BAB II

LANDASAN TEORI

2.1 Aplikasi *M-RESTO*

Menurut Dianna Effendi dan Agung Mulyanto (2013:2-1) dalam tulisannya pada PROSIDING SNASTIKOM 2013 menyatakan bahwa :

"*M-resto* adalah aplikasi yang diperuntukan bagi para pelaku bisnis di bidang kuliner dalam hal promosi restorannya dan bagi para pecinta kuliner dalam hal mengetahui informasi tentang menu dan harga makanan di suatu restoran atau *café* hingga dapat melakukan pemesanan menu makanan melalui aplikasi *m-resto*".

Pada awalnya *m-resto* dikenalkan oleh Nazzruddin Saffat H pada bukunya Pemrograman Aplikasi *Mobile Smartphone* dan Tablet PC Berbasis *Android*. Aplikasi tersebut dikostumisasi agar lebih menarik dan lebih interaktif antar pengguna sehingga aplikasi tersebut diharapkan mampu lebih berguna bagi pemakai aplikasi.

2.2 Metodelogi Penelitian

Metode penelitian merupakan prosedur dan cara melakukan pengolahan data yang diperlukan untuk menjawab atau memecahkan masalah penelitian untuk mencapai tujuan tertentu. Sugiyono (2010:2) menyatakan bahwa :

"Metode penelitian dapat diartikan sebagai cara ilmiah yang dilakukan untuk mendapatkan data yang objektif, valid, dan reliabel dengan tujuan dapat ditemukan, dibuktikan, dan dikembangkan suatu pengetahuan sehingga pada gilirannya dapat digunakan untuk memahami, memecahkan dan mengantisipasi masalah."

Dari penjelasan mengenai metode penelitian yang telah diuraikan, maka dapat disimpulkan bahwa metodelogi penelitian menggambarkan tahapan-tahapan dalam proses penelitian guna memecahkan masalah penelitian dari awal perencanaan hingga tercapainya tujuan penelitian, Tidak ada satu format yang baku tentang metodologi penelitian, tetapi setiap metodologi penelitian tidak terlepas dari kerangka metode ilmiah, pemilihan metodologi penelitian lebih tergantung pada jenis penelitian yang dilakukan, dan perkembangan ilmu pengetahuan memungkinkan munculnya metodologi penelitian yang baru (dinamis).

2.3 Metode Pendekatan dan Pengembangan Sistem

Metode pendekatan sistem merupakan salah satu cara dalam menyelesaikan masalah dengan melakukan identifikasi terhadap sejumlah kebutuhan-kebutuhan. Sedangkan pengembangan sistem dapat dilihat sebagai sebuah proses. Lebih jauh pengembangan sistem pada dasarnya adalah proses perubahan, penghalusan, transformasi atau tambahan pada produk yang sudah ada.

2.3.1 Tiga Pendekatan Dalam Pengembangan Informasi

Akan dijelaskan mengenai tiga pendekatan pada pengembangan sistem informasi yaitu pendekatan berorientasi proses, pendekatan berorientasi data, dan pengembangan berorientasi objek. Berikut penjelasan mengenai ketiga pendekatan tersebut :

2.3.1.1 Pendekatan Berorientasi Proses

Pendekatan berorientasi proses berfokus pada aliran, penggunaan, dan trasformasi data di dalam sistem informasi. Meliputi penciptaan representasi grafik seperti diagram aliran data(*data flow diagram*), bagan alir(*flow chart*), dan sebagainya. Pada pendekatan ini, data dilacak mulai dari asal/sumbernya, dikelola dalam beberapa proses/tahapan sampai ke tujuan akhirnya. Kelemahan dari pendekatan ini adalah struktur dasar/alami dari data tidak tergambarkan dengan baik dan data terikat pada suatu aplikasi tertentu.

2.3.1.2 Pendekatan Berorientasi Data

Pendekatan berorientasi data menggambarkan struktur dasar dari data, lepas dari dimana dan bagaimana data tersebut digunakan. Model data mendeskripsikan jenis/macam data dan hubungan antar data (E-R Diagram). Aturan bisnis menentukan bagaimana organisasi mengambil dan memproses data tersebut. Pendekatan ini biasanya fokus untuk perancangan basis data. Pada pendekatan ini, data didefinisikan lepas dari aplikasinya.

2.3.1.3 Pendekatan Berorientasi Objek

Pendekatan berorientasi objek mendeskripsikan sistem sebagai kelas(sekumpulan) objek dan keterhubungannya. Sebuah kelas objek merupakan abstraksi dari sehimpunan objek dengan atribut dan layanan(operasi) yang sama. UML menjadi standar untuk pemodelan berorientasi objek.

2.3.2 Model Pengembangan Sistem Informasi

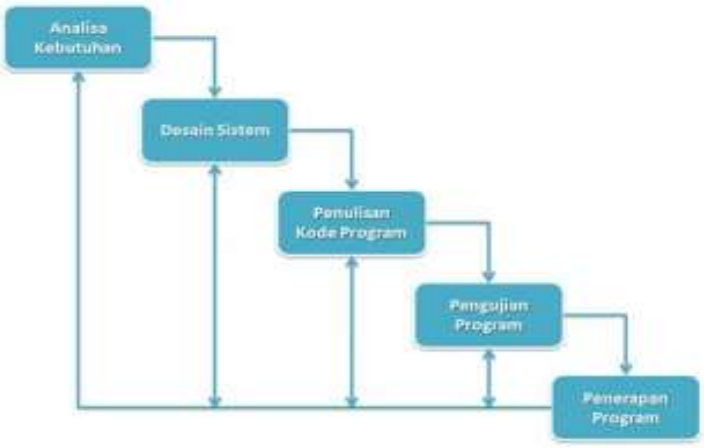
Dibahas hanya dua model pengembangan untuk membandingkan mana yang lebih efektif dalam penentuan model pengembangan yang akan ditentukan oleh penulis, kedua model tersebut yaitu model sekuensial linear dan model *prototype* sebagai pembandingnya.

2.3.2.1 Model Sekuensial Linear

Model sekuensial linear pertama kali muncul pada tahun 1970 yang diperkenalkan oleh Winston W. Royce. Walaupun sudah dikenal dalam waktu yang lama dan sering dianggap kuno, tetapi model ini paling banyak dipakai dalam industri perangkat lunak.

Model sekuensial linear berisi rangkaian proses yang disajikan secara terpisah, yaitu analisis kebutuhan perancangan, pengkodean, pengujian, serta implementasi dan pemeliharaan. Setelah setiap proses dilakukan, proses tersebut ditutup dan pengembangan dilanjutkan pada proses berikutnya.

Untuk mengatasi kekurangan-kekurangan tersebut, dibuatlah proses model sekuensial linear yang dimodifikasi. Keunggulan model ini dibandingkan model sekuensial linear biasa adalah memungkinkan tahap-tahap yang telah dilalui ditinjau kembali, sehingga jika terjadi kesalahan atau kekurangan dalam menentukan kebutuhan ditahap awal bisa dilakukan perbaikan atau penambahan lagi.



Gambar 2.1
Tahapan Model Sekuensial Linear

Model sekuensial linear melingkupi aktifitas-aktifitas sebagai berikut :

1. Rekayasa dan Pemodelan Sistem Informasi

Karena sistem bagian dari sebuah sistem yang lebih besar, kerja dimulai dengan membangun syarat dari semua elemen sistem dan mengalokasikan beberapa subset dari kebutuhan ke *software* tersebut. Pandangan sistem ini penting ketika *software* harus berhubungan dengan elemen-elemen yang lain seperti *software,* manusia, dan *database.*

2. Analisis dan Kebutuhan *Software*

Proses pengumpulan kebutuhan diintensifkan dan difokuskan, khususnya pada *software.* Untuk memahami sifat program yang dibangun, analis harus memahami dominasi informasi, tingkah laku, unjuk kerja, dan *interface* yang diperlukan. Kebutuhan baik untuk sistem maupun *software* didokumentasikan dan dilihat lagi dengan pelanggan/pengguna.

3. Desain

Desain *software* sebenarnya adalah proses multi langkah yang berfokus pada empat atribut sebuah program yang berbeda. Struktur data, arsitektur *software,* representasi *interface,* dan detail(algoritma) prosedural.

4. Generasi Kode

Desain harus diterjemahkan kedalam bentuk mesin yang bisa dibaca. Langkah pembuatan kode melakukan tugas tersebut. Jika desain dilakukan dengan cara yang lengkap pembuatan kode dapat diselesaikan secara mekanis.

5. Pengujian

Sekali program dibuat, pengujian program dimulai. Proses pengujian berfokus pada logika internal *software*, memastikan bahwa semua pernyataan sudah diuji, dan pada eksternal fungsional, yaitu mengarahkan pengujian untuk menemukan kesalahan-kesalahan dan memastikan bahwa input yang dibatasi akan memberikan hasil aktual yang sesuai dengan hasil yang dibutuhkan.

6. Pemeliharaan

Software akan mengalami perubahan setelah disampaikan kepada pelanggan. Perubahan akan terjadi karena kesalahan-kesalahan ditentukan, karena *software* harus disesuaikan untuk mengakomodasi perubahan-perubahan di dalam lingkungan eksternalnya, atau karena pelanggan membutuhkan perkembangan fungsional. Pemeliharaan *software* mengaplikasikan lagi setiap fase program sebelumnya dan tidak membuat yang baru.

2.3.2.1.1 Kelebihan Model Sekuensial Linear

Adapun kelebihan pada model pengembangan sistem informasi pada model sekuensial linear, diantaranya :

a. Merupakan model pengembangan paling handal dan paling lama digunakan.
b. Cocok untuk *system software* berskala besar.
c. Cocok untuk *system software* yang bersifat generik.
d. Pengerjaan *project system* akan terjadwal dengan baik dan mudah dikontrol.

2.3.2.1.2 Kekurangan Model Sekuensial Linear

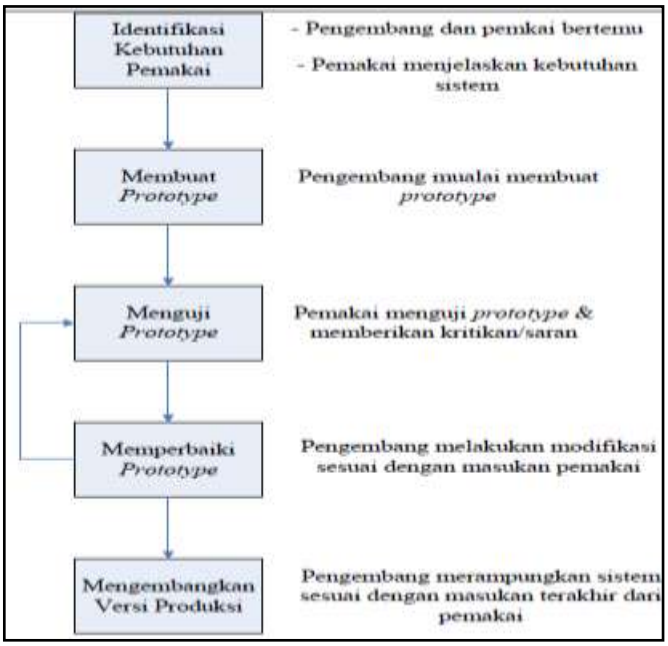
Adapun kekurangan pada model pengembangan sistem informasi pada model sekuensial linear, diantaranya :

a. Persyaratan sistem harus digambarkan dengan jelas. Rincian proses harus benar-benar jelas dan tidak boleh berubah-ubah.
b. Sulit untuk mengadaptasi jika terjadi perubahan spesifikasi pada suatu tahapan pengembangan.

2.3.2.2 Model Prototype

Model *prototype* merupakan suatu metode pengembangan sistem yang berguna dalam hal untuk pembuatan sesuatu perancangan program yang akan dibuat dengan cepat dan bertahap, sehingga dapat segera dievaluasi oleh pemakai. Selain itu model *prototype* menjadikan proses pengembangan sistem informasi lebih cepat dan lebih mudah, terutama pada keadaan kebutuhan pemakai sulit untuk diidentifikasi.

Berikut ini merupakan mekanisme pengembangan sistem dengan menggunakan Prototype :



Gambar 2.2
Mekanisme Pengembangan Sistem Dengan *Prototype*
Sumber : Abdul Kadir, "Pengenalan Sistem Informasi". (2002:416)

Tahapan-tahapan *prototyping* yaitu :

1. Pengumpulan kebutuhan *user* dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.
2. Membangun *prototyping* dengan membuat perancangan sementara yang berfokus pada penyajian kepada *user,* misalnya dengan membuat format input dan format output.
3. Evaluasi *prototyping* dilakukan oleh *user* apakah *prototyping* yang sudah dibangun sudah sesuai dengan keinginan *user*. Jika sudah sesuai maka

langkah 4 akan diambil, jika tidak *prototyping* direvisi dengan mengulangi langkah 1, 2, dan 3.

4. Mengkodekan sistem yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.
5. Menguji sistem. Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini dilakukan dengan *White Box, Black Box, Basis Path*, pengujian arsitektur, dan lain-lain.
6. Mengevaluasi apakah sistem yang sudah jadi telah sesuai dengan yang diharapkan . Jika ya, langkah 7 dilakukan, jika tidak, ulangi langkah 4 dan 5.
7. Menggunakan sistem. Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan.

2.3.2.2.1 Kelebihan *Prototype*

Adapun kelebihan pada model pengembangan sistem informasi menggunakan model *prototype*, diantaranya :

a. *Prototype* melibatkan user dalam analisa dan desain.
b. Meningkatkan kepuasan pemakai dan mengurangi resiko pemakai tidak menggunakan sistem mengingat keterlibatan mereka yang sangat tinggi sehinga sistem memenuhi kebutuhan mereka dengan lebih baik.
c. Mempunyai kemampuan menangkap *requirement* secara konkret dari pada secara abstrak.
d. Mempersingkat waktu pengembangan Sistem Informasi.
e. Menghemat biaya.

2.3.2.2.1 Kelemahan *Prototype*

Adapun kelemahan atau kekurangan pada model pengembangan sistem informasi menggunakan model *prototype*, diantaranya :

a. Proses analisis dan perancangan terlalu singkat.

b. Mengesampingkan alternatif pemecahan masalah.

c. Bisa kurang fleksibel dalam mengahdapi perubahan.

d. *Prototype* yang dihasilkan tidak selamanya mudah dirubah.

e. *Prototype* terlalu cepat selesai.

Apabila tidak dikelola dengan baik, siklus *prototype* menjadi tidak pernah berakhir. Hal ini disebabkan permintaan terhadap perubahan terlalu mudah untuk dipenuhi.

2.4 UML

UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standar. (Chonoles, 2003: bab 1) mengatakan sebagai bahasa, berarti UML memiliki sintaks dan semantik. Ketika kita membuat model menggunakan konsep UML ada aturan-aturan yang harus diikuti. Bagaimana elemen-elemen pada model-model yang kita buat berhubungan satu dengan lainnya harus mengikuti standar yang ada.

UML diaplikasikan untuk maksud tertentu, antara lain :

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.

3. Menjabarkan sistem secara terinci untuk menganalisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

UML mungkin bisa disebut pengganti dari *Object Oriented Analysis and Design* (OOAD) yang tersebar selama perang metode antara tahun 80-an sampai awal 90-an. UML merupakan bahasa komponen dari sebuah metode dan dilengkapi secara terpisah oleh *Rational Unified Process* (RUP). UML merupakan gabungan dari tiga metode utama bahasa pemodelan di dalam industry yaitu : *Booch*, *Rumbaugh*, dan *Jacobson*. UML melalui proses standarisasi dengan *Object Management Group* (OMG), dan sekarang menjadi standar OMG.

Aturan-aturan pada UML bisa diklasifikasikan ke dalam :

1. *Syntactic* : rincian aspek dan kombinasi aturan.
2. *Semantic* : rincian arti dari simbol, secara individual dan dalam konteks.
3. *Pragmatic* : panduan bagaimana menggunakan bahasanya (dari simbol).

Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah : metodologi *Booch*, metodologi *Coad*, metodologi OOSE, metodologi OMT, metodologi *Shlaer-Mellor*, metodologi *Wirfs-Brock*, dan lainnya. Pada masa itu terkenal sebagai masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasinya, yang mengakibatkan timbul masalah baru apabila kita bekerja sama dengan group/instansi lain yang menggunakan metodologi yang berlainan.

2.4.1 Diagram UML

UML dikelompokan atas beberapa diagram-diagram sistem menurut aspek atau sudut pandang tertentu. Sedangkan diagram menggambarkan permasalahan maupun solusi dari permasalahan suatu model. Berikut penjelasan mengenai beberapa Diagram UML, yaitu :

1. *Use Case*

Diagram *use case* memperlihatkan himpunan *use-case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.

2. Diagram Kelas

Diagaram ini memperlihatkan kumpulan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram kelas umum dijumpai pada model berorientasi objek.

3. Diagram Aktivitas

Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu sistem dan memberi tekanan pada aliran kendali antar objek.

4. Diagram Urutan (*sequence)*

Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.

5. Diagram Komunikasi

Diagram komunikasi sebagai pengganti diagram kolaborasi yang menekankan organisasi struktural dari objek-objek yang menerima serta mengirim pesan.

6. Diagram *statechart*

Diagram *statechart* memperlihatkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian, serta aktifitas.

7. Diagram Paket

Diagaram ini memperlihatkan kumpulan kelas-kelas, merupakan bagian dari diagram komponen.

8. Diagram Komponen

Diagram komponen memperlihatkan organisasi serta ketergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya.

9. Diagram *Deployment*

Diagram *deployment* memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*). Memuat simpul-simpul beserta komponen-komponen yang ada di dalamnya. Diagram *deployment* berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen.

2.5 *Android*

Android telah menjadi sebuah sistem operasi yang banyak diandalkan pengguna ponsel khususnya ponsel pintar. Berikut beberapa pembahasan mengenai *android* :

2.5.1 Pengertian *Android*

Android adalah sistem operasi untuk telepon seluler berbasis linux yang mencangkup sistem operasi, *middleware*, dan aplikasi. *Android* menyediakan *platform* terbuka bagi para pengembang untuk melakukan pengembangan sesuai dengan yang diharapkan.

2.5.2 Sejarah *Android*

Awalnya Google Inc membeli *Android* Inc, yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan *android*, dibentuklah *Open Handset Aliance,* konsorium dari 34 perusahaan perancti keras, peranti lunak, dan telekomunikasi. Termasuk Google, HTC, Motorolla, Intel, Nvidia, T-*Mobile*, dan Qualcomm.

Pada saat perintisan perdana *Android* yaitu pada tanggal 5 November 2007, *Android* bersama *Open Handset Aliance* menyatakan untuk mendukung pengembangan *open source* pada perangkat *mobile*. Sementara itu, di lain pihak Google merilis kode-kode *Android* di bawah lisensi Apache, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler.

Pada September 2007 Google mengenalkan Nexus One, salah satu jenis *smartphone* yang menggunakan *Android* sebagai sistem operasinya. *Smartphone*

tersebut diproduksi oleh HTC *corporation* dan tersedia di pasaran pada tanggal 5 Januari 2010.

Pada tanggal 9 Desember 2008 diumumkan anggota baru yang bergabung kedalam *Open Handset Aliance* dalam program kerja *Android* ARM Holdings, Atheros Communications, diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc. Seiring pembentukan *Open Handset Alliance*, OHA mengumumkan produk perdana mereka, *android* perangkat *mobile* yang merupakan modifikasi *kernel Linux* 2.6. Sejak *Android* dirilis, telah dilakukan berbagai pembaharuan berupa perbaikan *bug* dan penambahan fitur baru.

Pada saat ini, kebanyakan vendor-vendor *smartphone* sudah memproduksi *smartphone* berbasis *android*, vendor-vendor itu antara lain HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, LG, Acer, Philips, T-Mobile, Nexian, IMO, Asus dan masih banyak lagi vendor *smartphone* didunia yang memproduksi *android*. Hal ini karena *android* adalah sistem operasi yang *open source* sehingga bebas didistribusikan dan dipakai oleh vendor manapun.

2.6 *JQuery Mobile*

Perkembangan pengembangan aplikasi untuk *mobile* telah diambang perubahan secara besar-besaran, sehingga dibutuhkan pemrograman dimana bisa dijalankan di seluruh sistem operasi. Dibuatlah *jquery mobile framework* untuk

mengatasi permasalahan tersebut. Adapun beberapa pembahasan mengenai *jquery framework* sebagai berikut :

2.6.1 Pengenalan *JQuery Mobile*

Menurut Zaki (2012 : 2). *JQuery* adalah *library JavaScript* yang populer untuk membangun *Web* interaktif yang kaya serta bisa dipakai juga untuk aplikasi *Web*. *JavaScript* sendiri merupakan bahasa pemograman *client – side* untuk memberikan efek dinamis pada aplikasi. Namun dalam pembangunanya *jQuery* tidak memiliki fitur yang khusus dirancang untuk membangun aplikasi *Web mobile*, sehingga fitur yang telah dibuat tidak akan tertampak baik saat dijalankan dalam perangkat *mobile*.

Untuk masalah tersebut, tim pengembang *jQuery* membuat sebuah *library* yang dikhususkan untuk pengembangan aplikasi *mobile* bernama *JQuery Mobile*.

Zaki (2012 : 2) menyebutkan, *jQuery Mobile* adalah *framework* yang dibangun di atas *JQuery*, yang menyediakan berbagai elemen *user-interface* dan fitur yang digunakan dalam aplikasi *mobile*.

Penggunaan *jQuery Mobile* memungkinkan pengembang dapat mengembangkan program dengan lebih cepat, karena aksebilitas lebih cepat dan tidak memerlukan konfigurasi *javascript*. Sehingga *library* ini dapat lebih mudah diakses dan memiliki ukuran yang lebih ringkas. Dukungan penggunaan layar sentuh membuat aplikasi yang dibangun lebih optimal pada beberapa piranti yang ada atau dapat dikatakan pula aplikasi tersebut dapat bersifat *multi platform*.

2.6.2 Sifat dari *JQuery Mobile*

Sifat-sifat yang dimiliki oleh *framework JQuery Mobile* diantaranya :

a. Memudahkan pengembangan *user interface* untuk *mobile web apps*.
b. Konfigurasi antar mukanya bersifat *markup-driven*, yang berarti dapat membuat seluruh aplikasi antar muka dasar dalam kode HTML, tanpa perlu menulis satu baris *JavaScript*.
c. Menyediakan event khusus untuk mendeteksi event dari piranti *mobile* dan piranti *touchscreen* seperti *tap*, *tap-and-hold*, *swipe*, dan perubahan orientasi.
d. Memastikan bahwa *user interface* bekerja pada *web browser.*