cepat. Pengembang dapat menggunakan *low-code platform* untuk membangun aplikasi dengan tampilan visual aplikasi secara langsung dan dapat diintegrasikan dengan database [12]. Saat ini tersedia beberapa *low-code platform* yang digunakan di industri pengembangan perangkat lunak, contohnya yaitu Outsystems dan Mendrix. Mendrix merupakan *low-code platform* yang memungkinkan untuk membangun dan mengembangkan aplikasi mobile dan web. Mendrix menawarkan *no-code* dan *low-code tools* dalam satu *platform* yang terintegrasi, serta dirancang dapat mempercepat pengiriman aplikasi di seluruh siklus pengembangan dari ide awal hingga ke tahap pengembangan [13].

### 3.3. Outsystems

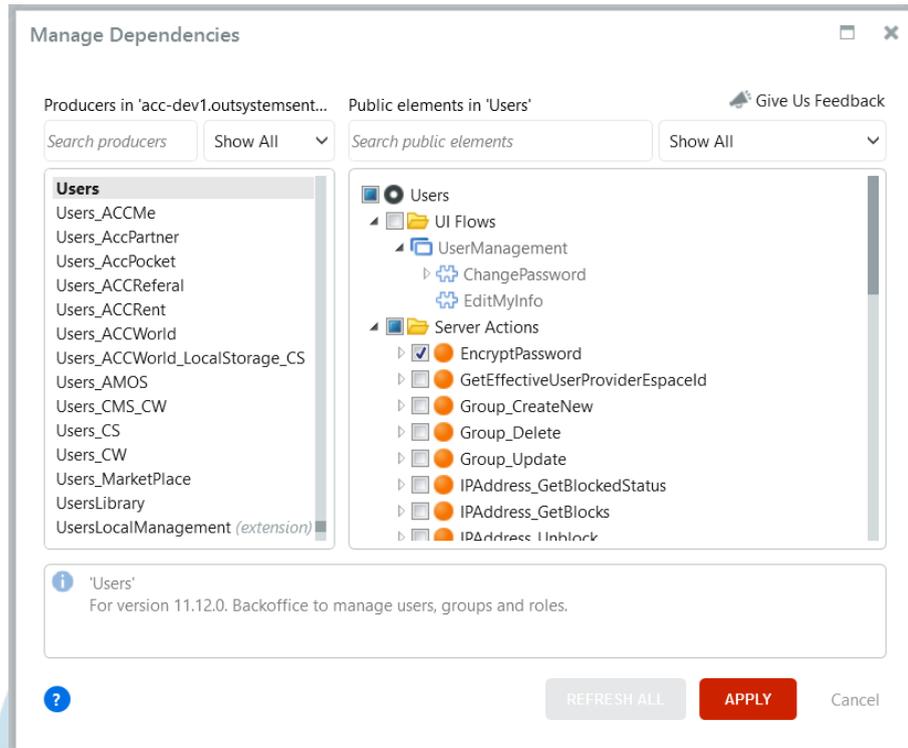
Outsystems merupakan *platform* aplikasi *low-code* yang dapat digunakan untuk membangun aplikasi *mobile* dan web. Outsystems diluncurkan oleh sebuah tim yang dipimpin oleh CEO Outsystems saat ini yaitu Paulo Rosado di Lisbon, Portugal pada tahun 2001. Kantor utama Outsystems saat ini berada di kota Boston,

Amerika Serikat. Outsystems didedikasikan untuk menjawab tantangan transformasi digital dan siklus pengiriman yang lebih cepat [14]. Outsystems memiliki visi untuk mengubah perangkat lunak perusahaan. Saat ini, outsystems telah diakui sebagai dalam ruang *platform* aplikasi modern dengan memiliki 435.000 anggota komunitas, 350 mitra dan ribuan pelanggan aktif di 87 negara dan 22 industri [15].

Outsystems memiliki sebuah aplikasi desktop yang bernama Service Studio. Service Studio merupakan *environment* pengembangan yang berfungsi untuk tempat mengembangkan aplikasi di *server*, menentukan model data, membuat antarmuka pengguna untuk web tradisional, web reaktif, dan aplikasi *mobile*, serta melakukan *debug* aplikasi [16]. *Environment* di dalam Outsystems juga memungkinkan pengembang untuk menguji fungsionalitas dari aplikasi yang dibangun sebelum dapat dirilis atau melanjutkan ke tahap produksi. *Environment* bisa dibagi menjadi tiga jenis menurut fungsi masing-masing. Pertama adalah *Development Environment*, berfungsi untuk membangun dan mengembangkan aplikasi. Kedua yaitu *Testing Environment*, digunakan untuk proses pengujian fungsionalitas dari aplikasi yang telah dikembangkan oleh tim QC (*Quality Control*). Pengujian dilakukan untuk menemukan *error* dan *bug* dari aplikasi, apabila *error* dan *bug* ditemukan maka aplikasi akan diperbaiki oleh tim pengembang sampai aplikasi siap untuk masuk ke tahap selanjutnya. Ketiga adalah *Production Environment*, digunakan untuk merilis aplikasi yang telah melalui proses pengujian oleh tim QC sehingga aplikasi sudah siap digunakan dan dimanfaatkan oleh pengguna.

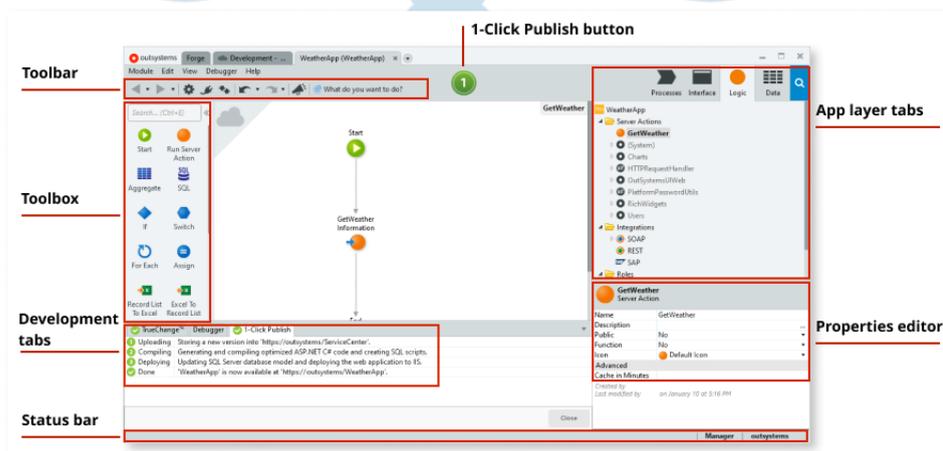
Untuk melakukan akses ke *environment* di Service Studio, seorang pengembang harus login terlebih dahulu untuk masuk ke *environment* yang mereka miliki. Setelah login, pengembang dapat melihat semua daftar aplikasi yang sudah dibangun dan dikembangkan sebelumnya.





Gambar 3.2 Tampilan pengaturan dependensi antar modul

Outsystems menyediakan *workspace* yang digunakan oleh pengembang untuk mendesain antarmuka, membuat fungsi, dan melakukan *debug aplikasi*. Pada gambar 3.3 merupakan gambar *workspace* terbagi menjadi delapan bagian yang memiliki kegunaan dan fungsi masing-masing, delapan bagian tersebut antara lain yaitu [16]:

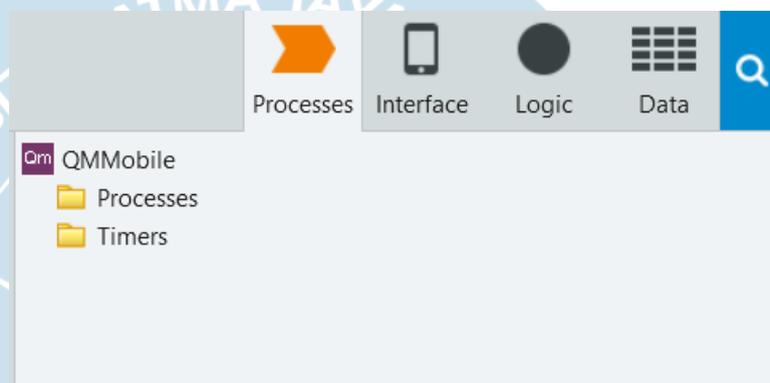


Gambar 3.3 Tampilan *Workspace* Outsystems Service Studio

- 1) *Main Editor*: sebagai tempat untuk mendesain antarmuka dan membuat logic aplikasi.
- 2) *Toolbar*: merupakan gabungan *tools* berupa *shortcut* dengan fungsi tertentu.
- 3) *Toolbox*: terdiri dari dua tipe yang berbeda, *Toolbox UI* berisi *widgets* yang berfungsi untuk mendesain antarmuka aplikasi dan *Toolbox Logic* berisi *logic* yang digunakan dalam membuat fungsi.
- 4) *Development Tabs*: menampilkan beberapa informasi yang dibutuhkan saat proses pengembangan dan terbagi dalam empat bagian antara lain:
  - a) *TrueChange tab*: menampilkan *error* dan *warnings* yang ada pada modul. Untuk mencarinya dapat dengan cara mengklik dua kali *error* dan *warnings* yang ingin dicari.
  - b) *Debugger tab*: berfungsi untuk melakukan debug aplikasi. *Debug* aplikasi dilakukan untuk melihat alur aplikasi langkah demi langkah dan mendeteksi protensial *error* yang terjadi.
  - c) *1-Click Publish tab*: menampilkan proses kemajuan dan hasil dari proses *deployment*.
  - d) *Search Result tab*: tab ini akan muncul saat pengembang melakukan pencarian sebuah konten di dalam modul. Untuk mencari letak konten tersebut dapat dengan mengklik dua kali konten yang ingin dicari.
- 5) *Status Bar*: menampilkan informasi tentang pengguna dan *environment* yang saat ini digunakan, serta kapan terakhir kali aplikasi dipublikasikan.
- 6) *Properties Editor*: menampilkan properti dari element yang dipilih.
- 7) *1-Click Publish Button*: berfungsi untuk melakukan *deployment* modul ke *environment* yang digunakan saat ini oleh pengguna.
- 8) *Application Layer Tabs*: mempunyai lima bagian dengan fungsi masing-masing, bagian-bagian tersebut antara lain:
  - a) *Processes*

Tab ini mempunyai dua folder yaitu *Processes* dan *Timers*. Pada gambar 3.4 merupakan tampilan dari tab *processes* yang dan dua folder dibawahnya. Folder *processes* berisi file proses yang

memungkinkan pengembang melakukan integrasi proses bisnis ke dalam aplikasi. Sebuah proses dirancang dalam sebuah aliran proses, yang biasanya mewakili aktivitas yang harus dilakukan selama siklus hidup entitas [17]. Selanjutnya dalam folder *timers* terdapat *timers* yang digunakan untuk menjalankan *logic* yang tidak tersinkronasi dan menjalankan logika dengan waktu tertentu di aplikasi Outsystems, contohnya logika untuk mengirim email dengan waktu yang sudah ditentukan [18].

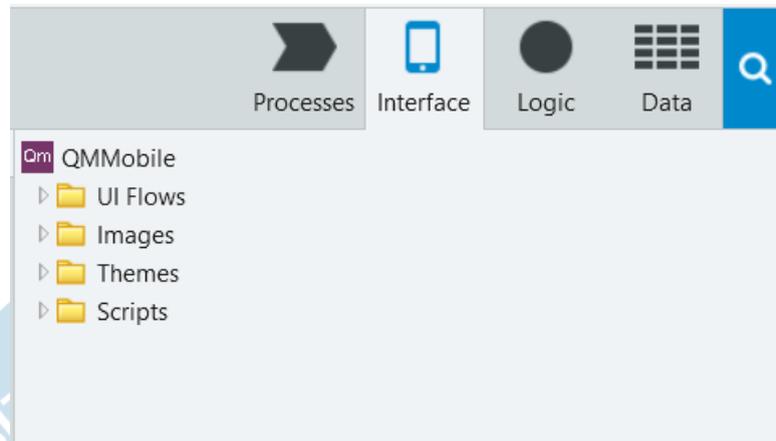


Gambar 3.4 Tampilan *Application Layer Tabs* bagian *Processes*

b) *Interface*

Tab ini berisi empat folder yaitu *UI Flows*, *Images*, *Themes*, dan *Script*. Pada gambar 3.5 merupakan tampilan dari tab *interface* yang memiliki empat folder dibawahnya. Folder pertama yaitu *UI Flows*, merupakan elemen yang terdiri dari *Screens* dan *Blocks*. Ketika pengguna membuat aplikasi baru, maka secara otomatis aplikasi memiliki beberapa *UI Flows default* seperti antarmuka untuk *login*. Kedua yaitu *Images*, berisi kumpulan gambar yang digunakan dalam aplikasi dan format gambar yang dapat digunakan adalah GIF, JPEG, dan PNG. Ketiga yaitu *Themes*, berisi detail tema visual aplikasi dan file *Cassading Style Sheets (CSS)* yang dapat digunakan untuk mendesain antarmuka aplikasi. Keempat yaitu *Scripts*, berisi Javascript yang dapat melakukan sebuah fungsi yang spesifik dan

pengembang dapat menambahkan serta memodifikasi *Script* sesuai dengan kebutuhan aplikasi.

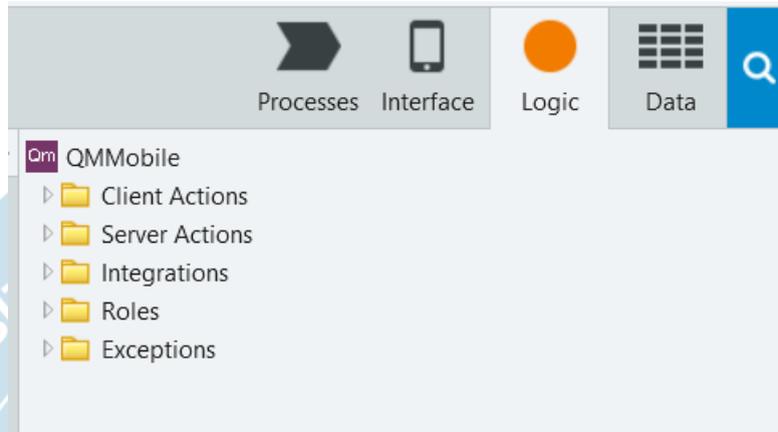


Gambar 3.5 Tampilan *Application Layer Tabs* bagian *Interface*

c) *Logic*

Tab ini berisi *Client action*, *Server Action*, *Integrations*, *Roles*, dan *Exception*. Pada gambar 3.6 merupakan tampilan dari tab *logic* yang terdapat lima folder dibawahnya. Tab *logic* berfungsi menyimpan fungsi-fungsi yang dapat mendukung jalannya sebuah aplikasi. Pertama yaitu *Client actions*, merupakan fungsi untuk menjalankan logika di sisi klien atau user. Kedua adalah *Server Action*, merupakan fungsi untuk menjalankan logika di sisi server. Pengembang dapat memanfaatkan *Server Action* sebagai fungsi yang dapat dipanggil di fungsi lain, hal ini memungkinkan pengembang dapat membuat logika yang terpusat dan jika ada perubahan logika bisa dilakukan dalam satu tempat. Ketiga yaitu *Integrations*, digunakan untuk mengintegrasikan modul Outsystems dengan *web service*. Terdapat tiga cara yang disediakan yaitu dengan SOAP (*Simple Object Access Protocol*), REST (*Representational State Transfer*), dan SAP Systems. Keempat adalah *Roles*, dibagi menjadi dua jenis yaitu *Anonymous* dan *Registered*. Roles digunakan untuk membatasi pengguna mengakses layar dan operasi.

Kelima yaitu *Exception*, digunakan untuk memunculkan pesan pemberitahuan pada layar ketika terjadi kemungkinan alur aplikasi berjalan tidak normal. Salah satu contohnya adalah pesan *invalid login* ketika user salah memasukkan *username* dan *password*.

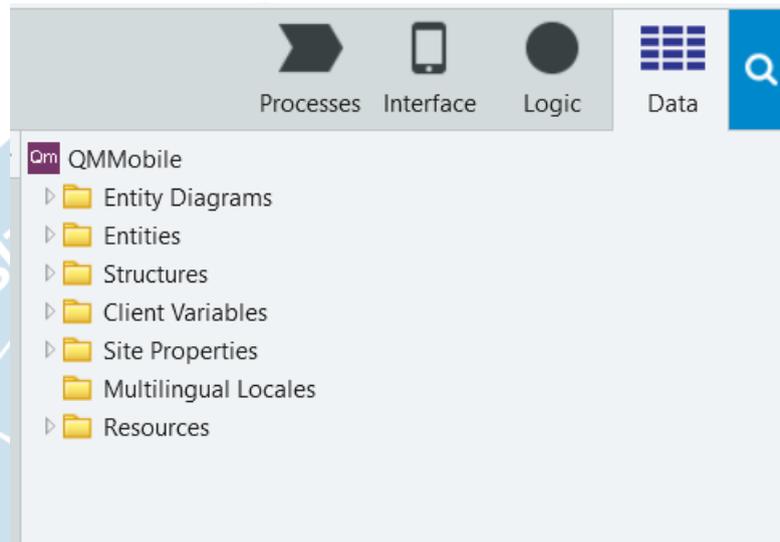


Gambar 3.6 Tampilan *Application Layer Tabs* bagian *Logic*

d) *Data*

Tab ini terdiri dari tujuh folder berbeda dengan fungsi masing-masing. Pada gambar 3.7 merupakan tampilan dari tab *data* beserta tujuh folder dibawahnya. Kegunaan dari tab data yaitu menyimpan data-data yang dibuat oleh pengembang. Pertama folder *Entity Diagram*, digunakan untuk melihat dan membuat struktur *database* beserta relasi antar tabel. Kedua folder *Entities*, berisi kumpulan tabel *database* yang digunakan dalam pengembangan aplikasi. Ketiga folder *Structures*, berisi file *Structures* yang merupakan variabel yang menyimpan variabel dengan tipe data berbeda dan dikelompokkan, serta dapat digunakan dalam *logic* dan antarmuka. Keempat yaitu folder *Client Variables*, merupakan tempat untuk menyimpan tipe data dasar pada sisi *client*, seperti menyimpan *cache* dan konfigurasi pada aplikasi. Kelima yaitu folder *Site Properties*, merupakan tempat untuk menyimpan variabel global yang memiliki nilai *constant* atau nilai yang tidak dapat diubah oleh program

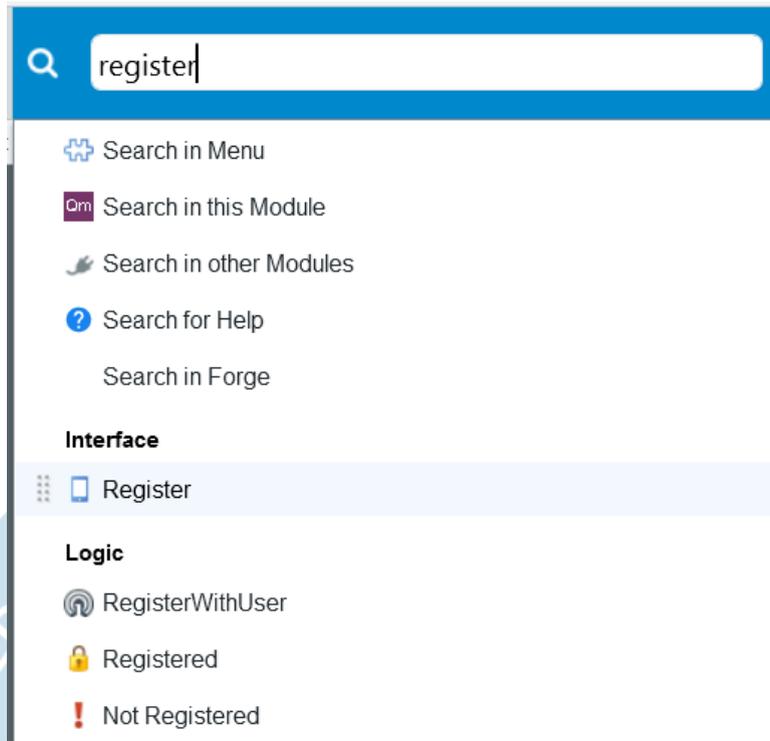
selama program dijalankan. Keenam yaitu folder *Multilingual Locales*, digunakan untuk menambahkan bahasa baru yang digunakan dalam membangun aplikasi. Ketujuh yaitu folder *Resources*, berisi beragam file yang bisa ditampilkan dalam modul seperti dokumen PDF dan file video.



Gambar 3.7 Tampilan *Application Layer Tabs* bagian *Data*

e) *Search*

*Search* merupakan fitur yang disediakan untuk pengguna dalam mencari konten dalam modul. Pengguna dapat memasukkan nama konten yang dicari melalui kolom *Search*. Setelah mendapatkan konten yang dicari, pengguna dapat mengklik konten pada list hasil pencarian untuk dapat ke lokasi konten tersebut. Pada gambar 3.8 merupakan contoh ketika pengguna melakukan pencarian dengan kata kunci *register* dan pada list pencarian muncul sebuah *screen* dengan nama *register*.



Gambar 3.8 Tampilan *Application Layer Tabs* bagian *Search*

### 3.4. Aplikasi *Mobile*

Aplikasi *mobile* atau *mobile apps* merupakan aplikasi yang dibangun untuk dapat berjalan di perangkat *mobile*, seperti *smartphone*, *tablet*, dan *smartwatch*. Aplikasi *mobile* dikembangkan di *platform mobile* dengan bahasa pemrograman tertentu dan dirancang dapat berjalan sesuai fungsi aplikasi. Proses pendistribusian aplikasi *mobile* biasanya melalui layanan yang disediakan oleh perusahaan dari *mobile operating systems*, contohnya *App Store* yang ada pada *operating systems* iOS dan *Google Play store* ada pada *operating systems* android. Aplikasi *mobile* dalam proses pengembangannya telah diintegrasikan dengan fitur yang terdapat pada perangkat *mobile*, seperti kompas, kamera, dan *GPS (Global Positioning System)*. Fitur tersebut dapat membantu aplikasi *mobile* dalam menjalankan berbagai fungsi tertentu, yaitu menunjukkan arah navigasi, melakukan pengambilan gambar dan mengetahui lokasi pengguna atau tempat tertentu.

Aplikasi *mobile* mempunyai beberapa keunggulan yang bisa dimanfaatkan oleh pengguna. Keunggulan pertama yang didapat yaitu *User Interface* dan *User*

*Experience* aplikasi *mobile* yang mudah digunakan dan menarik. Kedua, ada beberapa aplikasi yang bisa digunakan tanpa harus terkoneksi dengan internet. Ketiga, *user* atau pengguna dapat mengakses aplikasi dimana saja melalui *smartphone* yang dimilikinya. Aplikasi *mobile* secara spesifik memiliki beberapa kategori, seperti aplikasi komunikasi yang dapat digunakan untuk berkomunikasi dengan mudah, aplikasi pendidikan yang berisi konten edukasi dan materi pembelajaran, dan aplikasi sosial media yang digunakan untuk berinteraksi dan terhubung dengan pengguna sosial media lain [19]. Dengan beragam kategori dan manfaatnya, aplikasi *mobile* dapat membantu pengguna untuk lebih mudah untuk mendapatkan informasi yang diinginkan dan dapat terhubung dengan lebih banyak orang.

