

3. PERANCANGAN DAN IMPLEMENTASI

3.1. Analisis Kebutuhan

Gempabumi adalah suatu getaran lempengan bumi yang terjadi secara alami. Lembaga Pemerintahan Non Kementrian BMKG adalah bagian dalam pemerintah yang memantau terjadinya gempabumi. Informasi adalah salah satu kebutuhan bagi masyarakat. Maka penulis ingin membuat suatu aplikasi yang dapat membantu pengguna untuk mendapatkan informasi mengenai gempabumi di negara Indonesia.

Pada pembangunan aplikasi penulis membutuhkan data gempabumi yang akan ditampilkan pada aplikasi bergerak yang dibangun. Terdapat API yang sudah tersedia dan dapat digunakan oleh penulis dalam membangun aplikasi. API yang sudah tersedia, data gempabumi terbaru, gempabumi kurang dari 5 skala richter, gempabumi dirasakan dan jaringan stasiun gempabumi.

Pembangunan aplikasi membutuhkan perangkat keras untuk membuat dan menjalankan aplikasi informasi gempabumi dan membutuhkan perangkat lunak yang dibutuhkan dalam pembangunan. Spesifikasi *hardware* komputer dan *software* yang digunakan dalam pembuatan aplikasi Android menggunakan *Hybrid Technology*. Perangkat keras yang digunakan adalah :

1. Laptop MacBook Pro (Retina, 13-Inch, Late 2012)
2. Intel Core i5 2.5GHz
3. Memory 8 GB 1600 MHz DDR3
4. Graphics Intel HD Graphics 4000 1536 MB

Perangkat lunak yang digunakan dalam pembangunan aplikasi bergerak mengenai informasi gempabumi berbasis Android adalah :

1. Sistem Operasi OS X EI Capitan Version 10.11.5
2. Atom 1.7.4
3. git version 2.6.4 (Apple Git-63)
4. Terminal Version 2.6.1 (361.1)
5. Node JS atau *Node Package Manager* (npm)
6. Ionic dan Cordova (menggunakan npm)

7. Bower
8. *Android Software Development Kit (SDK)*
9. *Java Development Kit (JDK)*
10. *Android Debug Bridge (ADB)*
11. *Android Virtual Device (AVD)*
12. *Safari Version 9.1.2 (11601.7.7)*

Seiring dengan pesatnya kemajuan teknologi, penulis membuat aplikasi ini untuk memberikan suatu pelayanan kepada publik, dan untuk menyeimbangkan pesatnya kemajuan teknologi dalam bidang aplikasi bergerak. Aplikasi ini dibuat untuk membantu masyarakat Indonesia dalam mengetahui informasi gempabumi terbaru atau sebelumnya, dan aplikasi ini layanan alternatif untuk mengetahui informasi gempabumi di wilayah Negara Indonesia.

3.2. Perancangan

Aplikasi ini dirancang dengan beberapa tahap perancangan, yaitu dimulai dari perancangan *storyboard* tentang jalan cerita aplikasi, perancangan struktur navigasi tentang cara menggunakan aplikasi, perancangan antarmuka tentang tampilan aplikasi, dan perancangan alur program.

3.2.1 Perancangan Storyboard

Perancangan *storyboard* adalah perancangan tentang jalan cerita atau alur cerita dari aplikasi yang disajikan dalam bentuk gambar. *Storyboard* ini akan menggambarkan tentang alur dari aplikasi informasi gempabumi. *User* atau pengguna akan disajikan beberapa menu untuk memilih tampilan informasi yang ingin ditampilkan pada *smartphone* pengguna.

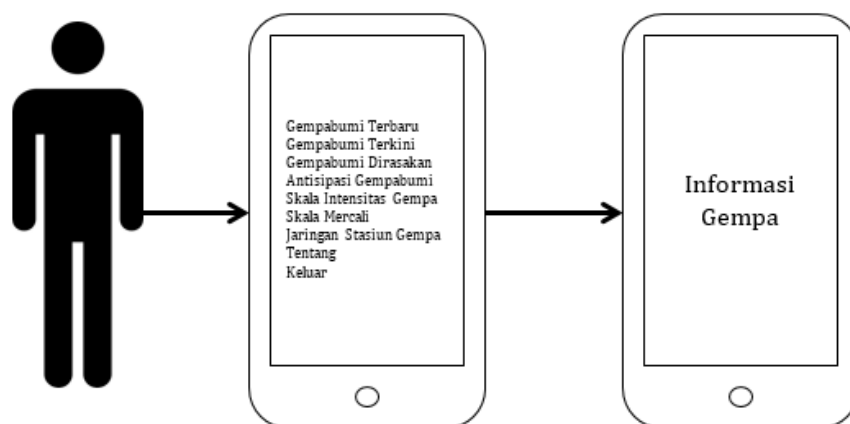
Pengguna akan dihadapkan dengan 8 pilihan menu yang diatur oleh tampilan *sidemenu* dan satu *button* keluar untuk menutup aplikasi. Menu pada aplikasi ini memiliki informasi yang dibutuhkan oleh pengguna, dalam hal mendapatkan informasi seputar gempabumi. Apabila pengguna memilih menu gempabumi terbaru, maka akan muncul informasi satu gempabumi terbaru yang disajikan dalam peta dari Google Maps dan terdapat informasi berupa lokasi,

tanggal dan jam, *latitude* dan *longtitude*, lintang dan bujur, magnitude dan kedalaman.

Apabila pengguna memilih menu gempabumi kurang dari 5 skala richter, maka akan muncul beberapa daftar gempabumi yang kurang dari 5 skala richter dan terdapat informasi berupa lokasi, tanggal dan jam, *latitude* dan *longtitude*, lintang dan bujur, magnitude dan kedalaman. Apabila pengguna memilih menu gempabumi dirasakan, maka akan muncul beberapa daftar gempabumi dirasakan dan terdapat informasi berupa lokasi, tanggal dan jam, *latitude* dan *longtitude*, lintang dan bujur, magnitude dan kedalaman.

Apabila pengguna memilih menu antisipasi gempabumi, maka akan muncul beberapa informasi yang harus dilakukan sebelum, sesaat dan sesudah terjadinya gempabumi. Informasi disajikan dalam bentuk gambar dan deskripsikan pada setiap informasi antisipasi gempabumi. Apabila pengguna memilih menu skala mercalli, maka akan muncul beberapa informasi tingkat-tingkatan mengenai skala mercalli. Informasi disajikan dalam bentuk gambar dan deskripsi pada setiap informasi skala mercalli.

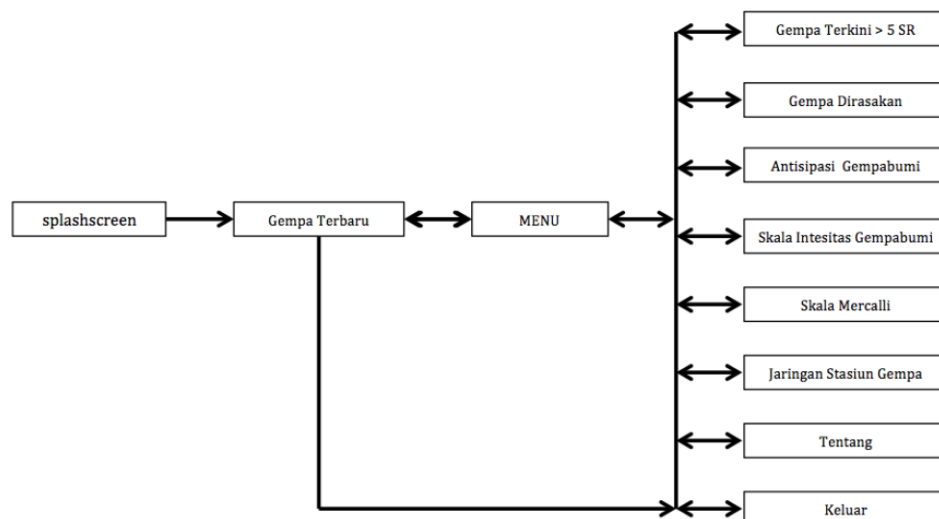
Menu skala intensitas gempabumi terdapat informasi mengenai tingkatan dalam pendefinisian getaran didalam bumi. Jika pengguna memilih menu jaringan stasiun gempabumi, maka akan ditampilkan informasi lokasi-lokasi yang memantau gempabumi di wilayah Negara Indonesia yang dapat dilihat pada Gambar 3.1.



Gambar 3.1 *Storyboard* informasi gempabumi

3.2.2. Perancangan Struktur Navigasi

Struktur navigasi merupakan rancangan hubungan atau rantai kerja yang menggambarkan sistem secara keseluruhan. Aplikasi ini menggunakan struktur navigasi *hirarchi* atau bercabang untuk menggambarkan sistem aplikasi. Struktur navigasi bercabang yang digunakan pada aplikasi ini karena aplikasi ini memiliki data-data yang beda satu sama lain pada setiap tampilan. Perpindahan tampilan tidak searah karena terdapat tampilan *sidemenu* yang disediakan oleh kerangka kerja Ionic, struktur navigasi pembangunan aplikasi informasi gempabumi dapat dilihat pada Gambar 3.2.



Gambar 3.2 Struktur Navigasi Informasi Gempabumi

Struktur navigasi ini dimulai dari *splashscreen*, yang berlanjut secara otomatis ke tampilan gempabumi terbaru. Pada tampilan menu akan terdapat sembilan kategori yang berupa *ion-item* atau pada struktur navigasi ini dikatakan sebagai menu gempabumi terbaru, gempabumi kurang dari 5 skala richter, gempabumi yang dirasakan, antisipasi gempabumi, skala intensitas gempa, skala mercalli, jaringan stasiun gempabumi, tentang dan keluar.

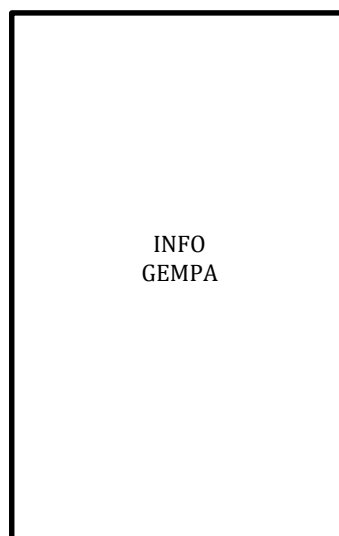
Apabila pengguna memilih menu gempabumi terbaru, maka akan menampilkan informasi gempabumi yang sangat terbaru, dan jika pengguna memilih menu gempabumi kurang dari 5 skala richter, maka aplikasi akan

menampilkan informasi gempa-gempa yang kurang dari 5 skala richter. Jika pengguna memilih menu antisipasi gempabumi, maka aplikasi akan menampilkan informasi antisipasi gempabumi sebelum, sesaat dan sesudah terjadinya gempabumi.

Pengguna dapat memilih menu skala intensitas gempa, skala mercalli atau jaringan stasiun gempabumi, apabila pengguna memilih menu skala intensitas maka akan menampilkan informasi tentang tingkatan getaran pada bumi, apabila pengguna memilih menu skala mercalli maka aplikasi akan menampilkan informasi seputar tingkat-tingkatan skala mercalli. Apabila pengguna memilih menu jaringan stasiun gempabumi, maka aplikasi akan menampilkan informasi seputar lokasi pantau gempabumi di wilayah Negara Indonesia.

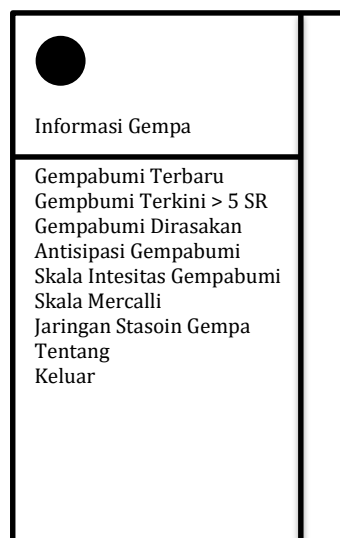
3.2.3. Perancangan Antarmuka

Perancangan antarmuka merupakan salah satu unsur terpenting dalam pembangunan aplikasi. Perancangan antarmuka dapat menentukan keberhasilan dari sebuah produk. Antarmuka yang dibuat harus semenarik mungkin agar pengguna tertarik untuk menggunakan aplikasi informasi gempabumi ini. Sebuah antarmuka dirancang sesuai kebutuhan dari aplikasi. Rancangan ini dibuat dengan menyesuaikan pengguna yang menjadi tujuan akhir dari aplikasi. Rancangan pertama adalah pembuatan *splashscreen*.

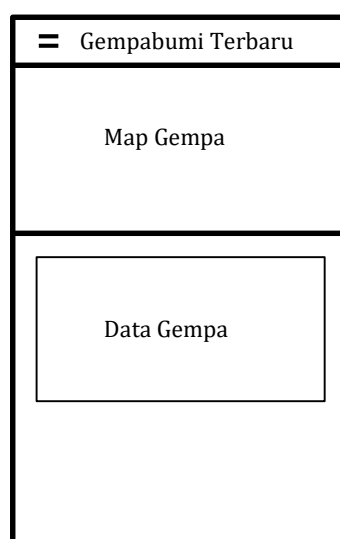


Gambar 3.3 Perancangan antarmuka *splashscreen*

Gambar 3.3 merupakan rancangan awal ketika masuk kedalam aplikasi informasi gempabumi. Ketika pengguna membuka aplikasi ini maka akan tampil *splashscreen* seperti diatas. Setelah itu akan ada tampilan selanjutnya dimana terdapat 8 menu yaitu menu gempabumi terbaru, gempabumi kurang dari 5 skala richter, gempabumi dirasakan, antisipasi gempabumi, skala intensitas gempa, skala mercalli dan jaringan stasiun gempabumi, tentang dan satu *button* keluar.



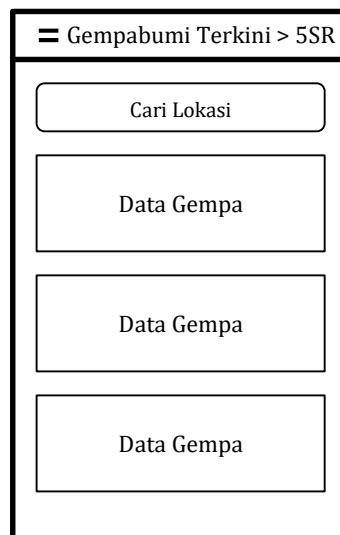
Gambar 3.4 Perancangan antarmuka tampilan *sidemenu*



Gambar 3.5 Perancangan antarmuka gempabumi terbaru

Gambar 3.4 merupakan rancangan tampilan menu awal, dimana tampilan ini untuk memilih menu-menu pada aplikasi. Tampilan ketiga adalah tampilan gempabumi terbaru, pada tampilan ini terdapat sebuah *card panel* yang didalamnya terdapat peta dari Google Maps dan beberapa informasi mengenai gempabumi terbaru terdapat pada Gambar 3.5.

Jika pengguna kembali memilih menu pada *sidemenu* maka akan keluar beberapa menu seperti pada Gambar 3.4, jika pengguna memilih menu gempabumi kurang dari 5 skala richter. Pada tampilan ini terdapat *list card panel* dari beberapa data gempabumi kurang dari 5 skala richter seperti Gambar 3.6.



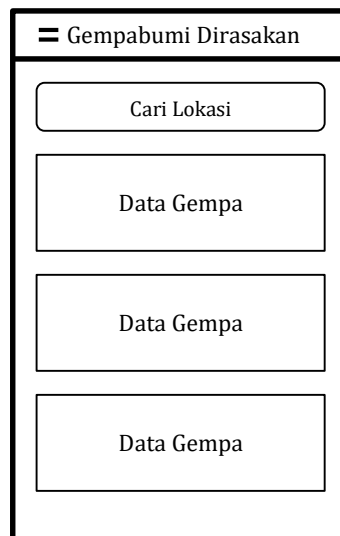
Gambar 3.6 Perancangan antarmuka gempabumi terkini

Pengguna dapat melihat gempabumi yang dirasakan di sekitar wilayah Indonesia dengan memilih menu gempabumi dirasakan. Pada tampilan ini terdapat beberapa data gempabumi yang dirasakan di Indonesia, ditampilkan dalam bentuk *list card panel* yang dapat dilihat pada Gambar 3.7.

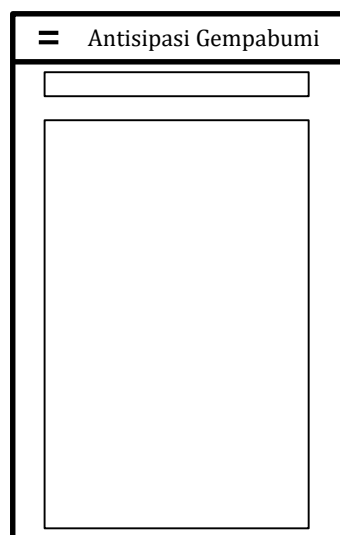
Antisipasi gempabumi dan skala mercalli dapat dilihat oleh pengguna untuk mengetahui apa yang harus dilakukan sebelum, saat dan sesudah terjadinya gempabumi, dan mengetahui tingkat-tingkatan guncangan atau getaran gempabumi. Apabila pengguna memilih menu antisipasi gempabumi dan skala mercalli. Pada tampilan antisipasi gempabumi terdapat *card panel* yang berisi

informasi yang di lengkapi dengan animasi gambar dan deskripsi mengenaiya dapat dilihat pada Gambar 3.8.

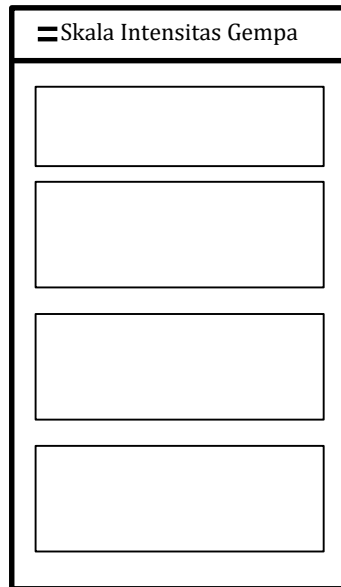
Pada tampilan skala intensitas gempa terdapat *card panel* yang berisi gambar animasi dan penjelasan tentang setiap tingkatan-tingkatan getaran didalam bumi, dapat dilihat pada Gambat 3.9.



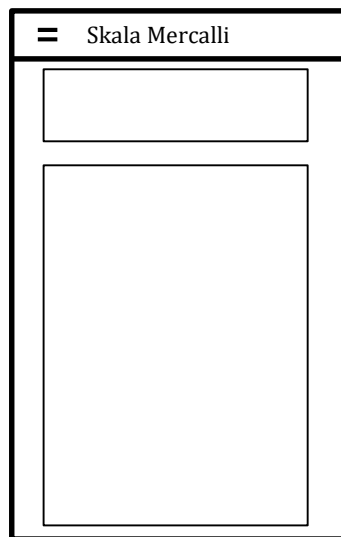
Gambar 3.7 Perancangan antarmuka gempabumi dirasakan



Gambar 3.8 Perancangan antarmuka antisipasi gempabumi



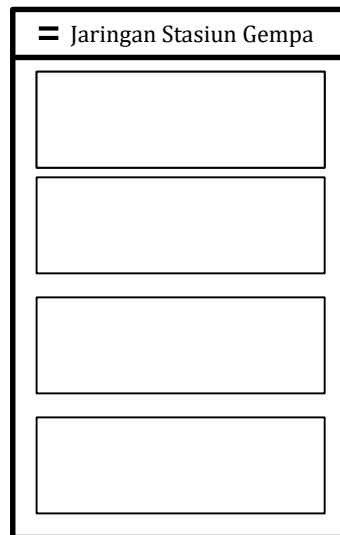
Gambar 3.9 Perancangan skala intensitas gempabumi



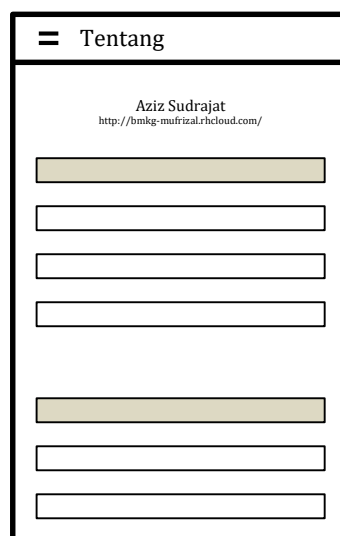
Gambar 3.10 Perancangan skala mercalli

Pada rancangan yang terdapat pada Gambar 3.10 tampilan skala mercalli terdapat *card panel* yang berisi gambar animasi dan penjelasan tentang setiap tingkatan-tingkatan skala mercalli. Pengguna dapat mengetahui tentang lokasi-lokasi yang memantau terjadinya gempabumi di negara Indonesia, dengan memilih menu jaringan stasiun gempabumi. Pada tampilan ini terdapat panel

informasi yang berisi data-data dari setiap stasiun atau lokasi-lokasi yang memantau terjadinya gempa bumi di negara Indonesia. Tampilan dapat dilihat pada Gambar 3.11.



Gambar 3.11 Perancangan jaringan stasiun gempabumi

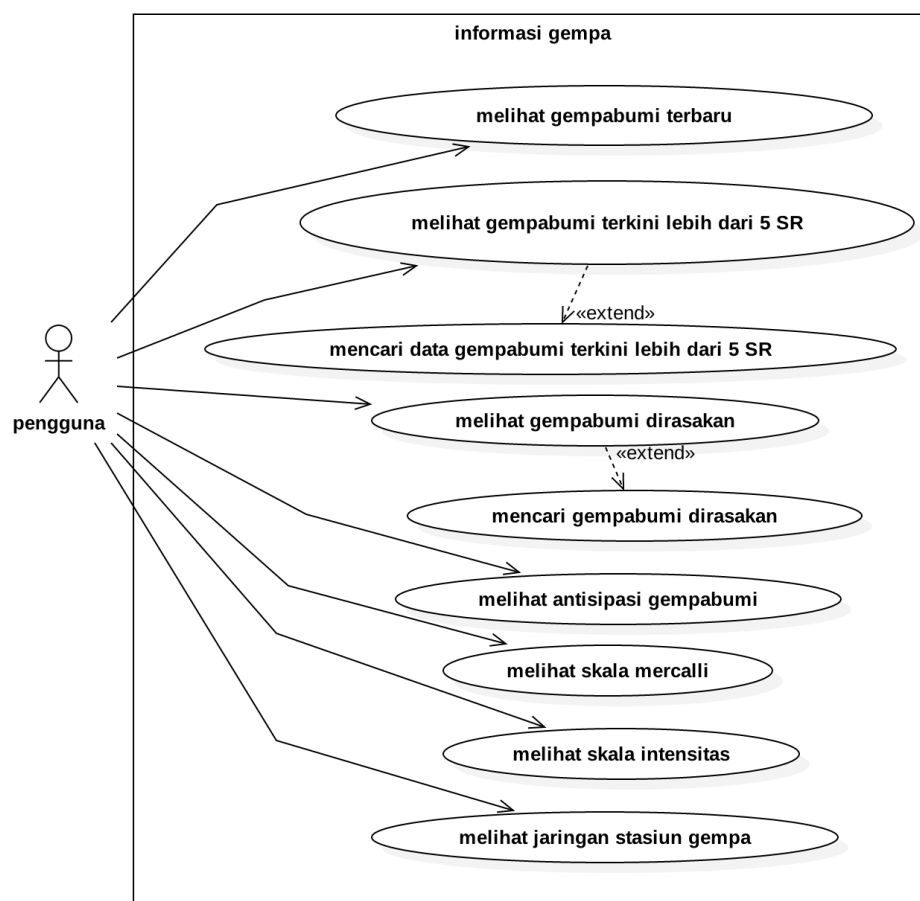


Gambar 3.12 Perancangan tentang

Pada tampilan yang dirancang pada Gambar 3.12 menjelaskan tentang informasi pembuatan aplikasi ini, aplikasi dibuat menggunakan *Hybrid Technology* yaitu kerangka kerja Ionic dan menggunakan beberapa *library* yang dibutuhkan dalam aplikasi informasi gempa bumi.

3.2.4. Perancangan Alur Program

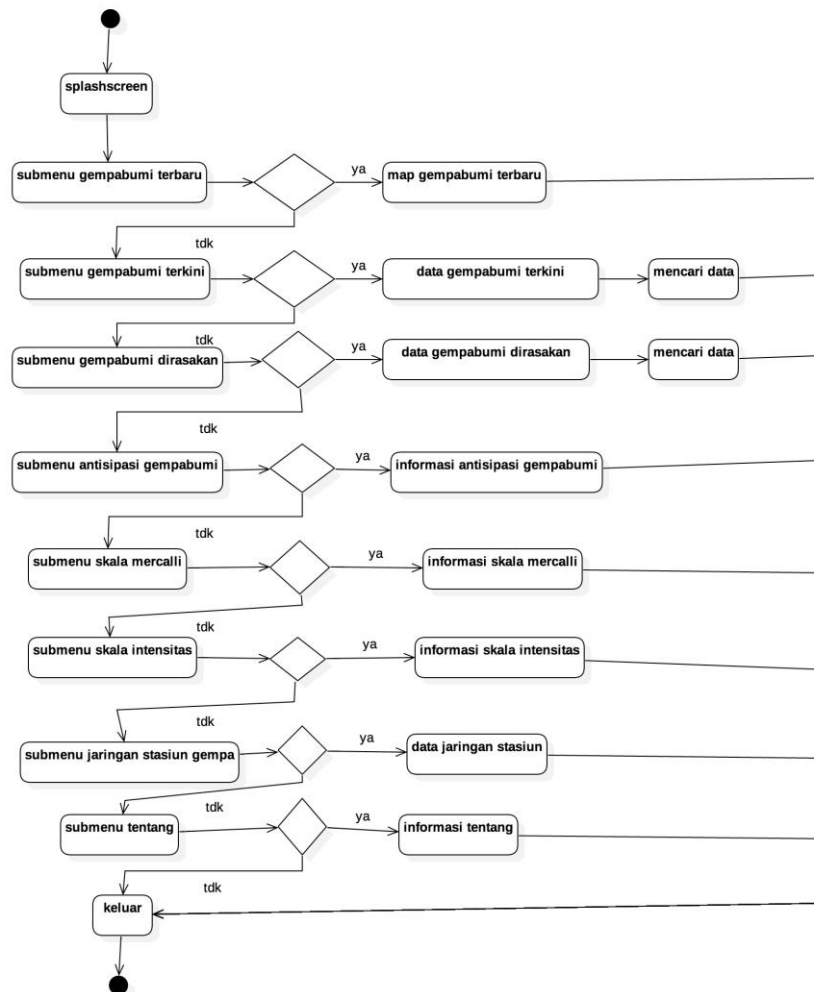
Perancangan aplikasi ini juga menggunakan *Unified Model Language* (UML) untuk memvisualisasikan rancangan model sistem atau alur program. Model sistem ini akan dirancang menggunakan dua buah diagram yang terdapat pada UML yaitu *Use case Diagram* dan *Activity Diagram*. Pada *Use Case Diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Diagram ini akan mempresentasikan antara *actor* dengan sistem yang ada terdapat pada Gambar 3.13.



Gambar 3.13 *Use Case Diagram*

Pada rancangan *Use Case Diagram* terlihat bahwa pengguna berinteraksi secara langsung dengan aplikasi informasi, dimana dengan pengguna dapat melihat beberapa informasi dan mencarinya berdasarkan lokasi pada data

gempabumi. Selanjutnya alur dari aplikasi akan digambarkan dengan *Activity Diagram*. Diagram ini akan menggambarkan alur aktifitas di dalam aplikasi, bagaimana aktifitas berawal dan aplikasi berakhir yang dapat dilihat pada Gambar 3.14.



Gambar 3.14 *Activity Diagram*

Pada *Activity Diagram* menggambarkan alur aktifitas dari aplikasi informasi gempabumi. Aplikasi ini dimulai dari sebuah *smartphone* yang didalamnya terdapat aplikasi informasi gempabumi. Pada aplikasi informasi gempabumi, akan terdapat tampilan awal sebelum masuk kedalam menu aplikasi yaitu tampilan *splashscreen*, dimana tersedia delapan menu yang akan menampilkan beberapa

tampilan informasi seputar gempa bumi. Apabila pengguna memilih salah satu dari menu tersebut, maka menu tersebut akan mengarah kepada data yang dimiliki setiap menu tersebut.

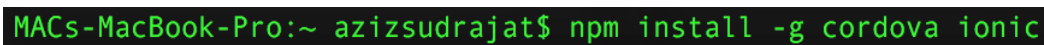
3.3. Implementasi dan Pembuatan Aplikasi

Setelah melakukan perancangan *storyboard*, struktur navigasi, antarmuka dan alur program. Tahap selanjutnya yang akan dilakukan adalah pelaksanaan dari rancangan yang telah dibuat secara matang dan terperinci. Seluruh perancangan akan diimplementasikan ke dalam pembangunan program, dan dalam pembangunan program terdapat beberapa proses dan langkah-langkah.

Pembuatan *project* Ionic dalam aplikasi informasi gempa bumi ini menggunakan tampilan *sidemenu* yang telah disediakan Ionic. Pembuatan halaman-halaman tampilan gempa bumi dengan mengikuti judul menu pada *sidemenu project* Ionic. Pada tampilan utama akan menampilkan map lokasi gempa bumi paling terbaru dengan memanfaatkan *library* dari Google Maps.

3.3.1. Persiapan Project Ionic

Sebelum memulai pembuatan aplikasi atau pengkodean program Ionic, ada beberapa *dependency* yang harus terpasang di dalam komputer kita, seperti *Node Package Manager* (npm), *Front-End Package Manager* (bower), Ionic dan Cordova. Pertama yang harus dilakukan adalah menginstal npm terlebih dahulu yang dapat diunduh dari <https://nodejs.org/en/download/>, tahap selanjutnya menginstal Ionic bersamaan dengan Cordova.



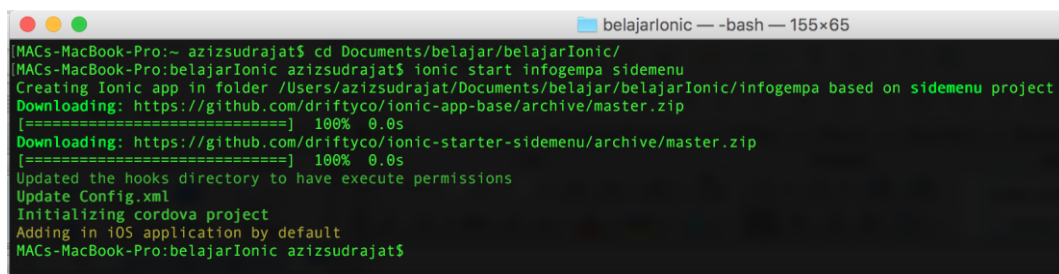
```
MACs-MacBook-Pro:~ azizsudrajat$ npm install -g cordova ionic
```

Gambar 3.15 Instal Ionic dan Cordova

Ionic merupakan sebuah *framework* untuk mempermudah pengembang *website* dalam membuat aplikasi *Cross Platform* tanpa harus menguasai pemrograman *Android Native* dan *Objectiv-C* untuk iOS. Dalam membangun aplikasi menggunakan Ionic, pengembang *website* sudah terbiasa dengan bahasa

pemrograman yang digunakan karena Ionic menggunakan HTML, CSS dan JS yang dibungkus oleh Angular.

Ionic memerlukan terminal pada komputer, konfigurasi Ionic seluruhnya dilakukan pada terminal seperti mengambil *file* Ionic pun menggunakan terminal.

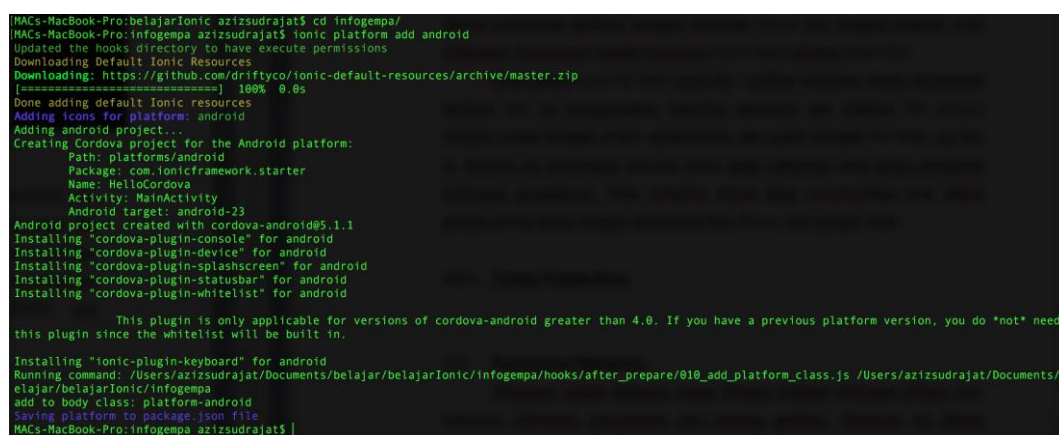


```

MACs-MacBook-Pro:~ azizsudrajat$ cd Documents/belajar/belajarIonic/
MACs-MacBook-Pro:belajarIonic azizsudrajat$ ionic start infogempa sidemenu
Creating Ionic app in folder /Users/azizsudrajat/Documents/belajar/belajarIonic based on sidemenu project
Downloading: https://github.com/driftyco/ionic-app-base/archive/master.zip
[=====] 100% 0.0s
Downloading: https://github.com/driftyco/ionic-starter-sidemenu/archive/master.zip
[=====] 100% 0.0s
Updated the hooks directory to have execute permissions
Update Config.xml
Initializing cordova project
Adding in iOS application by default
MACs-MacBook-Pro:belajarIonic azizsudrajat$
  
```

Gambar 3.16 Start *project* Ionic

Setelah sudah dapat mengunduh *file* Ionic menggunakan terminal, lalu dalam pengunduhan penulis menyantumkan nama folder dengan *infogempa* dan jenis tampilan *sidemenu*, karena dalam pembangunan aplikasi ini menggunakan MacOS secara *default build platform* berupa iOS. Ionic dapat menambahkan *build platform* android dengan membutuhkan JAVA_HOME dan ANDROID_HOME dalam membuat ke Android *Package* (APK).



```

MACs-MacBook-Pro:belajarIonic azizsudrajat$ cd infogempa/
MACs-MacBook-Pro:infogempa azizsudrajat$ ionic platform add android
Updated the hooks directory to have execute permissions
Downloading Default Ionic Resources
Downloading: https://github.com/driftyco/ionic-default-resources/archive/master.zip
[=====] 100% 0.0s
Done adding default Ionic resources
Adding icons for platform: android
Adding android project...
Creating Cordova project for the Android platform:
  Path: platforms/android
  Package: com.ionicframework.starter
  Name: HelloCordova
  Activity: MainActivity
  Android target: android-23
Android project created with cordova-android@5.1.1
Installing "cordova-plugin-console" for android
Installing "cordova-plugin-device" for android
Installing "cordova-plugin-splashscreen" for android
Installing "cordova-plugin-statusbar" for android
Installing "cordova-plugin-whitelist" for android
This plugin is only applicable for versions of cordova-android greater than 4.0. If you have a previous platform version, you do "not" need this plugin since the whitelist will be built in.
Installing "ionic-plugin-keyboard" for android
Running command: /Users/azizsudrajat/Documents/belajar/belajarIonic/infogempa/hooks/after_prepare/010_add_platform_class.js /Users/azizsudrajat/Documents/belajar/belajarIonic/infogempa
add to body class: platform-android
Saving platform to package.json file
MACs-MacBook-Pro:infogempa azizsudrajat$
  
```

Gambar 3.17 Ionic *platform* Android

Pemasangan *library* yang digunakan dalam pembangunan program ini menggunakan bower, Bower berfungsi untuk mengambil *Package Front-end* yang

digunakan oleh penulis. Dalam pembangunan aplikasi, untuk membuat tampilan aplikasi menjadi lebih menarik penulis menggunakan *library* ionic-material. Pada tampilan utama di aplikasi ini penulis ingin menampilkan lokasi map, dengan dari itu penulis menggunakan *library* ngMap yang dapat dijalankan oleh Ionic.

```
[MACs-MacBook-Pro:infogempa azizsudrajat$ bower install ionic-material
```

Gambar 3.18 Instal *library* ionic-material

```
[MACs-MacBook-Pro:infogempa azizsudrajat$ bower install ngmap
```

Gambar 3.19 Instal *library* ngmap

Setelah mengunduh *library* yang dibutuhkan menggunakan bower, setiap *file* pada folder *library* yang telah diunduh harus melakukan *path file* kedalam *index.html* di dalam folder *www/templates/*. Apabila *file* CSS menggunakan *tag link* dan jika *file* JS menggunakan *tag script*.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
  <title>Informasi Gempabumi</title>
  <link href="https://fonts.googleapis.com/css?family=RobotoDraft:400,500,700,400italic" rel="stylesheet">
  <link href="lib/ionic/css/ionic.css" rel="stylesheet">
  <link href="lib/ionic-material/dist/ionic.material.min.css" rel="stylesheet" />
  <link rel="stylesheet" href="css/animate/animate.min.css" media="screen" title="no title" charset="utf-8">
  <link href="css/style.css" rel="stylesheet">
  <script src="lib/ionic/js/ionic.bundle.js"></script>
  <script src="lib/ionic-material/dist/ionic.material.min.js"></script>
  <script src="cordova.js"></script>
  <script src="js/app.js"></script>
  <script src="js/controllers/Menu.js"></script>
  <script src="js/controllers/SkalaIntensitasGempabumi.js"></script>
  <script src="js/controllers/SkalaMMI.js"></script>
  <script src="js/controllers/AntisipasiGempabumi.js"></script>
  <script src="js/services.js"></script>
  <script src="js/controllers/GempaController.js"></script>
  <script src="https://maps-api-ssl.google.com/maps/api/js?libraries=places"></script>
  <script type="text/javascript" src="lib/ngmap/build/scripts/ng-map.min.js"></script>
</head>
<body ng-app="app">
  <ion-nav-view></ion-nav-view>
  <div id="modal" class=""></div>
</body>
</html>
```

Gambar 3.20 *Path link library* yang digunakan

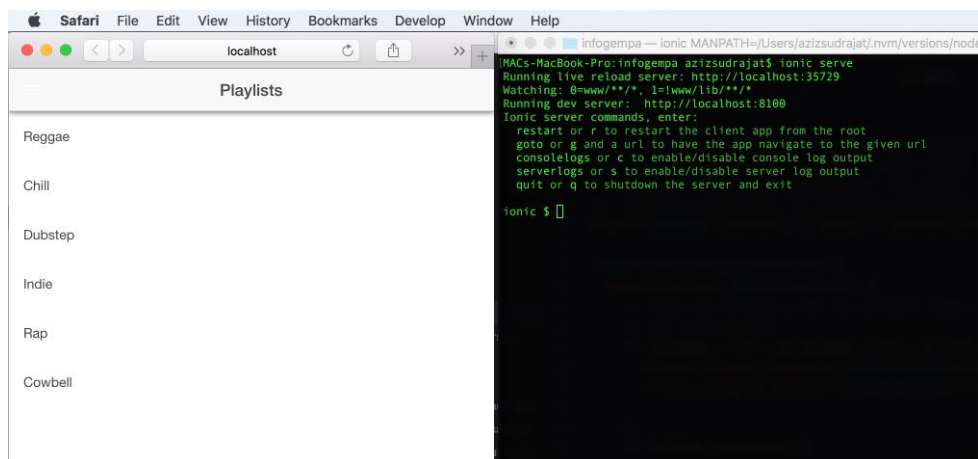
Pada Angular setiap pemberian *library* baru, Angular harus mengetahui tentang *library* tersebut. Angular terdapat *module* untuk mendefinisikan aplikasi

pada *tag* HTML dengan perintah *ng-app*. Setiap pembuatan *controllers* atau *library* baru, nama *controllers* atau *library* tersebut harus dimasukan kedalam *module app* Angular.

```
var app = angular.module('app', ['ionic', 'ionic-material', 'app.services', 'ngMap']);
```

Gambar 3.21 *Module* pada Angular

Setelah sudah melakukan *path file* pada *index.html* dan memasukan nama *library* ke dalam *module* Angular, terminal, untuk mencoba *project* Ionic menggunakan coba untuk menjalankan Ionic. Konfigurasi Ionic selalu didalam perintah *ionic serve*.



Gambar 3.22 Ionic pada web browser

Ionic tidak hanya dijalankan di dalam sistem operasi Android, akan tetapi aplikasi Ionic terbentuk oleh HTML, CSS dan JS. Jadi aplikasi Ionic dapat dibuka didalam *web browser*. Tampilan diatas adalah *sidemenu default* dari Ionic dan ada beberapa perintah untuk menjalankan Ionic Server.

Perintah *restart* atau *r* untuk mengulang aplikasi *client* dari folder *root*, *goto* atau *g* untuk pindah ke lain tampilan dengan menggunakan URL, *consolelogs* atau *c* untuk melihat kerja dari Ionic dan melihat *error* dari aplikasi yang dibuat, *quit* atau *q* untuk menutup aplikasi Ionic yang sedang berjalan pada Ionic Server.

Pada pembangunan aplikasi informasi gempabumi, penulis memanfaatkan API yang sudah tersedia, pada Ionic menggunakan Angular *\$http* untuk menarik data JSON dari API.

```
angular.module('app.services', [])

.factory('services', ['$http', function($http){
  var baseUrl = 'https://bmkg-mufrizal.rhcloud.com';

  return {
    getGempa: function() {
      return $http.get(baseUrl + '/api/gempa');
    },
    getGempaDirasakan: function(){
      return $http.get(baseUrl + '/api/gempadirasakan');
    },
    getStasiun: function(){
      return $http.get(baseUrl + '/api/stasiun');
    }
  }
}]);

.service('BlankService', [function() {
}]);
```

Gambar 3.23 Penarikan data JSON dengan Angular

3.3.2. Pembuatan Menu

Pada sebuah aplikasi bergerak pada *smartphone* di Andorid, apabila aplikasi yang dibuat memiliki lebih dari satu tampilan, menu adalah tampilan yang paling berperan penting untuk mengatur perpindahan tampilan. Pada menu di aplikasi ini, terdapat beberapa menu, yaitu menu gempabumi terbaru, gempabumi terkini > 5 SR, gempabumi dirasakan, antisipasi gempabumi, skala intensitas gempabumi, skala mercalli, jaringan stasiun dan tentang aplikasi.

Bahasa pemrograman yang digunakan dalam membuat *view* di Ionic menggunakan HTML dan CSS. *Tag* HTML pada Ionic sudah ditentukan oleh pengembang Ionic, setiap *tag* elemen yang tidak mengikuti *tag* HTML memiliki fungsinya sendiri yang disediakan oleh Ionic, seperti *tag* dalam membuat menu pada *file menu.html* disajikan pada Gambar 3.24.

```

<ion-list>
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.GempaTerbaru">
    <i class="icon ion-map"></i>Gempabumi Terbaru
  </ion-item>
  <!-- submenu gempabumi terbaru -->
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.GempaTerkini">
    <i class="icon ion-android-star"></i>Gempabarur Terkini > 5 SR
  </ion-item>
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.GempaDirasakan">
    <i class="icon ion-android-star-half"></i>Gemapbumi Dirasakan
  </ion-item>
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.AntisipasiGempabumi">
    <i class="icon ion-ios-heart"></i>Antisipasi Gemapbumi
  </ion-item>
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.SkalaIntesitas">
    <i class="icon ion-ios-analytics"></i>Skala Intesitas Gempa
  </ion-item>
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.SkalaMMI">
    <i class="icon ion-ios-analytics-outline"></i>Skala MMI
  </ion-item>
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.JaringanStasiun">
    <i class="icon ion-location"></i>Jaringan Stasiun Gempa
  </ion-item>
  <ion-item class="item-icon-left" nav-clear menu-close ui-sref="app.Tentang">
    <i class="icon ion-help-circled"></i>Tentang
  </ion-item>
  <ion-item class="item-icon-left" id="fab" nav-clear menu-close ng-click="showConfirm()">
    <i class="icon ion-close-circled"></i>Keluar
  </ion-item>
</ion-list>

```

Gambar 3.24 Penggalan program untuk membuat menu

Pada setiap *ion-item* memiliki *ui-sref* yang berfungsi sebagai alamat untuk perpindahan alamat. Angular bekerja untuk melakukan *routing* antara menu dan halaman .html yang akan dipanggil dalam *routing*. Sistem kerja Angular dalam *routing* dapat disebut sebagai *HTML templating*. Pada Ionic untuk melakukan *routing* menggunakan *library* angular-ui-route.

Pada *routing* Angular, terdapat *state* untuk *route* ke tujuan halaman yang sudah ditentukan oleh *ui-sref* di dalam *file* .html di *www/tempaltes*. Karena Ionic menggunakan bahasa *website*, pada *route* terdapat *Url Address Bar* yang berguna ketika Ionic Server dijalankan terbuka didalam *browser*. Cara penggunaan *route* Angular Ionic terdapat pada *file menu.html* atribut pada *tag* HTML yaitu *ui-sref*. pada *routing* halamn menggunakan library angular-ui-router, *library* tersebut milik Angular dalam melakukan perpindahan halaman yang biasa disebut dengan *HTML templating*. Penulisan kode program *route* pada *file* app.js terdapat pada Gambar 3.25.

```

app.config(function ($stateProvider, $urlRouterProvider) {
  $stateProvider

    .state('app', {
      url: '/app',
      abstract: true,
      templateUrl: 'templates/menu.html',
      controller: 'Menu'
    })

    .state('app.GempaTerkini', {
      url: '/GempaTerkini',
      views: {
        'menuContent': {
          templateUrl: 'templates/GempaTerkini5SR.html',
          controller: 'GempaTerkiniController'
        }
      }
    });
  // if none of the above states are matched, use this as the fallback
  $urlRouterProvider.otherwise('/app/GempaTerbaru');
});

```

Gambar 3.25 Potongan *Route* pada sidemenu

Ionic menggunakan Angular untuk memberikan aktifitas-aktifitas pada setiap halaman .html yang biasa disebut sebagai *controller*. Pembahasan *controller* akan lebih terstruktur apabila dibahas bersamaan dengan *file* .html yang berada pada Gambar 3.25.

3.3.3. Pembuatan Halaman Gempabumi Terbaru

Pembuatan aplikasi informasi gempabumi memanfaatkan Angular data *Binding* untuk menerapkan JSON dari API kedalam tampilan HTML yang berada pada *www/tampalters/GempaTerbaru.html* dan menggunakan *library* ngMap untuk menampilkan data gempabumi terbaru kedalam peta dari Google Maps dengan memanfaatkan data *latitude* dan *longitude* yang berada didalam JSON.

Tampilan halaman gempabumi terbaru akan dibangun berdasarkan rancangan pada Gambar 3.5 dan akan muncul pertama kali ketika aplikasi dibuka setelah *bejalannya splashscreen* pada aplikasi. Sebelum membuat tampilan pada *file GempaTebaru.html* terlebih dahulu membuat *controllers* untuk *file* html tersebut, terdapat pada Gambar 3.26.

```

.controller('GempaTerbaruController', function($scope, services) {
  $scope.dataGempa = {};
  function getGempa() {
    services.getGempa().success(function(data) {
      $scope.dataGempa = data.gempa[0];
      $scope.longitude = data.gempa[0].point.coordinates.split(",")[0];
      $scope.latitude = data.gempa[0].point.coordinates.split(",")[1];
      console.log($scope.longitude);
    });
  }
  getGempa();
})

```

Gambar 3.26 *Controller* tampilan gempabumi terbaru

Pada Angular, *Controller* sangat berperan penting dan pada Ionic *Controller* sudah harus ditentukan per-*file* .html ketika *routing* halaman *Templates*. Data JSON diambil dari nama objek Gempa, dan memisahkan data *latitude* dan *longitude* dengan fungsi *split* pada Angular untuk memisahkan 2 *value* didalam 1 *string*.

Data yang digunakan pada *controller* GempaTerbaruController diambil menggunakan fungsi *get* yaitu *getGempa*. Pada *file* yang berada dalam *www/js/services.js* terdapat fungsi Angular *\$http* *get* data JSON dari API yang terdapat pada Gambar 3.23. Setelah mengkonfigurasi *controller* di GempaTerbaruController, pindah kepada *file* *GempaTerbaru.html* untuk membuat *view* pada menu gempabumi terbaru terdapat pada Gambar 3.27.

```

<ion-view view-title="Gempabumi Terbaru" style="background-color:#90CAF9;">
  <ion-content>
    <ng-map style="border-shadow:none;" zoom="6" center="[{ {{latitude}}, {{longitude}} ]">
      <marker position="[{{latitude}}, {{longitude}}]" title="{{dataGempa.Wilayah}}"
animation="Animation.BOUNCE"></marker>
    </ng-map>
    <div class="list card">
      <div class="item item-body" style="background-color:#fafafa;">
        <h2>{{dataGempa.Wilayah}}</h2>
        <p>Tanggal Dan Jam : {{dataGempa.Tanggal}},{{dataGempa.Jam}}</p>

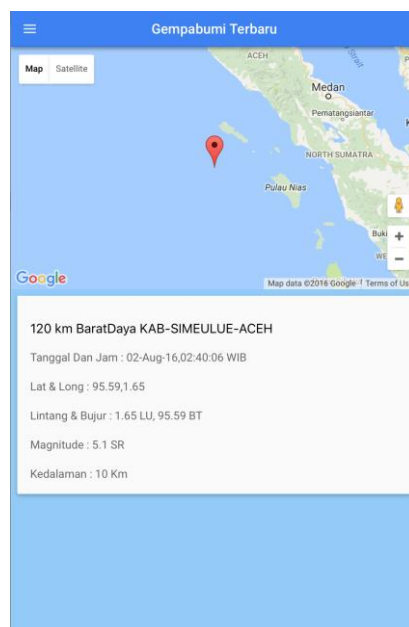
        <p>Lat & Long : {{dataGempa.point.coordinates}}</p>
        <p>Lintang & Bujur : {{dataGempa.Lintang}}, {{dataGempa.Bujur}}</p>
        <p>Magnitude : {{dataGempa.Magnitude}}</p>
        <p>Kedalaman : {{dataGempa.Kedalaman}}</p>
      </div>
    </div>
  </ion-content>
</ion-view>

```

Gambar 3.27 Potongan HTML gempabumi terbaru

Pembangunan tampilan gempabumi terbaru berhubungan dengan GempabumiTerbaruController dan *service.js* untuk dapat menggunakan Angular binding, karena Angular data binding dapat diaplikasikan dan dapat dilihat hasilnya apabila kode program sudah terhubung dengan benar, antara *view* dengan *controller* dan antara *controller* dengan *service API*.

Tampilan gempabumi terbaru dalam menampilkan data JSON BMKG menggunakan Google Maps agar lebih menarik. Setiap terdapat data terbaru pada informasi gempabumi di BMKG, secara otomatis pada tampilan gempabumi terbaru akan berubah mengikuti data dari BMKG, seperti pada Gambar 3.28.



Gambar 3.28 Tampilan gempabumi terbaru

3.3.4. Pembuatan Halaman Gempabumi Terkini > 5 SR

Tampilan halaman gempabumi terkini lebih besar dari 5 skala richter, akan menyajikan beberapa data gempabumi terbaru dengan magnitudenya melebihi 5 skala richter. Terdapat fitur *Search* untuk mencari lokasi gempabumi yang ingin diketahui pengguna dengan memanfaatkan fungsi filter pada Angular.

Sebelum membahas tentang *view* pada gempabumi terkini, penulis akan membahas *controller* yang berhubungan dengan tampilan gempabumi terkini yang terdapat pada gambar 3.23. Pada *controller* yang berhubungan dengan

tampilan *GempabumiTerkini5SR.html* sudah ditentukan pada *route* Angular, dapat di lihat pada Gambar 3.23.

```
.controller('GempaTerkiniController', function($scope, services) {

    $scope.dataGempa = {};

    function getGempa() {
        services.getGempa().success(function(data) {
            $scope.dataGempa = data.gempa;
            $scope.longitude = data.gempa.point.coordinates.split(",");
            $scope.latitude = data.gempa.point.coordinates.split(",");
            console.log($scope.longitude);
        });
    }

    getGempa();

})
```

Gambar 3.29 *Controller* tampilan gempabumi terkini

Penggalan pada Gambar 3.29 adalah *controller* pada tampilan gempabumi terkini yang lebih dari 5 skala richter. Berfungsi untuk mengambil objek pada API dan memecah *longitude* dan *latitude*, karena terdapat 2 *value* didalam 2 *string* penulis ingin menampilkan didalam tag html yang berbeda, jadi fungsi *split* digunakan untuk kasus ini.

Pada Angular *scope* adalah *super variable* dan *variable* pengikat yang menjalankan *viewcontroller* atau data *binding*. Data gempabumi terdapat pada *variable* *dataGempa*, lalu dibuatlah fungsi dengan nama *getGempa*. Pada fungsi memanggil *service* dan memecah *latitude* dan *longitude* dengan *split*. Hal itu dilakukan karna ada 2 *value* didalam 1 *string* pada objek JSON.

Setelah *controller* sudah benar, selanjutnya membuat *view* untuk mengaplikasikan *controller* yang sudah dibuat pada tampilan gempabumi terkini. Pada Angular *controller* bukan penyebutan seluruh *string* pada objek, untuk mengambil data atau *transferring* data menggunakan JSON cukup hanya mengambil objek nya saja, untuk seluruh *string*, penulis sudah harus mengetahui nya setelah melihat API yang sudah tersedia. Selanjutnya pada *view* gempabumi terkini yang menggunakan HTML, cukup mendefinisikan *variable* pada *scope* Angular dengan nama *string* pada objek, seperti pada Gambar 3.30.

```

<ion-view view-title="Gempabumi Terkini > 5 SR" style="background-color:#90CAF9;">
  <ion-content>
    <div class="list">
      <div class="item item-input-inset">
        <label class="item-input-wrapper">
          <i class="icon ion-ios-search"></i>
          <input type="text" placeholder="Cari Lokasi" ng-model="cari">
        </label>
      </div>
    </div>

    <div class="list card stable-bg" ng-repeat="g in dataGempa | filter: cari">
      <div class="item item-icon-left">
        <i class="icon ion-alert-circled"></i>
        <h2>{{ g.Wilayah }}</h2>
        <p>{{ g.Tanggal }},{{ g.Jam }}</p>
      </div>
      <div class="item item-body">
        <pre>Lat & Long    : {{ g.point.coordinates }}</pre>
        <pre>Lintang & Bujur : {{ g.Lintang }}, {{ g.Bujur }}</pre>
        <pre>Magnitude    : {{ g.Magnitude }}</pre>
        <pre>Kedalaman     : {{ g.Kedalaman }}</pre>
      </div>
    </div>
  </ion-content>
</ion-view>

```

Gambar 3.30 Kode HTML gempabumi terkini

Penggalan kode program diatas untuk menaruh beberapa *string* pada objek gempa yang akan ditampilkan pada *card stable-bg*, *card stable-bg* adalah nama *class* yang sudah disediakan oleh Ionic dan ionic-material. Fungsi *search* memanfaatkan *ng-model* dan filter milik Angular untuk menampilkan data gempa yang sesuai dengan *keyword* yang di masukan pada *list search*. Setelah itu filter di taruh pada *ng-repeat*, *ng-repeat* adalah fungsi untuk mengulang data.

Tampilan gempabumi terkini terhubung oleh GemapaTerkiniController terhubung dengan *service.js* untuk mengambil data pada API. Berbeda dengan bahasa pemrograman lainnya perulangan menggunakan *for*, *while* atau *do while*. Angular cukup efisien dalam penulisan kode program, penulis dengan hanya menggunakan *ng-repeat* yang disisipkan pada elemen HTML, maka dengan itu seluruh data pada API dapat terulang dan dapat ditampilkan pada *class card stable-bg* . Sintak ‘ | ‘ berfungsi sebagai pembatas pada kode program, kode program *filter: cari* harus sama *value* filter nya dengan *ng-model* yang berada pada *list search*, tampilan dapat dilihat pada Gambar 3.31.



Gambar 3.31 Tampilan gempabumi terkini > 5 SR

3.3.5. Pembuatan Halaman Gempabumi Dirasakan

Tampilan gempabumi dirasakan akan menginformasikan data-data gempabumi yang dirasakan oleh alat seismometer. Data gempabumi tidak hanya yang lebih dari 5 skala richter akan tetapi yang lebih kecil dari 5 skala richter pun akan ditampilkan apabila seismometer dapat merasakannya. Seismometer tidak hanya dapat merasakan gempabumi didaratan saja akan tetapi dapat merasakan getaran dari laut dan mengetahui kedalamannya.

Seperti sebelumnya, penulis akan membahas *controller* yang terbuhung dengan tampilan *GempabumiDirasakan.html*. Pada *controller* gempabumi dirasakan tidak jauh berbeda dan hampir seluruhnya sama, yang berbeda hanya nama *controller* dan nama objek yang dipanggil dari *file service.js*. Pada objek *Gempa* yang digunakan oleh tampilan gempabumi dirasakan sudah pasti *string* dan *value* berbeda dengan tampilan lainnya. Pada objek gempa terdapat 1 *string* yang memiliki 2 *value*, terdapat fungsi *split* pada Angular untuk memisahkan value tersebut. Kode program *controller* *GempaDirasakanController* dapat dilihat pada Gambar 3.32.


```

.controller('GempaDirasakanController', function($scope, services) {

    $scope.dataGempa = {};

    function getGempaDirasakan() {
        services.getGempaDirasakan().success(function(data) {
            $scope.dataGempa = data.Gempa;
            $scope.longitude = data.Gempa.point.coordinates.split(",");
            $scope.latitude = data.Gempa.point.coordinates.split(",");
            console.log($scope.longitude);
        });
    }

    getGempaDirasakan();

})

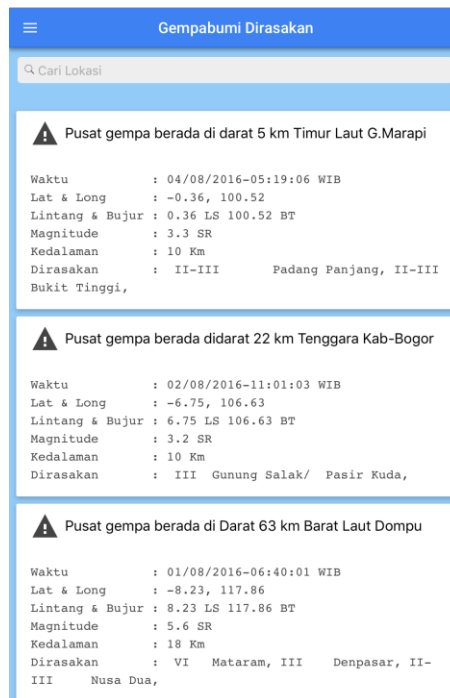
```

Gambar 3.32 Controller tampilan gempabumi dirasakan

Pada Gambar 3.32 adalah potongan *controller* yang terdapat pada *file GempaController.js* untuk menaruh beberapa *string* pada objek gempa yang akan ditampilkan pada *card stable-bg*, *card stable-bg* adalah nama *class* yang sudah disediakan oleh Ionic dan ionic-material. Fungsi *search* memanfaatkan *ng-model* dan filter milik Angular untuk menampilkan data gempa yang sesuai dengan *keyword* yang di masukan pada *list search*. Setelah itu filter di taruh pada *ng-repeat*, *ng-repeat* adalah fungsi untuk mengulang data.

Pada Angular *scope* adalah super *variable* dan *variable* pengikat yang menjalankan *viewcontroller* atau data *binding*. Data gempabumi terdapat pada *variable* *dataGempa*, lalu dibuatlah fungsi dengan nama *getGempaDirasakan*. Pada fungsi memanggil *service* dan memecah *latitude* dan *longitude* dengan *split*. Hal itu dilakukan karna ada 2 *value* didalam 1 *string* pada objek JSON.

Berbeda dengan bahasa pemrograman lain nya perulangan menggunakan *for*, *while* atau *do while*. Angular cukup efisien dalam penulisan kode program, penulis dengan hanya memanfaatkan *ng-repeat* seluruh data pada API dapat terulang dan dapat ditampilkan seluruh objek data pada *class card stable-bg*. Sintak ‘ | ‘ berfungsi sebagai pembatas pada kode program, kode program *filter: cari* harus sama *value* filter nya dengan *ng-model* yang berapa pada *list search*, tampilan dapat dilihat pada Gambar 3.33.



Gambar 3.33 Tampilan gempabumi dirasakan

3.3.6. Pembuatan Halaman Antisipasi Gempabumi

Pada tampilan antisipasi gempabumi akan memberikan informasi mengenai hal-hal yang dibutuhkan sebelum terjadinya gempabumi, hal-hal yang harus diketahui pengguna saat terjadinya gempabumi dan hal-hal apa saja yang harus dilakukan setelah terjadinya gempabumi. Tampilan antisipasi gempabumi dibuat sedikit lebih menarik dengan adanya gambar yang menggambarkan setiap nilai-nilai yang terkandung pada teks yang dijelaskan pada informasi antisipasi gempabumi.

Tampilan ini tidak memiliki aktifitas yang harus dilakukan oleh *controller* pada Angular, akan tetapi harus mencantumkan nama *controller* pada *route* yang terdapat pada Gambar 3.19 dan harus menuliskan *.controller* dengan nama *controller* yang harus sama pada *route* dan memberikan fungsi *scope*, karena *scope* adalah super *variable* untuk menjalankan *binding* atau *viewcontroller*. AntisipasiController dapat dilihat pada Gambar 3.34.

```
app.controller('AntisipasiController', function($scope, $stateParams) {

});
```

Gambar 3.34 *Controller* tampilan antisipasi gempabumi

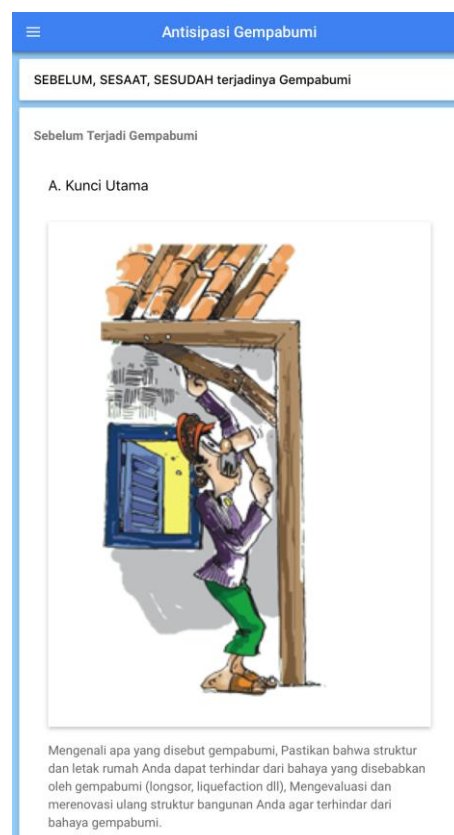
Penampilan pada Gambar 3.34 tidak ada yang menarik, karena tidak ada aktifitas pada tampilan antisipasi gempabumi. Tampilan antisipasi gempabumi hanya menampilkan teks dan gambar agar lebih menarik sebagai informasi untuk pengguna. Pada *file* AntisipasiGempabumi.html kode program lebih panjang karena seluruh informasi teks atau gambar ditaruh pada *tag* HTML pada *file* tersebut. Pada *file* gambar untuk tampilan antisipasi gempabumi terdapat pada *path* *img/antisipasi/sebelum/*, *img/antisipasi/sesat/* dan *img/antisipasi/sesudah/*.

Pada *view* untuk antisipasi gempabumi berbeda dengan tampilan sebelumnya, karena tidak sama sekali memanfaatkan fungsi-fungsi pada Angular. Seluruhnya menggunakan *tag* HTML yang sudah disediakan Ionic dan *tag* HTML dasar dalam membangun *website*, dapat dilihat pada Gambar 3.35.

```
<ion-view view-title="Antisipasi Gempabumi" style="background-color:#90CAF9;">
  <ion-content>
    <div class="card animated rubberBand">
      <div class="item item-text-wrap">
        <h3 class="has-subheader" style="color: #000;">SEBELUM, SESAAT, SESUDAH terjadinya
        Gempabumi</h3>
      </div>
    </div>
    <div class="card">
      <div class="item item-divider">
        Sebelum Terjadi Gempabumi
      </div>
      <div class="list-card">
        <ion-list>
          <ion-item>
            <div class="item">
              <h2>A. Kunci Utama</h2>
            </div>
            <div class="item item-body">
              
              <p>
                Mengenali apa yang disebut gempabumi, Pastikan bahwa struktur dan letak rumah Anda dapat
                terhindar dari bahaya yang disebabkan oleh gempabumi (longsor, liquefaction dll), Mengevaluasi dan
                merenovasi ulang struktur bangunan Anda agar terhindar dari bahaya gempabumi.
              </p>
            </div>
          </ion-item><!-- /item -->
        </div><!-- /list-card -->
      </div><!-- /card sebelum terjadi gempa -->
```

Gambar 3.35 Potongan HTML antisipasi gempabumi

Pada potongan kode program ditampilkan antisipasi gempabumi menggunakan nama *class* yang sudah disediakan oleh *documentation* Ionic di *website* resmi Ionic. Membuat header menggunakan *tag* HTML dasar seperti h1 sampai h6 tergantung tingkat besar *value font*, untuk membuat paragraf menggunakan *tag p*, untuk menampilkan gambar menggunakan *tag img* dan harus menentukan letak *file* gambar tersebut pada atribut *src* dan tampilan Antisipasi Gempabumi dapat dilihat pada Gambar 3.36.



Gambar 3.36 Tampilan Antisipasi Gempabumi

3.3.7. Pembuatan Halaman Skala Intensitas Gempabumi

Penginformasian tentang keadaan tingkatan getaran didalam bumi dan dampak akibat terjadinya gempabumi berada pada tampilan skala intensitas gempabumi. Tingkatan skala intensitas Gempabumi disimbolkan dengan angka romawi dan warna disetiap tingkatan, penginformasian pada setiap tingkatan akan diperjelas pada tampilan skala mercalli. Pada tampilan skala intensitas gempabumi

untuk mendapatkan animasi *expand* pada pengertian SIG, penulis memanfaatkan fungsi Angular seperti *ng-click* dan *ng-class*.

Pada penggunaan fungsi Angular tersebut, teks harus ditaruh dalam *controller* miliki tampilan skala intensitas gempabumi dan menaruh *variable items* pada *scope* dan didalamnya terdapat *string title* dan *text* yang berisikan teks, agar dapat menggunakan fungsi *binding* pada HTML, dapat dilihat pada gambar 3.37.

```
app.controller('SkalaIntensitasGempabumi', function ($scope, $stateParams, ionicMaterialInk, ionicMaterialMotion) {
    ionicMaterialInk.displayEffect();

    $scope.items = [{
        title: 'Skala Intensitas',
        text: 'SIG adalah Skala Intensitas Gempabumi. Skala ini menyatakan dampak yang ditimbulkan akibat terjadinya gempabumi. Skala Intensitas Gempabumi (SIG-BMKG) digagas dan disusun dengan mengakomodir keterangan dampak gempabumi berdasarkan tipikal budaya atau bangunan di Indonesia. Skala ini disusun lebih sederhana dengan hanya memiliki lima tingkatan yaitu I-V. SIG-BMKG diharapkan bermanfaat untuk digunakan dalam penyampaian informasi terkait mitigasi gempabumi dan atau respon cepat pada kejadian gempabumi merusak. Skala ini dapat memberikan kemudahan kepada masyarakat untuk dapat memahami tingkatan dampak yang terjadi akibat gempabumi dengan lebih baik dan akurat.'
    }];

    $scope.toggleItem= function(item) {
        if ($scope.isItemShown(item)) {
            $scope.shownItem = null;
        } else {
            $scope.shownItem = item;
        }
    };

    $scope.isItemShown = function(item) {
        return $scope.shownItem === item;
    };
});
```

Gambar 3.37 Controller tampilan Skala Intensitas Gempabumi

Pada Gambar 3.37 berfungsi untuk menaruh *value* pada string *title* dan *text* dan mendefinisikan animasi *expand* pada Angular, animasi yang akan digunakan pada tag HTML. Terdapat *variable toggleItem* yang berfungsi untuk menutup *expand* dan *variable toggleShow* yang berfungsi untuk menjalankan animasi *expand* pada tampilan skala intensitas gempabumi. Setelah membuat *controller* untuk skala intensitas gempabumi, selanjutnya mengaplikasikan pada *file*

SkalaIntesitasGempabumi.html dengan memanfaatkan sintak pada Angular untuk menjalankan animasi, dapat dilihat pada Gambar 3.38.

```
<div class="card animated rubberBand" ng-controller="SkalaIntesitasGempabumi">
  <ion-item ng-repeat="item in items" class="item item-text-wrap">
    <div ng-class="isItemShown(item) ? 'item-expand active' : 'item-expand inactive'">
      <h1>{{ item.title }}</h1>
      <p>{{ item.text }}</p>
    </div>
    <div ng-click="toggleItem(item)" class="item-expand-footer">
      <i ng-class="isItemShown(item) ? 'ion-ios-minus-outline' : 'ion-ios-plus-outline'"></i>
      {{ isItemShown(item) ? 'Tutup' : 'Buka' }}
    </div>
  </ion-item>
</div>
```

Gambar 3.38 Potongan program Skala Intesitas Gempabumi

Penulisan kode program untuk mengaktifkan fungsi *expand*, penulis harus mendefinisikan *controller* tampilan skala intensitas gempabumi dengan sintak *ng-controller*. Fungsi *ng-class* pada Angular untuk mengambil *class* yang akan diaktifkan untuk fungsi *isItemShow* yang sudah diberikan perintah pada *scope items*. Sintak *ng-click* berfungsi menjalankan perintah apabila *value* yang berada didalamnya mendapatkan *action click* dari pengguna, dan menjalankan fungsi *toggleItem(value_nama_class)*.

Penulisan kode program belum cukup sampai disitu untuk mendapatkan tampilan skala intensitas gempabumi yang diharapkan, karena masih ada beberapa informasi yang harus diberikan untuk pengguna. Terdapat enam tingkatan yang akan diinformasikan. Potongan kode program terdapat pada Gambar 3.39.

```
<div class="card animated jello">
  <div class="item item-button-right">
    SIG : I
    <button class="button button-stable">
      <i class="icon ion-ios-pulse-strong"></i>
    </button>
  </div>
  <div class="item item-body">
    <pre>Warna          : Putih</pre>
    <pre>Deskripsi Sederhana : TIDAK DIRASAKAN (Not Felt)</pre>
    <pre>Deskripsi Rinci    : Tidak dirasakan atau dirasakan hanya oleh beberapa orang tetapi terekam
oleh alat.</pre>
    <pre>Skala MMI          : I-II</pre>
    <pre>PGA(gal)           : < 2.9</pre>
  </div>
</div>
```

Gambar 3.39 Potongan program sig(lanjutan)

Pada potongan kode program ditampilkan skala intensitas gempa bumi menggunakan nama *class* yang sudah disediakan oleh *documentation* Ionic di *website* resmi Ionic. Membuat *header* menggunakan *tag* HTML dasar seperti *h1* sampai *h6* tergantung tingkat besar *value font*, untuk membuat paragraf menggunakan *tag* *p*, *tag* *pre* pada HTML adalah teks terformat yang mengikuti penulisan teks yang berada pada *tag* *pre* tersebut. Terdapat *tag* *button* untuk memberikan tombol pada tampilan dan terdapat *tag* *i*, pada Ionic dipakai untuk memberikan *icon*. Pada tampilan untuk memberikan *icon* dibutuhkan *tag* *i* dan *class* *icon*, lalu diisi nama *icon*.

Tampilan skala intensitas gempa bumi terdapat *expand item* dan animasi transisi *card panel* setiap data informasinya dengan memanfaatkan *library* *animate* CSS. Pengguna *library* *animate* CSS dengan menambahkan nama *class* *animated* ke dalam divisi *tag* HTML. Tampilan skala intensitas gempa bumi dapat dilihat pada Gambar 3.40.



Gambar 3.40 Tampilan skala intensitas gempabumi

3.3.8. Pembuatan Halaman Skala Mercalli

Pada pembuatan tampilan skala mercalli, untuk menginformasikan kepada pengguna satuan yang digunakan untuk mengukur kekuatan gempa bumi. Informasi ini berguna, apabila pengguna merasakan getaran yang berasal dari dalam bumi dan dapat mengetahui tingkat keseriusan geratan tersebut. Setelah itu pengguna dapat melihat antisipasi yang harus dilakukan yang dapat dilihat pada Gambar 3.36.

Pada tampilan skala mercalli untuk mendapatkan animasi *expand* pada pengertian skala mercalli, penulis memanfaatkan fungsi Angular seperti *ng-click* dan *ng-class*. Pada penggunaan fungsi Angular tersebut, teks harus ditaruh dalam *controller* miliki tampilan skala mercalli dan menaruh *variable* pada *scope* agar dapat menggunakan fungsi *binding* pada HTML, dapat dilihat pada gambar 3.42.

```
app.controller('SkalaMMI', function($scope, $stateParams) {

    $scope.items = [{
        title: 'Skala Mercalli',
        text: 'Skala Mercalli adalah satuan untuk mengukur kekuatan gempa bumi. Satuan ini diciptakan oleh seorang vulkanologis dari Italia yang bernama Giuseppe Mercalli pada tahun 1902. Skala Mercalli terbagi menjadi 12 pecahan berdasarkan informasi dari orang-orang yang selamat dari gempa tersebut dan juga dengan melihat serta membandingkan tingkat kerusakan akibat gempa bumi tersebut. Oleh itu skala Mercalli adalah sangat subjektif dan kurang tepat dibanding dengan perhitungan magnitudo gempa yang lain. Oleh karena itu, saat ini penggunaan Skala Richter lebih luas digunakan untuk mengukur kekuatan gempa bumi. Tetapi skala Mercalli yang dimodifikasi, pada tahun 1931 oleh ahli seismologi Harry Wood dan Frank Neumann masih sering digunakan terutama apabila tidak terdapat peralatan seismometer yang dapat mengukur kekuatan gempa bumi di tempat kejadian.'
    }];

    $scope.toggleItem= function(item) {
        if ($scope.isItemShown(item)) {
            $scope.shownItem = null;
        } else {
            $scope.shownItem = item;
        }
    };

    $scope.isItemShown = function(item) {
        return $scope.shownItem === item;
    };
});
```

Gambar 3.41 *Controller* tampilan skala mercalli

Pada Gambar 3.41 berfungsi untuk menaruh *value* pada *string title* dan *text* dan mendefinisikan animasi *expand* pada Angular, animasi yang akan

digunakan pada *tag* HTML. Setelah membuat *controller* untuk tampilan skala mercalli, selanjutnya mengaplikasikan pada *file* SkalaMMI.html dengan memanfaatkan sintak pada Angular untuk menjalankan animasi, dapat dilihat pada Gambar 3.42.

```
<div class="card animated rubberBand" ng-controller="SkalaMMI">
  <ion-item ng-repeat="item in items" class="item item-text-wrap">
    <div ng-class="isItemShown(item) ? 'item-expand active' : 'item-expand inactive'">
      <h1>{{ item.title }}</h1>
      <p>{{ item.text }}</p>
    </div>
    <div ng-click="toggleItem(item)" class="item-expand-footer">
      <i ng-class="isItemShown(item) ? 'ion-ios-minus-outline' : 'ion-ios-plus-outline'"></i>
      {{ isItemShown(item) ? 'Tutup' : 'Buka' }}
    </div>
  </ion-item>
</div>
```

Gambar 3.42 Potongan kode program skala mercalli

Pada penulisan kode program di Gambar 3.42 untuk mengaktifkan fungsi *expand*, penulis harus mendefinisikan *controller* tampilan skala mercalli dengan atribut *ng-controller* pada elemen *tag* HTML. Fungsi *ng-class* pada Angular untuk mengambil *class* yang akan diaktifkan untuk fungsi *isItemShow* yang sudah diberikan perintah pada *scope items*. Sintak *ng-click* berfungsi menjalankan perintah apabila *value* yang berada didalamnya mendapatkan *action click* dari pengguna, dan menjalankan fungsi *toggleItem(value_nama_class)*.

Penulisan kode program belum cukup sampai disitu untuk mendapatkan tampilan skala intensitas gempabumi yang diharapkan, karena masih ada beberapa informasi yang harus diberikan untuk pengguna. Terdapat enam tingkatan yang akan diinformasikan. Potongan kode program terdapat pada Gambar 3.43.

```
<div class="card animated zoomInDown">
  <div class="item item-avatar">
    
    <h2>I MMI</h2>
    <p>Tingkat 1</p>
  </div>

  <div class="item item-body">
    
    <p>Getaran tidak dirasakan kecuali dalam keadaan luarbiasa oleh beberapa orang</p>
  </div>
</div><!-- /card -->
</div><!-- /card -->
```

Gambar 3.43 Potongan kode program skala mercalli(lanjutan)

Pada potongan kode program *view* skala mercalli mengikuti nama *class* yang sudah disediakan oleh Ionic yang dapat dilihat pada *website* resmi Ionic. *Tag* *img* pada HTML berfungsi untuk menampilkan gambar, akan tetapi *tag* *img* memiliki atribut *src* untuk mengetahui lokasi gambar yang akan ditampilkan. *Tag* *h2* sama artinya dengan *h1*, *h3* sampai *h6* yang membedakan ukuran *font*-nya saja, *h1* memiliki ukuran *font* paling besar dan mengecil pada *tag* *h6*, dan *tag* *h* biasa disebut sebagai *header*.

Penulisan *class* *card* dan beberapa *class* yang lain nya memiliki *property* dan *value*. Apabila pengembang atau pembuat menggunakan *library* Ionic, beberapa *class* seperti *class* diatas sudah terdapat pada *file* *lib/ionic/ionic.css*. kerangka kerja Ionic sudah menyiapkan yang dibutuhkan dalam pembuatan aplikasi infomasi gempabumi. Beberapa *class* yang digunakan pada tampilan ini menggunakan *CSS Component* pada dokumentasi Ionic.

Pada *class* pemisah antara tingkatan menggunakan fungsi yang sudah ada pada Ionic yaitu *card images*. Penulisan kode program memiliki 2 *class* didalamnya, *class* *item* *item-avatar* yang berfungsi penulis dapat memberikan gambar kecil selayaknya profil pada inisialisasi. Tampilan dapat dilihat pada Gambar 3.44.



Gambar 3.44 Tampilan skala mercalli

3.3.9. Pembuatan Halaman Jaringan Stasiun Gempabumi

Tampilan jaringan stasiun gempabumi menginformasikan tentang lokasi-lokasi stasiun Meteorologi, stasiun Klimatologi, stasiun Geofisika dan balai besar BMKG. Data-data yang disajikan berupa alamat stasiun, *website*, nomer telpon dan tahun operasi. Tampilan ini memberikan informasi seputar lokasi stasiun BMKG, agar setidaknya pengguna mengetahui begitu banyak tempat untuk mendapatkan getaran bumi dan menginformasikan kepada masyarakat Indonesia.

Sebelum membahas tentang *view* pada gempabumi terkini, penulis akan membahas *controller* yang berhubungan dengan tampilan gempabumi terkini yang terdapat pada gambar 3.46. *Controller* yang berhubungan dengan tampilan GempabumiTerkini5SR.html sudah ditentukan pada *route* Angular, dapat di lihat pada Gambar 3.25.

```
.controller('StasiunGempaController', function($scope, services) {

    $scope.dataGempa = {};

    function getStasiun() {
        services.getStasiun().success(function(data) {
            $scope.dataStasiun = data.Stasiun;
            $scope.longitude = data.Stasiun.point.coordinates.split(",");
            $scope.latitude = data.Stasiun.point.coordinates.split(",");
            console.log($scope.longitude);
        });
    }

    getStasiun();

});
```

Gambar 3.45 *Controller* tampilan jaringan stasiun

Penggalan pada Gambar 3.45 adalah *controller* pada tampilan jaringan stasiun gempabumi. Berfungsi untuk mengambil objek pada API dan memecah *longitude* dan *latitude*, karena terdapat 2 *value* didalam 2 *string* penulis ingin menampilkan didalam *tag* HTML yang berbeda, jadi fungsi *split* digunakan untuk kasus ini.

Setelah *controller* sudah benar, selanjutnya membuat *view* untuk mengaplikasikan *controller* yang sudah dibuat pada tampilan jaringan stasiun gempa. Pada Angular *controller* bukan penyebutan seluruh *string* pada objek, untuk mengambil data atau *transferring* data menggunakan JSON cukup hanya

mengambil objeknya saja, untuk seluruh *string*, penulis sudah harus mengetahuinya setelah melihat API yang sudah tersedia. Selanjutnya pada *view* jaringan stasiun yang menggunakan HTML, cukup mendefinisikan *variable* pada *scope* Angular dengan nama *string* pada objek, seperti pada Gambar 3.46.

```
<ion-view title="Jaringan Stasiun Gempa" style="background-color:#90CAF9;">
  <ion-content padding="true">
    <div class="list card" ng-repeat="s in dataStasiun">
      <div class="item item-icon-left" style="background-color:#64B5F6;">
        <i class="icon ion-android-globe"></i>
        <h2>{{s.Type}}</h2>
      </div>
      <div class="item item-body" style="background-color:#fafafa;">
        <h3>Kode UAKPB : {{s.Kode_UAKPB}}</h3>
        <pre>Basis Wilayah : {{s.Bawil}}</pre>
        <pre>Tahun Operasi : {{s.TahunOperasi}}</pre>
        <pre>Lat & Long : {{s.point.coordinates}}</pre>
        <pre>Lintang : {{s.Lintang}}</pre>
        <pre>Bujur : {{s.Bujur}}</pre>
        <pre>Desa : {{s.Bujur}}</pre>
        <pre>Kecamatan : {{s.Kecamatan}}</pre>
        <pre>Kabupaten : {{s.Kabupaten}}</pre>
        <pre>Provinsi : {{s.Provinsi}}</pre>
        <pre>Alamat : {{s.Alat}}</pre>
        <pre>Telp : {{s.Telp}}</pre>
        <pre>Fax : {{s.Fax}}</pre>
        <pre>Kepala : {{s.Kepala}}</pre>
        <pre>Elevasi : {{s.Elevasi}}</pre>
      </div>
    </div>
  </ion-content>
</ion-view>
```

Gambar 3.46 Kode HTML tampilan jaringan stasiun

Penjelsan pada potongan kode HTML tampilam jaringan stasiun sama dengan penjelasan kode HTML pada tampilan lainnya. Mendefinisikan *controller* pada *ng-controller* Angular, memasukan *string* pada *variable scope* Angular dan menaruhnya pada *tag* pre HTML. Terdapat *tag* button untuk memberikan tombol pada tampilan dan terdapat *tag* i, pada Ionic dipakai untuk memberikan *icon*. Pada tampilan untuk memberikan *icon* dibutuhkan *tag* i dan *class icon*, lalu diisikan nama *icon*.

Terdapat atribut *ng-repeat* pada *tag* HTML yang merupakan fungsi dari Angular untuk mengulang data yang berisi pada objek dari API, hasil tampilan dapat dilihat pada Gambar 3.47.



Gambar 3.47 Tampilan Jaringan Stasiun

3.3.10. Pembuatan Halaman Tentang

Tampilan tentang berisi informasi mengenai *library* yang digunakan dalam pembangunan aplikasi. Pada pembuatan tampilan tentang, penulis tidak perlu mengisi kode pada *controller* karena tidak ada aktifitas apapun didalam tampilan tentang. Penulisan kode program menggunakan *class card item-icon-left* agar dapat menginformasikan *library* beserta *icon* dari *library* tersebut.

Pembuatan tampilan halaman tentang mengikuti rancangan yang sudah dibuat pada Gambar 3.12, tampilan terbangun dari bahasa HTML dan CSS yang yang tidak berbeda dengan tampilan lainnya dan terdapat animasi yang menggunakan *library animate.css*. Pada TentangController tidak berisi kode apapun dapat dilihat pada Gambar 3.48.

```
app.controller('TentangController', function($scope, $stateParams) {

});
```

Gambar 3.48 *Controller* halaman tentang

Pada kode program controller halaman tentang, terdapat pembelian *variable app* yang mendefinisikan *module* pada *app.js*. Selanjutnya mengenai kode pada *view* tentang menggunakan *class im-wrapper* yang berfungsi memberikan padding pada setiap sisi agar konten didalamnya menjadi lebih mendalam. Penggalan kode program dapat dilihat pada Gambar 4.49.

```
<ion-view view-title="Tentang" style="background-color:#90CAF9;">
  <ion-content class="im-wrapper">
    <div class="text-center">
      <h1 class="animated lightSpeedIn">Aziz Sudrajat</h1>
      <p class="animated lightSpeedIn">http://bmkg-mufrizal.rhcloud.com/</p>
    </div>
    <div class="card stable-bg animated wobble">
      <div class="item">
        <h2>Terbuat Oleh</h2>
      </div>
    </div>
    <div class="card animated bounceIn">
      <a class="item item-icon-left">
        <i class="icon ion-ionic"></i>
        Ionic 1.3
      </a>
    </div>
    <div class="card animated bounceIn">
      <a class="item item-icon-left">
        <i class="icon ion-ionic"></i>
        HTML5
      </a>
    </div>
    <div class="card animated bounceIn">
      <a class="item item-icon-left">
        <i class="icon ion-ionic"></i>
        CSS3
      </a>
    </div>
  </ion-content>
</ion-view view>
```

Gambar 3.49 Potongan program tentang

Pada potongan kode program tampilan tentang, terdapat 10 *card stable-bg*. Fungsi *card* sudah terdapat pada *class* Ionic dan mendapati *design* lebih menarik karena *library* ionic-material. Terdapat pula didalam *class* tersebut *animated wobble*, itu merupakan nama *class* pada *library animate.css* yang memberikan animasi pada *class card* tersebut.

Halaman ini merupakan tampilan terakhir pada aplikasi informasi gempabumi. Pembangunan yang terakhir pada aplikasi informasi gempabumi adalah membuat aksi menutup aplikasi menggunakan menu kembali dan tombol kembali pada *smartphone* pengguna. Pembahasan tersebut terdapat pada subab selanjutnya.

3.3.11. Pembuatan Tombol Keluar

Pada *smartphone* pengguna terdapat tombol kembali yang berfungsi untuk kembali pada halaman sebelumnya dan menutup aplikasi. Pada pembangunan aplikasi gempabumi ini Angular berperan penting untuk aktifitas ini, terdapat suatu metode utilitas yang dapat digunakan untuk mengambil informasi perangkat dengan perintah *\$ionicPlatform*.

Penulisan kode program untuk fungsi menutup aplikasi harus diketikan pada *app.run function* pada *app.js*, hal tersebut pendefinisian *state* fungsi yang dijalankan berdasarkan *module* Angular di Ionic. Penulisan kode program dapat dilihat pada Gambar 3.50.

```
$ionicPlatform.registerBackButtonAction(function(event) {
  if (true) { // your check here
    $ionicPopup.confirm({
      title: 'Keluar',
      template: 'Anda yakin ingin menutup aplikasi ini ?'
    }).then(function(res) {
      if (res) {
        ionic.Platform.exitApp();
      }
    })
  }
}, 100);
```

Gambar 3.50 Kode program BackButtonAction

Pada penulisan kode program di Gambar 3.50 menggunakan metode *registerBackButtonAction* yang berfungsi agar aplikasi Ionic dapat mengetahui aksi tombol kembali pada *smartphone* pengguna. Metode tersebut memiliki percabangan yang berfungsi apabila pengguna menekan tombol kembali maka aplikasi akan mengeluarkan *Popup* konfirmasi untuk menutup aplikasi.

Pada tampilan *sidemenu* terdapat menu keluar, berfungsi sama seperti *BackButtonAction* pada *smartphone* untuk menutup aplikasi. Menu keluar terdapat pada *sidemenu* atau *menu.html* yang memiliki *controller* dengan nama *MenuController*. Terdapat *div id* pada *menu.html*, lalu Angular mengambil *id* dan menunggu kejadian klik pada menu keluar. Apabila terjadi aktifitas klik pada menu keluar, aplikasi akan mengeluarkan *Popup* konfirmasi untuk menutup aplikasi. Kode program dapat dilihat pada Gambar 3.51.

```
app.controller('MenuController', function ($scope, $ionicPopup) {
  var fab = document.getElementById('fab');
  fab.addEventListener('click', function () {
    $scope.showConfirm = function() {
      var confirmPopup = $ionicPopup.confirm({
        title: 'Keluar',
        template: 'Anda yakin ingin menutup aplikasi ini ?'
      });
      confirmPopup.then(function(res) {
        if(res) {
          // console.log('You are sure');
          ionic.Platform.exitApp();
        } else {
          console.log('You are not sure');
        }
      });
    };
  });
});
```

Gambar 3.51 *Controller* tampilan *sidemenu*

Pada Gambar 3.15 mengambil elemen *id* pada HTML dengan nama *fab*, lalu pada *id fab* ditambahkan pendeteksi klik dan didalam nya terdapat fungsi *showConfirm*. Fungsi tersebut menggunakan fungsi *ionicPopup* untuk mnampilkan *alert* pilihan. Terdapat kondisi apabila pengguna memilih OK pada *alert* maka aplikasi akan tertutup, lalu apabila pengguna memilih *Cancel* maka aplikasi tidak tertutup.

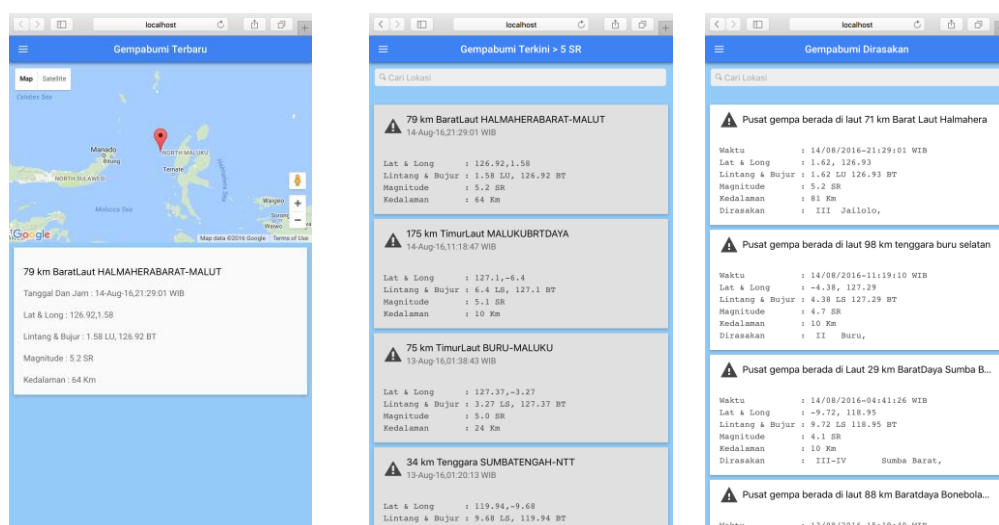
3.4. Uji Coba Aplikasi

Pada pengujian aplikasi ini dilakukan dalam tiga tahap uji coba, yaitu pengujian aplikasi menggunakan *Browser*, pengujian aplikasi menggunakan Android emulator dan pengujian pada pengguna. Hasil dari pengujian aplikasi menggunakan *Browser* dan Android emulator berjalan dengan baik. Pengujian dari beberapa pengguna dapat dilihat hasilnya pada Tabel 3.1.

3.4.1. Pengujian Aplikasi pada Browser

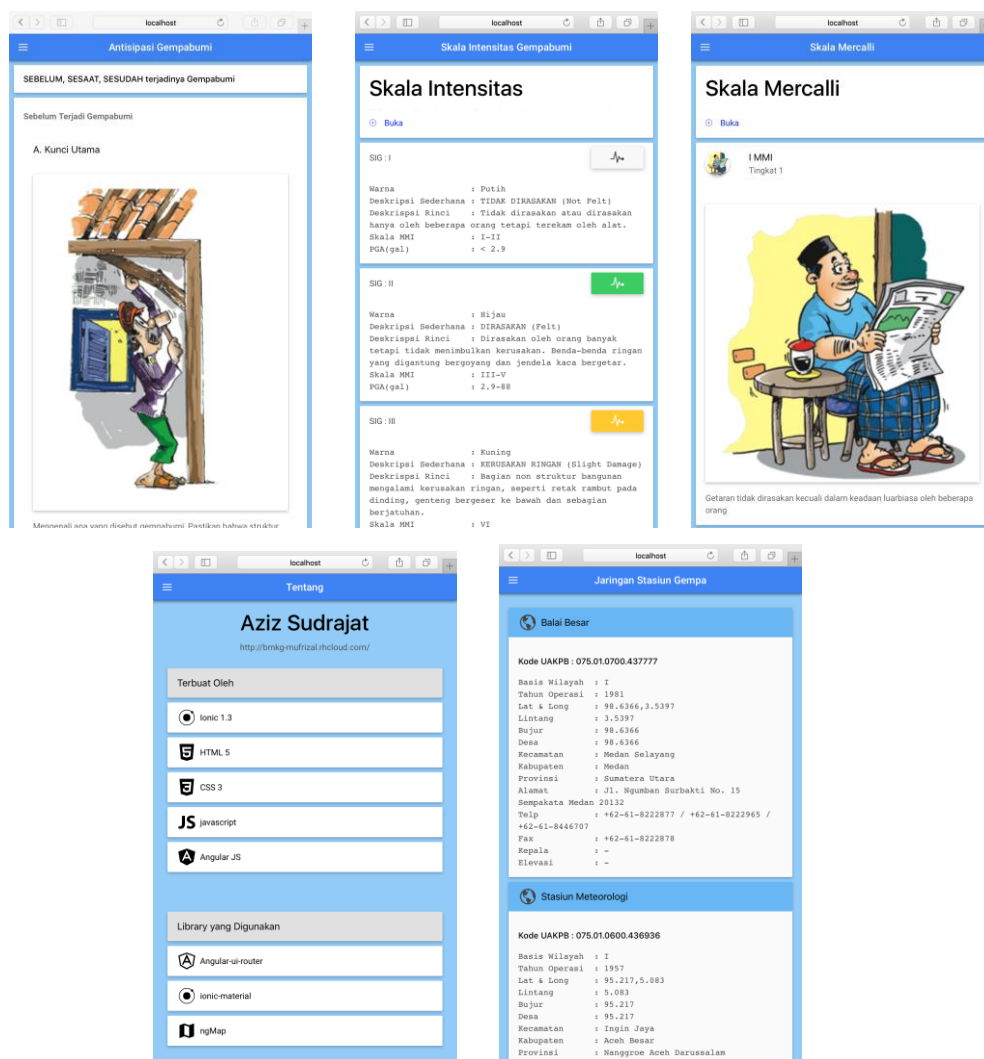
Pada pengujian aplikasi informasi gempa bumi pada *web browser* membutuhkan terminal yang berfungsi untuk melakukan konfigurasi Ionic Server. Ionic terbangun menggunakan bahasa pemrograman *website*, sehingga aplikasi informasi gempa bumi dapat dijalankan didalam *web browser*. Perintah untuk menjalankan Ionic pada *Browser* adalah *ionic serve*, seperti yang sudah terlihat pada Gambar 3.22.

Pengujian aplikasi informasi gempa bumi pada *Browser* memiliki kekurangan, yaitu tidak dapat melihat *splashscreen* dan tidak dapat menjalankan fungsi *BackButtonAction*. Aplikasi Ionic berjalan paling baik pada *Browser* dikarenakan terbangun dalam bahasa *website*. Tampilan aplikasi informasi gempa bumi terdapat pada Gambar 3.52.



Gambar 3.52 Tampilan Aplikasi pada Browser

Pada tampilan di Gambar 3.52 terdapat tampilan gempa bumi terbaru, gempa bumi terkini > 5 sr dan gempa bumi dirasakan. Pembangunan ketiga tampilan tersebut memanfaatkan API yang sudah ada dan terdapat animasi Ionic *loading* sebelum data tampil didalam *ion-view*. Tampilan aplikasi informasi gempa bumi tidak hanya itu masih terdapat beberapa tampilan lagi yang dapat dilihat pada Gambar 3.53.



Gambar 3.53 Tampilan Aplikasi pada Browser (lanjutan)

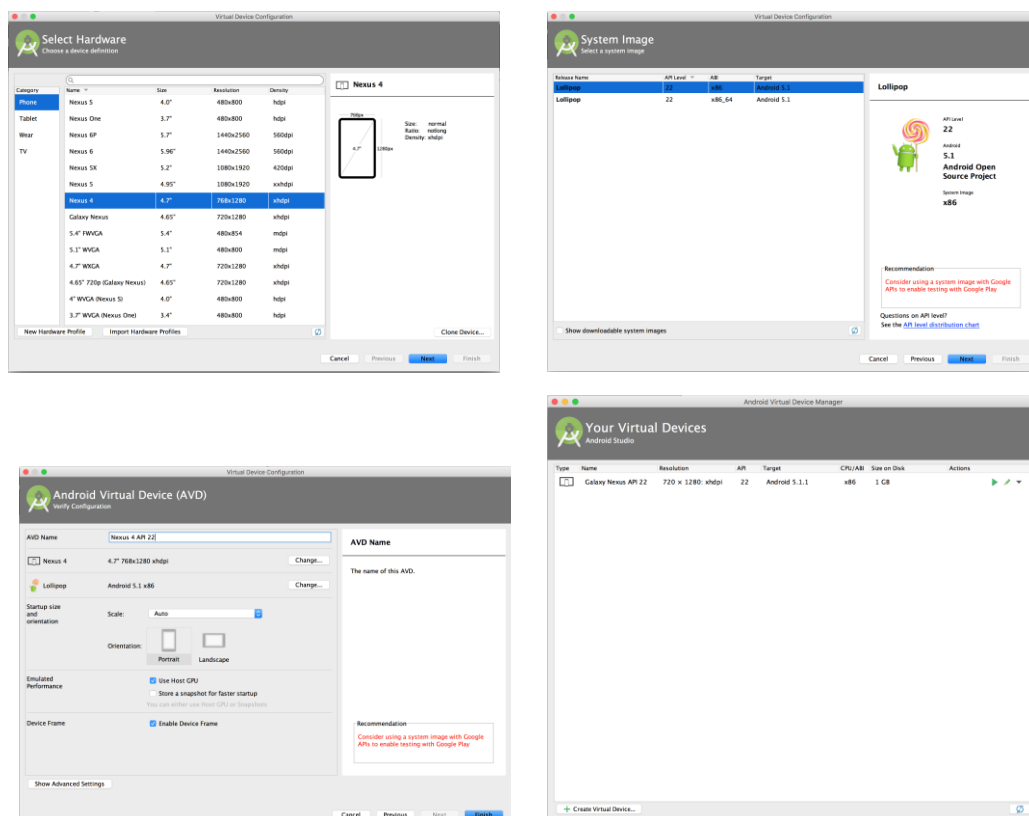
Pada Gambar 3.53 terdapat tampilan antisipasi gempa bumi, tampilan skala intensitas dan tampilan skala mercalli. Ketiga tampilan tersebut memberikan informasi dasar kepada pengguna mengenai gempa bumi. Suatu pengetahuan yang

harus diketahui pengguna, dalam pemberian informasi dasar ini diberikan pendekatan dengan gambar-gambar dan animasi pada setiap *card* nya.

Pada tampilan informasi gempabumi pada *Browser*, kedua tampilan tersebut adalah tampilan terakhir. Tampilan pertama yaitu jaringan stasiun gempabumi yang menginformasikan lokasi-lokasi BMKG pada setiap wilayah. Tampilan kedua yaitu tentang, tampilan tersebut memberitahukan *library* yang digunakan dalam pembangunan aplikasi.

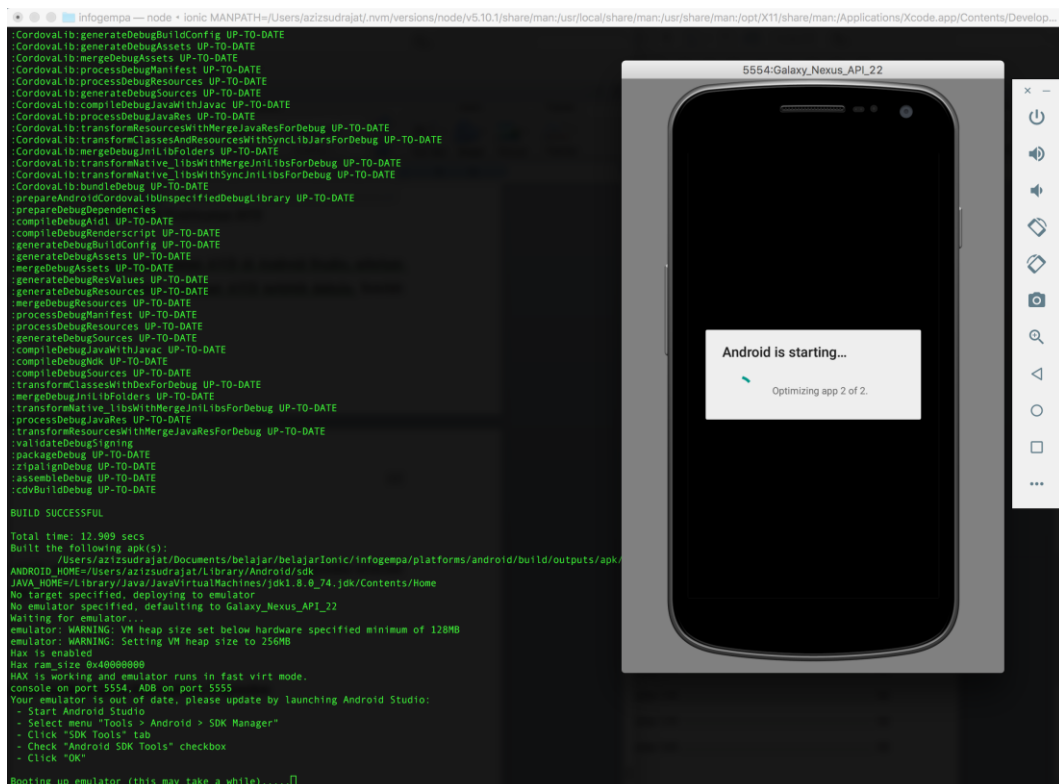
3.4.2. Pengujian Aplikasi pada AVD

Pada pengujian *Android Virtual Device* (AVD) tersebut membutuhkan Android Studio untuk membuat AVD. Penulis membuat AVD dengan sistem operasi Lollipop v51.1, resolusi 720 x 1280:xhdpi, ram 1GB dengan nama *device* Galaxi Nexus. Pembuatan AVD dapat dilihat pada Gambar 3.54.



Gambar 3.54 Pembuatan AVD

Pada Gambar 3.54 tahapan pembuatan AVD di Android Studio, sebelum menjalankan aplikasi Ionic diharuskan membuat AVD terlebih dahulu. Setelah sudah tebuat AVD, proses menjalankan aplikasi pada Ionic menggunakan terminal dengan Perintah *ionic run android*. Proses pengujian aplikasi menggunakan AVD dapat diliaht pada Gambar 3.55.



Gambar 3.55 Menjalankan AVD menggunakan terminal

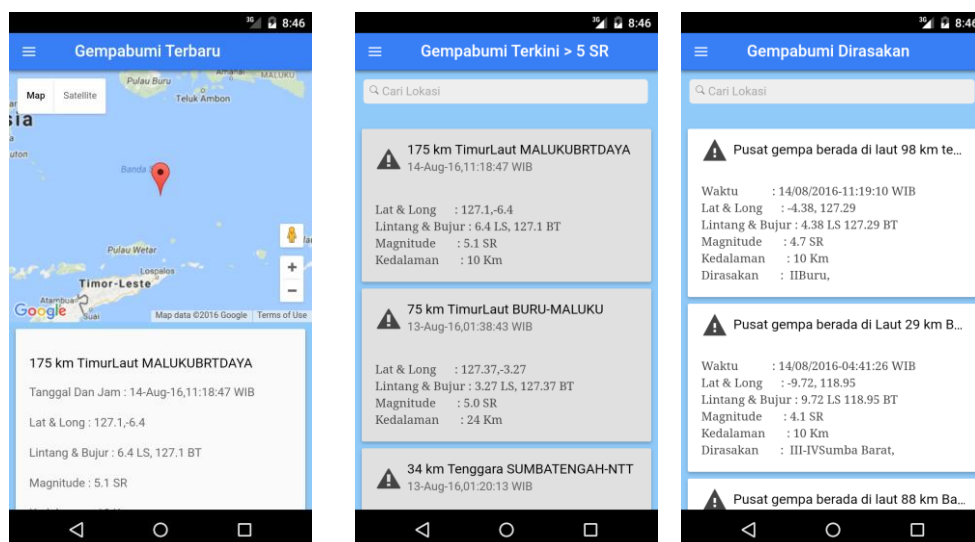
Setelah berhasil memanggil AVD menggunakan terminal, penulis dapat menguji aplikasi informasi gempabumi. Pengujian yang dilakukan adalah melihat tampilan gempabumi terbaru, gempabumi terkini, gempabumi dirasakan dan jaringan stasiun. Pada tampilan untuk mengetahui API dapat terpanggil atau tidak terdapat pada ionic *loading*. Apabila data dapat terpanggil ionic loading akan hilang tergantikan oleh data gempabumi dan sebaliknya.

Pengujian aplikasi sebelum masuk kedalam empat tampilan tersebut, aplikasi akan memunculkan *splashscreen*. tampilan *splashscreen* dapat dilihat pada Gambar 3.56.



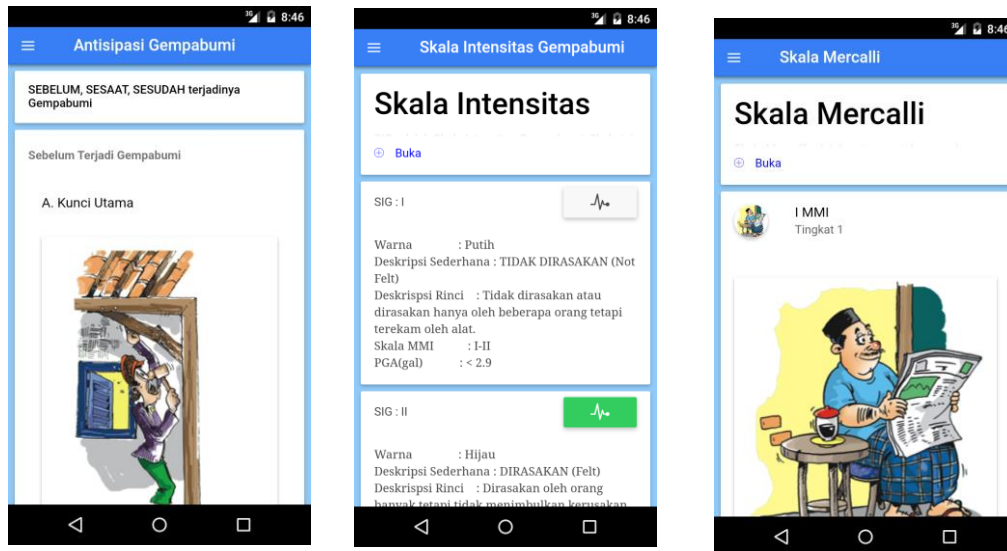
Gambar 3.56 *Splashscreen* pada AVD

Pada tampilan selanjutnya terdapat tampilan antisipasi gempabumi yang menginformasikan sebelum, sesaat dan setelah terjadinya gempabumi. Tampilan skala intensitas gempabumi menginformasikan satuan getaran pada gempabumi. Lalu selanjutnya tampilan skala mercalli memberikan informasi tingkatan pada getaran gempabumi. Tampilan dapat dilihat pada Gambar 3.57.



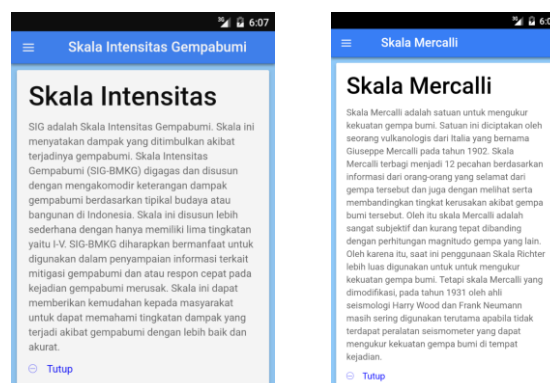
Gambar 3.57 Tampilan Aplikasi pada AVD

Pada Gambar 3.57 terdapat tampilan gempabumi terbaru, gempabumi terkini dan gempabumi dirasakan. Penguji cobaan tidak hanya pada tampilan itu saja, terdapat beberapa tampilan lagi yang harus diuji coba. Terdapat tiga tampilan yang akan ditampilkan pada Gambar 3.58.



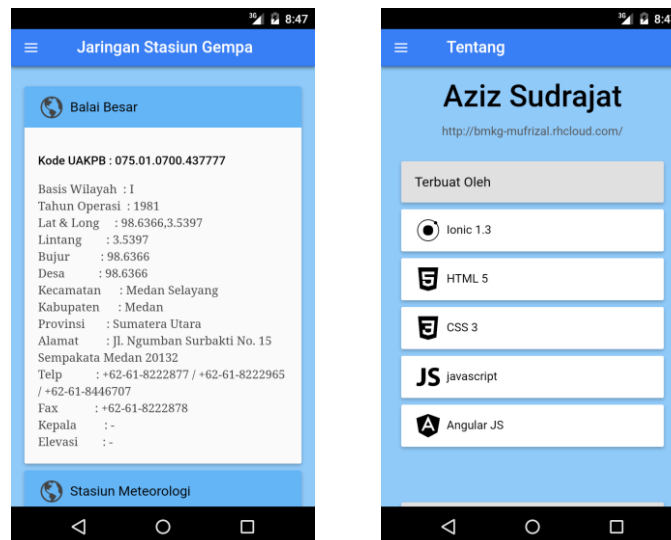
Gambar 3.58 Tampilan Aplikasi pada AVD (lanjutan 1)

Pada Gambar 3.58 terdapat tampilan antisipasi gempabumi, tampilan skala intensitas dan tampilan skala mercalli. Ketiga tampilan tersebut memberikan informasi dasar kepada pengguna mengenai gempabumi. Pada Tampilan skala intensitas gempabumi dan skala mercalli terdapat *expand item* untuk menjelaskan pengertian dari kata-kata tersebut. Tampilan terdapat pada Gambar 3.59.



Gambar 3.59 *Expand item* skala

Selanjutnya terdapat dua tampilan terakhir, tampilan jaringan gempa bumi untuk menginformasikan lokasi-lokasi pendeteksi gempa bumi. Tampilan tentang, berisi *library* yang digunakan dalam pembangunan aplikasi. Tampilan dapat dilihat pada Gambar 3.60.



Gambar 3.60 Tampilan Aplikasi pada AVD (lanjutan 2)

Pada tampilan selanjutnya, ketika pengguna menekan tombol *back* pada *smartphone* miliknya atau memilih menu keluar pada tampilan *sidemenu*, maka aplikasi akan mengeluarkan *Popup alert* seperti pada Gambar 3.61.



Gambar 3.61 *Popup alert* menutup aplikasi

3.4.3. Pengujian Aplikasi pada Pengguna

Penulis menganalisa hasil dari kuesioner yang telah diberikan kepada 21 (dua puluh satu) pengguna. Kuesioner yang diberikan adalah berupa pertanyaan seputar aplikasi Informasi Gempabumi yang telah pengguna pasang pada *smartphone* miliknya. Pada kuesioner terdapat sepuluh pertanyaan dan terdapat sepuluh hasil persentase dari pertanyaan-pertanyaan tersebut. 100 persen pengguna menilai tulisan pada aplikasi dapat terbaca, 95.5 persen pengguna menilai navigasi pada aplikasi mudah untuk digunakan. Penilaian selanjutnya 100 persen pengguna menilai pemilihan warna pada aplikasi ini sudah bagus, 90.5 persen pengguna menilai antarmuka aplikasi menarik dan 100 persen pengguna menilai aplikasi informasi gempabumi ini bermanfaat.

Penilaian selanjutnya 95.2 persen pengguna menilai aplikasi dapat dipasang pada *smartphone* pengguna, 95.2 persen pengguna menilai aplikasi dapat berjalan pada *smartphone* pengguna. Penilaian selanjutnya 95.2 persen pengguna menilai penempatan data gempabumi sudah bagus, 100 persen pengguna menilai aplikasi ini membantu LPNK BMKG dalam penginformasian gempa. Penilaian terakhir adalah mengenai animasi pada aplikasi, 100 persen pengguna menilai animasi pada aplikasi bagus. Hasil kuesioner tersebut dapat dilihat pada Tabel 3.1.

Tabel 3.1 Daftar Hasil Kuesioner

No	Pertanyaan	Setuju	Tidak Setuju
1	Apakah tulisan pada aplikasi informasi gempabumi ini dapat terbaca ?	100 %	0 %
2	Apakah navigasi pada aplikasi ini jelas ?	95.2 %	4.8 %
3	Apakah pemilihan warna pada aplikasi sudah bagus ?	100 %	0 %
4	Apakah antarmuka aplikasi menarik ?	90.5 %	9.5 %
5	Apakah informasi pada aplikasi ini bermanfaat ?	100 %	0 %
6	Apakah dapat dipasang pada <i>smartphone</i> responden ?	95.2 %	4.8 %
7	Apakah aplikasi dapat berjalan dengan baik ?	95.2 %	4.8 %
8	Apakah penempatan data gempabumi pada aplikasi sudah bagus ?	95.2 %	4.8 %
9	Apakah aplikasi dapat membantu LPNK BMKG dalam penginformasian gempa ?	100 %	0 %
10	Apakah animasi pada aplikasi bagus ?	100 %	0 %