

## SKRIPSI

# PEMBUATAN ULANG APLIKASI WSDC 2017 BALI DENGAN IONIC 6



Rajasa Cikal Maulana Solihin

NPM: 2017730084

PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS  
UNIVERSITAS KATOLIK PARAHYANGAN  
2022



**UNDERGRADUATE THESIS**

**RE-CREATION OF WSDC 2017 BALI APP WITH IONIC 6**



**Rajasa Cikal Maulana Solihin**

**NPM: 2017730084**

**DEPARTMENT OF INFORMATICS  
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES  
PARAHYANGAN CATHOLIC UNIVERSITY  
2022**



## ABSTRAK

*World Schools Debating Championships* (WSDC) merupakan sebuah turnamen debat Bahasa Inggris tahunan untuk tim-tim tingkat sekolah menengah yang mewakili berbagai negara. Pada tahun 2017, WSDC diselenggarakan di Bali, Indonesia. Untuk menunjang acara tersebut, diciptakan sebuah aplikasi WSDC 2017 Bali yang memiliki beberapa fitur seperti melihat jadwal acara, melihat pengumuman acara, melihat lokasi dan peta lokasi acara, dan fitur notifikasi. Aplikasi tersebut dibangun menggunakan Ionic Framework versi 3 dengan Cordova dan diimplementasikan menggunakan kerangka JavaScript Angular. Pada saat skripsi ini dibuat, Ionic Framework versi 3 sudah tidak lagi dikembangkan, dan Ionic Framework sudah mencapai versi 6. Maka dari itu, dibuatlah aplikasi pembaruan aplikasi WSDC 2017 Bali menggunakan Ionic Framework versi 6.

Aplikasi WSDC 2017 Bali dengan Ionic Framework 6 diimplementasikan dengan menggunakan kerangka JavaScript Angular. Lalu untuk menggunakan fitur *native* dari perangkat menggunakan Capacitor. Capacitor memiliki *plugins* untuk mengakses *native* API secara penuh. Beberapa *plugins* yang digunakan pada aplikasi ini yaitu: Geolocation API, Google Maps API, Splash Screen, dan Brower API. Dengan Capacitor pula maka aplikasi WSDC 2017 Bali yang semula ditulis menggunakan bahasa pemrograman web, dapat diimplementasikan untuk berjalan di dalam sistem operasi Android dan iOS. Namun pada skripsi ini hanya akan diuji pada perangkat Android.

Pengujian aplikasi WSDC 2017 Bali dilakukan terhadap beberapa responden dengan cara mengunduh, menginstal, dan membandingkan aplikasi WSDC 2017 Bali terdahulu dan terbaru. Dari pengujian tersebut didapatkan hasil yaitu aplikasi dapat berjalan dengan baik pada perangkat Android mulai dari versi 5.1 sampai dengan versi 11, berikut dengan fitur-fitur yang tersedia. Dengan begitu, aplikasi WSDC 2017 Bali dengan Ionic Framework versi 6 telah berhasil dibangun.

**Kata-kata kunci:** WSDC, Ionic Framework, UI Component, Capacitor, Cordova, Angular



## ABSTRACT

The World Schools Debating Championships (WSDC) is an annual English language debate tournament for high school-level teams representing various countries. In 2017, WSDC was held in Bali, Indonesia. To support the event, a WSDC 2017 Bali application was created which has several features such as viewing the event schedule, displaying announcements, viewing the location and map of the event location, and notification features. The application was built using the Ionic Framework version 3 with Cordova and implemented using the Angular JavaScript framework. At the time this thesis was written, the Ionic Framework version 3 was no longer developed, and the Ionic Framework had reached version 6. Therefore, an update for the WSDC 2017 Bali application will be made using the Ionic Framework version 6.

The WSDC 2017 Bali application with Ionic Framework 6 is implemented using the Angular JavaScript framework. Then to use the native features of devices that use Capacitor. Capacitor has a plugin to access the full native API. Some of the plugins used in this application are: Geolocation API, Google Maps API, Splash Screen, and Browser API. With the Capacitor, the WSDC 2017 Bali application, which was originally created using a web programming language, can be implemented to run on Android and iOS operating systems. However, in this thesis, it will only be tested on Android devices.

Testing of the WSDC 2017 Bali application was carried out on several respondents by downloading, installing, and comparing the previous and latest WSDC 2017 Bali applications. From these tests, the results are that the application can run well on Android devices starting from version 5.1 to version 11, along with the existing features. That way, the recreation of the WSDC 2017 Bali application with the Ionic Framework version 6 has been successfully built.

**Keywords:** WSDC, Ionic Framework, UI Component, Capacitor, Cordova, Angular



# DAFTAR ISI

<b>DAFTAR ISI</b>	<b>ix</b>
<b>DAFTAR GAMBAR</b>	<b>xi</b>
<b>1 PENDAHULUAN</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	2
1.3 Tujuan . . . . .	3
1.4 Batasan Masalah . . . . .	3
1.5 Metodologi . . . . .	3
1.6 Sistematika Pembahasan . . . . .	3
<b>2 LANDASAN TEORI</b>	<b>5</b>
2.1 WSDC 2017 Bali . . . . .	5
2.2 Angular . . . . .	6
2.3 Ionic Framework . . . . .	9
2.3.1 Native API . . . . .	9
2.3.2 UI Component . . . . .	16
2.3.3 Migrasi Ionic 3 ke Ionic 6 . . . . .	22
<b>3 ANALISIS</b>	<b>31</b>
3.1 Analisis Sistem Kini . . . . .	31
3.1.1 Skenario Pengguna . . . . .	32
3.1.2 Struktur Ionic 3 . . . . .	34
3.2 Analisis Sistem Usulan . . . . .	54
3.2.1 Analisis Kebutuhan . . . . .	54
3.2.2 Tantangan Pengembangan Sistem Usulan . . . . .	61
<b>4 PERANCANGAN</b>	<b>63</b>
4.1 Perancangan Kelas . . . . .	63
4.2 Perancangan Struktur HTML . . . . .	72
<b>5 IMPLEMENTASI DAN PENGUJIAN</b>	<b>75</b>
5.1 Implementasi . . . . .	75
5.1.1 Lingkungan Implementasi . . . . .	75
5.1.2 Hasil Implementasi . . . . .	75
5.2 Pengujian . . . . .	82
5.2.1 Pengujian Fungsional . . . . .	82
5.2.2 Pengujian Eksperimental . . . . .	83
<b>6 KESIMPULAN DAN SARAN</b>	<b>85</b>
6.1 Kesimpulan . . . . .	85
6.2 Saran . . . . .	85

<b>DAFTAR REFERENSI</b>	<b>87</b>
<b>A KODE PROGRAM</b>	<b>89</b>
A.1 Komponen Announcements . . . . .	89
A.2 Komponen Draw . . . . .	90
A.3 Komponen Home . . . . .	92
A.4 Komponen Info . . . . .	94
A.5 Komponen Result . . . . .	95
A.6 Komponen Schedule . . . . .	96
A.7 Komponen Venues . . . . .	98
A.8 Komponen Venues Map . . . . .	99

## DAFTAR GAMBAR

2.1 Aplikasi WSDC 2017 Bali Saat Ini pada Perangkat Android . . . . .	5
2.2 Aplikasi WSDC 2017 Bali Saat Ini pada Perangkat Android . . . . .	6
3.1 <i>Use Case Diagram</i> Aplikasi WSDC 2017 Bali . . . . .	31
3.2 Wireframe Aplikasi WSDC 2017 Bali . . . . .	37
3.3 Wireframe Aplikasi WSDC 2017 Bali . . . . .	44
4.1 Diagram Kelas Keseluruhan . . . . .	63
5.1 Tangkapan Layar Halaman Splash Screen Aplikasi WSDC 2017 Bali . . . . .	77
5.2 Tangkapan Layar Sidemenu Aplikasi WSDC 2017 Bali . . . . .	77
5.3 Tangkapan Layar Halaman Home Aplikasi WSDC 2017 Bali . . . . .	78
5.4 Tangkapan Layar Halaman <i>Announcements</i> Aplikasi WSDC 2017 Bali . . . . .	78
5.5 Tangkapan Layar Halaman <i>Draw</i> Aplikasi WSDC 2017 Bali . . . . .	79
5.6 Tangkapan Layar Halaman Info WSDC 2017 Bali . . . . .	79
5.7 Tangkapan Layar Halaman <i>Result</i> Aplikasi WSDC 2017 Bali . . . . .	80
5.8 Tangkapan Layar Halaman <i>Schedule</i> Aplikasi WSDC 2017 Bali . . . . .	80
5.9 Tangkapan Layar Halaman <i>Venues</i> Aplikasi WSDC 2017 Bali . . . . .	81
5.10 Tangkapan Layar Halaman <i>Venues Map</i> Aplikasi WSDC 2017 Bali . . . . .	81



<sup>1</sup>

## BAB 1

<sup>2</sup>

### PENDAHULUAN

#### <sup>3</sup> 1.1 Latar Belakang

<sup>4</sup> Dalam pengembangan suatu aplikasi seluler, setiap vendor perangkat lunak memiliki alat-alat  
<sup>5</sup> yang unik yang hanya dimiliki oleh vendor tersebut, yaitu *Software Development Kit* (SDK) [1].  
<sup>6</sup> Setiap sistem operasi memiliki perangkat *native* masing-masing, yang digunakan sebagai alat untuk  
<sup>7</sup> memungkinkan pengembang dalam pengembangan aplikasi seluler pihak ketiga. Namun karena  
<sup>8</sup> SDK yang digunakan berbeda-beda tergantung kepada jenis sistem operasi perangkat seluler, maka  
<sup>9</sup> terdapat permasalahan, yaitu tidak dapat membuat aplikasi untuk platform yang berbeda, namun  
<sup>10</sup> dengan baris kode yang sama. Untuk mengatasi masalah tersebut, terdapat sebuah teknologi yang  
<sup>11</sup> dikembangkan oleh Adobe yaitu Apache Cordova. Apache Cordova memungkinkan aplikasi berbasis  
<sup>12</sup> web seperti HTML, *Cascading Style Sheets* (CSS), dan Javascript, dikemas sebagai aplikasi seluler  
<sup>13</sup> *native*, dan juga menyediakan akses ke fitur perangkat.

<sup>14</sup> Aplikasi yang dibangun menggunakan Cordova disebut sebagai aplikasi *hybrid*, sebuah aplikasi  
<sup>15</sup> berbasis web yang berjalan di dalam sebuah tempat dari aplikasi *native* [1]. Aplikasi *hybrid* yang  
<sup>16</sup> berjalan merupakan aplikasi *native* yang menjalankan aplikasi berbasis web di tampilan web ponsel.  
<sup>17</sup> Aplikasi tersebut bersikap dan berperilaku normal seperti aplikasi pada umumnya, dan juga memiliki  
<sup>18</sup> keseluruhan akses terhadap perangkat.

<sup>19</sup> Salah satu kerangka kerja yang dapat digunakan untuk mengembangkan aplikasi *hybrid* adalah  
<sup>20</sup> Ionic Framework. Ionic Framework merupakan sebuah kerangka kerja *open source* lintas platform  
<sup>21</sup> yang memungkinkan untuk mengembangkan aplikasi *hybrid* yang bekerja pada berbagai macam  
<sup>22</sup> platform seluler seperti Android, iOS, dan Windows [2]. Ionic memiliki berbagai macam *front-end*  
<sup>23</sup> *library* dan *User Interface*(UI) *Components* yang digunakan untuk perancangan aplikasi menggu-  
<sup>24</sup> nakan teknologi web seperti HTML, CSS, dan Javascript. Dengan Ionic Framework, memungkinkan  
<sup>25</sup> sebuah aplikasi yang dapat berjalan menggunakan fitur *native* dari sebuah perangkat.

<sup>26</sup> Ionic Framework mendukung komunikasi dengan menggunakan Native API, yaitu pengembangan  
<sup>27</sup> aplikasi langsung terintegrasi ke dalam platform [3]. Selain menggunakan Cordova, Capacitor juga  
<sup>28</sup> merupakan Native API yang terintegrasi untuk menambahkan fungsionalitas ke dalam aplikasi Ionic  
<sup>29</sup> apapun. Capacitor merupakan penerus dari Cordova yang menyediakan akses ke perangkat *native*  
<sup>30</sup> dan fitur platform, serta untuk menyediakan satu set API untuk mengembangkan aplikasi seluler  
<sup>31</sup> secara hibrida, *Progressive Web Apps* berbasis web, dan aplikasi komputer berbasis Electron [4].

<sup>32</sup> Pada Ionic 5 ke atas, terdapat beberapa kerangka Javascript yang dapat diimplementasikan  
<sup>33</sup> menggunakan *framework* Ionic, seperti Angular, React, dan Vue. Angular pada awalnya diciptakan  
<sup>34</sup> oleh karyawan Google, Misko Hevert dan Adam Abrons pada tahun 2008, yang masih bernama

1 AngularJS dan dikembangkan dalam JavaScript [5]. Pada saat itu sebagian besar situs web  
2 menggunakan aplikasi multi-halaman, yaitu ketika pengguna mengklik tautan, maka browser harus  
3 mengambil dokumen HTML yang diminta dari server. Namun dengan Angular, digunakannya  
4 *Single-page Application* (SPA), yaitu ketika halaman awal dimuat, semua yang dibutuhkan untuk  
5 membuat dan menampilkan sebuah halaman diunduh, kemudian ditampilkan kedalam layar. Dengan  
6 begitu, browser tidak perlu melakukan *refresh* terhadap halaman tersebut [6]. React adalah *library*  
7 JavaScript *open source* untuk membangun antarmuka pengguna, dikelola oleh Facebook, dapat  
8 digunakan dalam berbagai skenario termasuk aplikasi iOS dan Android [5]. Vue merupakan  
9 *framework* progresif untuk membangun antarmuka pengguna untuk web, yang dapat digunakan  
10 baik untuk projek kecil dan untuk *Single-Page Applications* (SPAs) [5].

11 *World Schools Debating Championships* (WSDC) merupakan sebuah turnamen debat Bahasa  
12 Inggris tahunan untuk tim-tim tingkat sekolah menengah yang mewakili berbagai negara [7]. Pada  
13 awalnya, kompetisi universitas dunia akan diselenggarakan di Sydney pada bulan Juli 1988. Anggota  
14 Federasi Debat Australia menyadari bahwa tidak ada acara serupa untuk siswa sekolah menengah.  
15 Namun kejuaraan universitas dunia ini menunjukkan potensi yang sangat besar untuk kompetisi  
16 debat internasional yang melibatkan siswa dari seluruh dunia. Pada tahun 1991, kejuaraan diadakan  
17 di Edinburgh. Dan sejak saat itu nama World Schools Debating Championships digunakan dan  
18 berlangsung hingga saat ini.

19 WSDC yang diselenggarakan di Bali, Indonesia pada tahun 2017 memiliki sebuah aplikasi ber-  
20 nama WSDC 2017 Bali yang dikembangkan oleh PT DNArtworks Komunikasi Visual menggunakan  
21 *framework* Ionic 3 untuk menunjang acara tersebut. Terdapat beberapa fungsi penting di dalam  
22 aplikasi ini, diantaranya adalah jadwal untuk kegiatan peserta, berita tentang acara WSDC yang  
23 sedang berlangsung, pemberitahuan mengenai kegiatan acara kepada peserta, informasi lokasi dan  
24 peta lokasi kegiatan acara yang sedang berlangsung, dan notifikasi untuk peserta.

25 Aplikasi WSDC 2017 Bali yang dibangun pada tahun 2017 oleh PT DNArtworks Komunikasi  
26 Visual menggunakan Ionic versi 3 yang pada saat skripsi ini ditulis sudah tidak mendapat pembaruan  
27 lagi. Sedangkan Ionic Framework semakin berkembang dan pada saat skripsi ini dibuat sudah  
28 mencapai Ionic versi 6. Selain itu, aplikasi WSDC 2017 Bali untuk sistem operasi iOS telah  
29 diturunkan dari App Store dikarenakan tidak mengalami pembaruan dalam jangka waktu tertentu.  
30 Maka dari itu diperlukan pembaruan pada aplikasi WSDC 2017 Bali. Pada skripsi ini akan dibuat  
31 sebuah aplikasi pembaruan dari aplikasi WSDC 2017 Bali saat ini, dengan menggunakan *framework*  
32 Ionic versi 6<sup>1</sup>. *Framework* yang lebih baru memungkinkan perawatan yang lebih efisien, serta  
33 dukungan teknologi yang lebih terbarukan.

## 34 1.2 Rumusan Masalah

35 Rumusan masalah yang akan dibahas pada skripsi ini adalah bagaimana melakukan migrasi aplikasi  
36 Android WSDC 2017 Bali ke *framework* Ionic versi 6?

<sup>1</sup>Saat pertama kali topik skripsi ini dibuat, Ionic Framework terbaru adalah Ionic Framework versi 5. Sedangkan pada bulan Desember 2021 Ionic Framework mengeluarkan versi terbaru, yaitu versi 6. Pembaruan yang semula dari Ionic Framework versi 3 ke Ionic Framework versi 5, sekarang menjadi Ionic Framework versi 6 atas saran dan masukan dari dosen pembimbing dan dosen pengaji.

### 1 1.3 Tujuan

- 2 Tujuan yang ingin dicapai dari penulisan skripsi ini adalah melakukan migrasi aplikasi Android  
3 WSDC 2017 Bali ke *framework* Ionic versi 6.

### 4 1.4 Batasan Masalah

- 5 Dalam skripsi ini dibuat batasan-batasan masalah dalam pembuatan perangkat lunak. Batasan-  
6 batasan masalah yang ditetapkan adalah sebagai berikut:  
7 1. Aplikasi ini tidak akan memiliki fitur notifikasi, karena acara WSDC 2017 Bali sudah selesai  
8 dan tidak diperlukan kembali fitur notifikasi.  
9 2. Walaupun Ionic Framework mendukung pengembangan aplikasi untuk sistem operasi iO,  
10 aplikasi hanya akan diuji pada *platform mobile* berbasis android. Sedangkan pengujian untuk  
11 perangkat berbasis iOS akan dilakukan oleh tim yang berbeda.

### 12 1.5 Metodologi

- 13 Langkah-langkah yang dilakukan dalam skripsi ini adalah sebagai berikut:  
14 1. Melakukan studi mengenai *framework* Ionic versi 3 dan versi 6.  
15 2. Menganalisis aplikasi WSDC 2017 Bali.  
16 3. Mempelajari bagaimana cara melakukan migrasi Ionic versi 3 ke versi 6.  
17 4. Mendesain kelas aplikasi.  
18 5. Membangun aplikasi WSDC dengan *framework* Ionic versi 6.  
19 6. Melakukan pengujian dan eksperimen.  
20 7. Menulis dokumen skripsi.

### 21 1.6 Sistematika Pembahasan

- 22 Sistematika penulisan setiap bab pada skripsi ini adalah sebagai berikut:  
23 1. Bab Pendahuluan  
24 Bab 1 berisi latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, dan  
25 sistematika pembahasan yang digunakan untuk menyusun skripsi ini.  
26 2. Bab Dasar Teori  
27 Bab 2 berisi teori-teori yang digunakan dalam pembuatan skripsi ini. Teori-teori tersebut  
28 yaitu WSDC, Angular, Ionic Framework, Capacitor, Cordova, UI Components, dan Migrasi  
29 Ionic.  
30 3. Bab Analisis  
31 Bab 3 berisi analisis yang dilakukan pada skripsi ini, meliputi analisis sistem kini, analisis  
32 kebutuhan aplikasi WSDC 2017 Bali yang akan dibangun, serta permasalahan pembangunan  
33 sistem usulan.  
34 4. Bab Perancangan  
35 Bab 4 berisi perancangan aplikasi meliputi perancangan kelas beserta dengan diagram kelas,  
36 deskripsi kelas dan fungsinya, serta perancangan struktur HTML.

- 1      5. Bab Implementasi dan Pengujian  
2       Bab 5 berisi implementasi dan pengujian aplikasi meliputi lingkungan implementasi, hasil  
3       implementasi, pengujian fungsional, dan pengujian eksperimental.
  - 4      6. Bab Kesimpulan dan Saran  
5       Bab 6 berisi kesimpulan dari hasil pembangunan aplikasi ini dan saran untuk pengembangan  
6       selanjutnya.

1

## BAB 2

2

### LANDASAN TEORI

- 3 Pada bab ini akan menjelaskan dasar-dasar teori mengenai Ionic, berikut dengan cara untuk  
4 melakukan migrasi dari Ionic 3 ke Ionic 6. Akan dibahas pula aplikasi WSDC 2017 Bali saat ini,  
5 Angular, Native API berupa Capacitor dan Cordova, dan UI Components.

#### 6 2.1 WSDC 2017 Bali

- 7 Aplikasi WSDC 2017 Bali digunakan untuk menunjang keberlangsungan acara WSDC 2017 yang dise-  
8 lenggarakan di Bali, Indonesia. Aplikasi WSDC 2017 Bali dapat diunduh untuk sistem operasi *andro-*  
9 *id* melalui URL <https://play.google.com/store/apps/details?id=org.wsdc2017indonesia.app>. Aplikasi ini dibangun dan dikembangkan oleh PT DNArtworks Komunikasi Visual yang  
10 rilis di Play Store pada tanggal 30 Juli 2017, dengan versi terakhir adalah versi 1.1.2 yang rilis  
11 pada 1 Agustus 2017. Selain rilis pada perangkat *android*, aplikasi ini juga rilis untuk perangkat  
12 bergerak berbasis sistem operasi iOS. Namun saat saat skripsi ini dibuat, aplikasi yang dibuat pada  
13 tahun 2017 tersebut sudah diturunkan dari App Store pada perangkat berbasis sistem opearsi iOS,  
14 dikarenakan tidak mengalami pembaruan dalam jangka waktu tertentu. Untuk membuka dan  
15 memakai aplikasi WSDC 2017 Bali saat ini, pengguna tidak diperlukan *login* agar dapat mengakses  
16 seluruh fitur yang tersedia. Untuk kepentingan skripsi ini, peneliti memiliki akses ke dalam kode  
17 program aplikasi WSDC 2017 Bali.  
18

(a) Home page

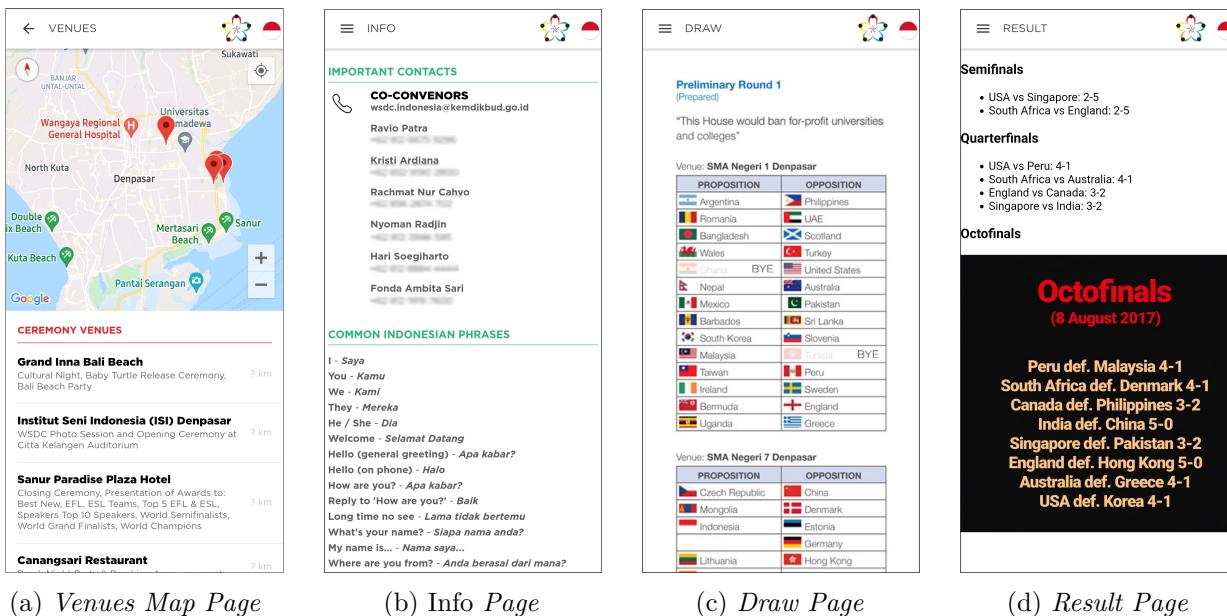
(b) Announcements Page

(c) Schedule Page

(d) Venues Page

Gambar 2.1: Aplikasi WSDC 2017 Bali Saat Ini pada Perangkat Android

- Fitur-fitur yang terdapat di aplikasi WSDC 2017 Bali saat ini yaitu :
1. *Home Page*: Pengguna dapat melihat pengumuman terbaru, dan *headline* dari berita-berita terkait acara WSDC 2017 Bali dengan tombol yang dapat diklik untuk melihat berita tersebut secara lebih detail (Gambar 2.1a).
  2. *Announcements*: Pengguna dapat melihat pemberitahuan tentang berjalannya acara WSDC 2017 Bali (Gambar 2.1b).
  3. *Schedule*: Pengguna dapat melihat jadwal acara WSDC 2017 Bali yang sudah diadakan (Gambar 2.1c).
  4. *Venues*: Pengguna dapat melihat berbagai macam lokasi acara WSDC 2017 Bali, mulai dari lokasi upacara, lokasi kompetisi, dan lokasi wisata edukasi (Gambar 2.1d). Masing-masing dari lokasi tersebut ditunjukkan dengan tanda merah pada peta dan dapat melihat jarak dari lokasi perangkat pengguna ke lokasi *venues* (Gambar 2.2a).
  5. *Info*: Pengguna dapat melihat informasi terkait dengan tim pengembang dari aplikasi WSDC 2017 Bali, kontak-kontak penting yang dapat dihubungi, dan kosa kata penting dalam Bahasa Indonesia (Gambar 2.2b).
  6. *Draw*: Pengguna dapat melihat melihat pembagian *venue* dan kubu proposisi atau oposisi dari hasil pengundian untuk para negara peserta WSDC 2017 Bali (Gambar 2.2c).
  7. *Result*: Pengguna dapat melihat informasi terkait hasil dari pertandingan pada semi final, perempat final, dan perdelapan final WSDC 2017 Bali (Gambar 2.2d).



Gambar 2.2: Aplikasi WSDC 2017 Bali Saat Ini pada Perangkat Android

## 2.2 Angular

Angular merupakan kerangka kerja Javascript terbuka yang dikembangkan oleh Google, dan merupakan penerus dari versi sebelumnya yaitu AngularJS [8]. Angular versi pertama dirilis pada tahun 2016 dengan nama Angular 2. Aplikasi Angular dapat dibangun dengan menggunakan JavaScript, atau TypeScript.

1 Angular dapat digunakan untuk membuat aplikasi Single-page-application (SPA), yaitu ketika  
 2 halaman awal dimuat, semua yang dibutuhkan untuk membuat dan menampilkan sebuah halaman  
 3 diunduh, kemudian ditampilkan kedalam layar [6]. Tampilan pada awalnya merupakan halaman  
 4 dengan kode-kode HTML yang belum utuh, kemudian tampilan halaman dibuat dan diunduh  
 5 untuk ditampilkan. Dengan begitu, *browser* tidak perlu melakukan penyegaran tampilan lagi untuk  
 6 menampilkan halaman baru.

7 Angular dapat dibangun dengan menggunakan JavaScript atau TypeScript. Namun, penggunaan  
 8 TypeScript saat ini menjadi lebih produktif dibandingkan dengan JavaScript [8]. TypeScript  
 9 mengikuti perkembangan terakhir dari ECMAScript, dan menambahkan *types*, *interface*, *decorators*,  
 10 *class member variables*, *generic*, *enum*, dan *keyword* seperti *public*, *protected*, dan *private*. TypeScript  
 11 juga dapat dimungkinkan untuk mendeklarasikan jenis tipe khusus yang dapat dikustomisasi. Maka  
 12 dari itu, Framework Angular sendiri ditulis menggunakan TypeScript.

13 Kelas TypeScript akan digunakan oleh Angular di dalam sebuah komponen. Komponen pada  
 14 Angular merupakan sebuah penyusun utama untuk aplikasi Angular. Setiap komponen yang ada  
 15 pada aplikasi, secara *default* memiliki *critical files* yang terdiri dari *file* HTML untuk mendeklarasi  
 16 halaman yang akan dimuat, *file* TypeScript, serta *file* css [9]. Di dalam komponen terdapat *module.ts*  
 17 yang merupakan NgModule dari sebuah komponen, *spec.ts* yang digunakan untuk menguji komponen,  
 18 dan *routing.module.ts* yang berisi definisi untuk bernavigasi antar bagian dalam aplikasi Angular.  
 19 Penjelasan untuk masing-masing *file* adalah sebagai berikut:

- 20 • *File routing.module.ts*

21 *File* ini digunakan untuk melakukan navigasi antar komponen. Untuk melakukannya, harus  
 22 melakukan *import* RouterModule dan Routes sehingga dapat memiliki fungsionalitas *routing* (Kode 2.1). Selanjutnya lakukan *import* untuk memberikan *router* suatu tempat untuk  
 23 dituju setelah mengonfigurasi rute.

```
25
26 1 import { RouterModule, Routes } from '@angular/router';
27 2 import { HeroesComponent } from './heroes/heroes.component';
```

Kode 2.1: Contoh *import* pada *routing.module.ts*

29 Setelah itu, lakukan konfigurasi *routes* (Kode 2.2). *Routes* akan memberi tahu Router tampilan  
 30 mana yang akan ditampilkan saat pengguna melakukan navigasi.

```
31
32 1 const routes: Routes = [
33 2   { path: 'heroes', component: HeroesComponent }
34 3 ];
```

Kode 2.2: Contoh Konfigurasi *Routes* pada *routing.module.ts*

- 36 • *File CSS*

37 *File* ini digunakan sebagai *template* CSS untuk sebuah komponen. Aplikasi angular ditata  
 38 dengan CSS standar, yang dapat menerapkan semua CSS stylesheets, *selectors*, *rules*, dan  
 39 *media queries* langsung ke aplikasi Angular.

- 40 • *File HTML*

41 *File HTML* pada Angular sama seperti HTML biasa, dan bisa ditambahkan dengan sintaks  
 42 Angular. HTML pada komponen digunakan untuk mendeklarasi halaman yang akan dimuat.

1     • *File* specs.ts

2       *File* ini digunakan untuk melakukan pengujian terhadap aplikasi Angular.

3     • *File* TypeScript

4       Di dalam TypeScript, terdapat sebuah kelas komponen. Kelas tersebut memiliki anotasi  
 5       dengan sebuah *decorator* (Kode 2.3) yang ditempatkan sesuai dengan komponen UI nya  
 6       berada. Komponen akan berisi *instance* dari service class yang diimplementasi dari *business*  
 7       *logic* tanpa UI. Sebuah service class merupakan implementasi dari *business logic*. Angular  
 8       akan menempatkan *services* ke dalam komponen atau ke dalam *services* lainnya dengan  
 9       menggunakan *dependency injection* (DI).

```
10
11 @Component({
12   ...
13 })
14 export class AppComponent{
15   ...
16 }
17
```

Kode 2.3: Anotasi Komponen dengan *Decorator*

18       Selain itu, di dalam TypeScript digunakan untuk mereferensikan HTML *template* dan CSS.  
 19       Pada setiap komponen terdapat HTML *template* yang berada di dalam komponen tersebut,  
 20       atau di dalam file yang berada di luar file komponen yang direferensikan dengan menggunakan  
 21       properti **templateUrl** di dalam komponen pada *file* TypeScript. File HTML *template* yang  
 22       terpisah dengan komponen, memberikan efek kode yang lebih bersih di dalam komponen nya.  
 23       Selain HTML *template*, file lain seperti file CSS dapat diletakkan terpisah dari file komponen  
 24       dengan menggunakan **styleUrls** untuk mereferensikan file CSS ke dalam komponen (Kode 2.4).  
 25       Lalu terdapat juga properti selector untuk mendefinisikan nama dari tag yang bisa digunakan  
 26       atau dipanggil oleh komponen lain.

```
27
28 @Component({
29   selector: 'app-search',
30   templateUrl: './search.component.html',
31   styleUrls: ['./search.component.css']
32 })
33 export class SearchComponent{
34   ...
35 }
```

Kode 2.4: Contoh Merefrensikan HTML dan CSS di Dalam Komponen

37     • *File* module.ts

38       Komponen-komponen yang ada akan dikelompokkan menjadi Angular modules, yaitu sebuah  
 39       kelas yang diberi **@NgModule()**. Angular module biasanya merupakan sebuah kelas kecil  
 40       yang tidak memiliki isi, kecuali menulis kode bootstrap secara manual ke dalam aplikasi.  
 41       Contohnya, ketika sebuah aplikasi merupakan turunan dari AngularJS yang lama, *decorator*  
 42       **@NgModule()** menampilkan semua komponen dan *artifacts* lain termasuk *services*, *directive*,  
 43       dan yang lainnya, maka semua itu harus disertakan ke dalam module (Kode 2.5).

```

1  1 @NgModule({
2   2   declarations: [
3   3     AppComponent
4   4   ],
5   5   imports: [
6   6     BrowserModule
7   7   ],
8   8   bootstrap: [AppComponent]
9   9 })
10 10 export class AppModule{
11 11   ...
12 12 }

```

Kode 2.5: Module dengan Komponen

15 Di dalam aplikasi Angular, terdapat beberapa komponen, seperti Parent Component, dan Child  
 16 Component. Parent Component dapat mengirimkan data ke properti Child Component-nya. Namun,  
 17 Child Component tidak tahu siapa yang mengirimkannya. Sedangkan Child Component juga dapat  
 18 mengirimkan data ke Parent Component, meskipun dia tidak tahu siapa Parent Component-nya.  
 19 Arsitektur seperti ini dapat membuat komponen menjadi mandiri dan dapat digunakan kembali.

## 20 2.3 Ionic Framework

21 Ionic Framework merupakan sebuah kerangka kerja *open source* lintas platform yang memungkinkan  
 22 untuk mengembangkan aplikasi hibrida yang bekerja pada berbagai macam platform seluler seperti  
 23 *android*, iOS, dan Windows [2]. Ionic memiliki berbagai macam *front-end library* dan komponen  
 24 *User Interface*(UI) yang digunakan untuk perancangan aplikasi menggunakan teknologi web seperti  
 25 HTML, CSS, dan Javascript, dengan integrasi untuk berbagai *framework* seperti Angular, React,  
 26 dan Vue. Saat pertama kali dibuat, Ionic menggunakan AngularJS. Namun, seiring saat Angular  
 27 versi 2 yang menggunakan Typescript dirilis, Ionic versi 2 dan selanjutnya menggunakan Angular.  
 28 Pada tahun 2019, Ionic mendukung penggunaan *framework* lain selain Angular, yaitu React dan  
 29 Vue. Di dalam Ionic, Angular digunakan untuk membangun aplikasi dan perutean, sehingga  
 30 aplikasi dapat sejalan dengan ekosistem Angular lainnya. Ionic menyediakan *toolkit* Angular untuk  
 31 membangun aplikasi dan terintegrasi dengan Angular CLI resmi yang menyediakan fitur khusus  
 32 untuk aplikasi Ionic Angular. Pada saat skripsi ini dibuat, Ionic versi terbaru adalah Ionic versi 6,  
 33 dengan dukungan penggunaan Angular versi 12 sampai dengan yang lebih baru.

### 34 2.3.1 Native API

35 Native API memungkinkan pengembangan aplikasi langsung terintegrasi ke dalam platform. Pe-  
 36 ngembang dapat membuat aplikasi pada perangkat *mobile* untuk dapat diimplementasikan ke  
 37 berbagai *platform*, seperti iOS dan Android, setelah pengembangan selesai di dalam *framework*  
 38 *native* tanpa perlu perubahan, dan tidak mempengaruhi performa dari aplikasi tersebut [3].

1 Ionic mendukung komunikasi dengan menggunakan Native API yang terintegrasi untuk menambahkan fungsionalitas ke dalam aplikasi Ionic apapun dengan menggunakan Capacitor atau Cordova.  
2 Dengan terpasangnya Ionic Native, maka aplikasi akan memiliki antar muka yang diperlukan untuk  
3 berinteraksi dengan salah satu *plug-in*, yaitu Capacitor atau Cordova.

5 **2.3.1.1 Capacitor**

6 Tujuan dari Capacitor adalah untuk menyediakan akses ke perangkat *native* dan fitur platform, serta  
7 untuk menyediakan satu set API untuk mengembangkan aplikasi seluler secara hibrida, *Progressive*  
8 *Web Apps* berbasis web, dan aplikasi komputer berbasis Electron [4]. Capacitor merupakan penerus  
9 dari Cordova, dengan tujuan untuk memungkinkan aplikasi web modern berjalan di semua platform  
10 utama. Capacitor juga mendapat dukungan terhadap banyak *plugin* Cordova.

11 Capacitor melakukan pendekatan secara hybrid yang memungkinkan pembuatan aplikasi  
12 seluler untuk beberapa platform dengan hanya dari satu basis kode [10]. Capacitor menampilkan  
13 aplikasi seluler dalam komponen *native* WebView yang dibungkus di dalam *native app*, meng-  
14 hasilkan komponen antarmuka pengguna yang memiliki tampilan dan rasa yang mirip dengan  
15 komponen *native* dari sistem operasi suatu perangkat. Capacitor menjadi sebuah pembungkus aplikasi  
16 web di dalam Android atau iOS *native*. Capacitor didasarkan pada aplikasi web yang dibuat  
17 dengan *framework* Ionic.

18 Capacitor memiliki *plugins* untuk mengakses *native API* secara penuh. *Plugins* yang dimiliki  
19 oleh Capacitor terbagi menjadi dua, yaitu:

20 1. *Official Plugins*

21 *Official Plugins* merupakan sekumpulan *plugin* resmi yang dikelola oleh tim Capacitor untuk  
22 menyediakan akses ke *native API* suatu perangkat. Terdapat beberapa *plugin* pada *Official*  
23 *Plugins*, diantaranya adalah sebagai berikut:

24 (a) Browser API

25 Browser API menyediakan kemampuan untuk membuka browser dalam aplikasi. Untuk  
26 menginstal Browser API dapat dilakukan melalui *command line* dengan perintah seperti  
27 pada kode 2.6.

28 1 `npm install @capacitor/browser`  
29 2 `npx cap sync`

30 Kode 2.6: Kode untuk Menginstal Browser API

32 Browser API memiliki beberapa *method*, yaitu:

- 33 • `open()`: *method* ini digunakan untuk membuka halaman *browser* dengan opsi yang  
34 sudah ditentukan.
- 35 • `close()`: *method* ini digunakan untuk menutup halaman *browser* yang sedang dibuka.  
36 *Method* ini hanya dapat digunakan pada web dan iOS.

- 1 • addListener('browserFinished', ...): *method* ini merupakan sebuah *listener* untuk  
2     *browser finished event* dan dipanggil ketika *browser* ditutup oleh pengguna. *Method*  
3     ini hanya dapat digunakan pada Android dan iOS.
- 4 • addListener('browserPageLoaded', ...): *method* ini merupakan sebuah *listener* untuk  
5     *page loaded event* dan aktif ketika URL diteruskan ke *method finished loading*. *Method*  
6     ini hanya dapat digunakan pada Android dan iOS.
- 7 • removeAllListeners(): *method* ini digunakan untuk menghapus semua *native listeners*  
8     untuk *plugin* ini.

9 (b) Geolocation API

10 Geolocation API menyediakan *method* untuk mendapatkan dan melacak posisi perangkat  
11 saat ini menggunakan *Global Positioning System* (GPS), dengan latitude dan longitude,  
12 juga informasi mengenai ketinggian, arah, dan kecepatan jika tersedia. Untuk menginstal  
13 Geolocation API dapat dilakukan melalui *command line* dengan perintah seperti pada  
14 kode 2.7.

```
15 1 npm install @capacitor/geolocation
16 2 npx cap sync
```

Kode 2.7: Kode untuk Menginstal Geolocation API

19 Pada perangkat Android, dibutuhkan izin untuk menggunakan Geolocation API 2.8. Izin  
20 tersebut ditambahkan ke dalam baris kode pada *file* AndroidManifest.xml. Izin tersebut  
21 digunakan untuk meminta data lokasi (*fine* dan *coarse*), dan izin untuk menggunakan  
22 GPS. Dengan menggunakan GPS, Geolocation API akan mengembalikan koordinat dari  
23 perangkat yang berupa latitude dan longitude. Contoh dari penggunaan Geolocation  
24 API untuk mengambil koordinat perangkat tertera pada kode 2.9.

```
25 1 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"
26   />
27 2 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
28   />
29 3 <uses-feature android:name="android.hardware.location.gps" />
```

Kode 2.8: *Permissions* Geolocation API pada Android

```
32 1 import { Geolocation } from '@capacitor/geolocation';
33 2
34 3 const printCurrentPosition = async () => {
35   4   const coordinates = await Geolocation.getCurrentPosition();
36   5
37   6   console.log('Current position:', coordinates);
38   7 }
```

Kode 2.9: *Permissions* Geolocation API pada Android

41 Untuk mengambil posisi dari perangkat dengan menggunakan *method* getCurrentPo-  
42 sition() seperti pada kode 2.9. Selain itu terdapat beberapa *method* lain yang dapat  
43 diimplementasikan, diantaranya yaitu:

- 1 • `watchPosition()`: digunakan untuk mendaftarkan fungsi handler yang akan dipanggil secara otomatis setiap kali posisi perangkat berubah.
- 2 • `clearWatch()`: digunakan untuk membatalkan pendaftaran fungsi handler yang sebelumnya diinstal menggunakan `watchPosition()`.
- 3 • `checkPermissions()`: digunakan untuk mengecek izin penggunaan lokasi.
- 4 • `requestPermissions()`: digunakan untuk meminta izin penggunaan lokasi.

7 (c) Splash Screen API

8 Splash Screen API digunakan sebagai penyedia *method* untuk menampilkan atau me-  
 9 nyembunyikan gambar *splash*. Untuk menginstal Splash Screen API dapat dilakukan  
 10 melalui *command line* dengan perintah seperti pada kode 2.10. Contoh penggunaan  
 11 Splash Screen dapat dilihat pada kode 2.11

```
12 1 npm install @capacitor/splash-screen
13 2 npx cap sync
```

Kode 2.10: Kode untuk Menginstal Splash Screen API

```
16 1 import { SplashScreen } from '@capacitor/splash-screen';
17 2
18 3 // Hide the splash (you should do this on app launch)
19 4 await SplashScreen.hide();
20 5
21 6 // Show the splash for an indefinite amount of time:
22 7 await SplashScreen.show({
23 8   autoHide: false
24 9 });
25 10
26 11 // Show the splash for two seconds and then automatically hide it:
27 12 await SplashScreen.show({
28 13   showDuration: 2000,
29 14   autoHide: true
30 15 });
```

Kode 2.11: Contoh Kode Penggunaan Splash Screen API

33 Secara *default*, Splash Screen diatur secara otomatis untuk disembunyikan dalam waktu  
 34 500ms. Selain itu, pada kondisi tertentu Splash Screen tidak sepenuhnya menutupi layar,  
 35 seperti pada bagian-bagian sudut-sudut layar. Splash Screen dapat mengatur warna  
 36 latar belakang, dibandingkan dengan menampilkan warna transparan saat Splash Screen  
 37 tidak sepenuhnya menutupi layar. Selain itu, Splash Screen juga dapat dibuat untuk  
 38 menampilkan *spinner* diatas Splash Screen.

39 Selain itu, terdapat beberapa *Official Plugins* lain yang dimiliki Capacitor, yaitu Action Sheet,  
 40 App, App Launcher, Camera, Clipboard, Device, Dialog, Filesystem, Google Maps, Haptics,  
 41 Keyboard, Local Notifications, Motion, Network, Push Notifications, Screen Reader, Share,  
 42 Status Bar, Storage, Text Zoom, dan Toast.

1    2. *Community Plugins*

2    *Community Plugins* merupakan sekumpulan *plugin* yang diciptakan oleh komunitas untuk  
 3    menambahkan fungsionalitas ke dalam aplikasi. Perbedaan dengan *Official Plugins* adalah  
 4    dimana tim Capacitor tidak secara resmi melakukan pemeliharaan terhadap *Community*  
 5    *Plugins*. Salah satu *plugin* yang diciptakan oleh komunitas adalah *plugin* Google Maps  
 6    yang menggunakan *native* Google Maps SDK. Untuk menginstal *plugin* Google Maps dapat  
 7    dilakukan melalui *command line* dengan perintah seperti pada kode 2.12.

```
8
9 1 npm i --save @capacitor-community/google-maps
10 2 npx cap sync
```

Kode 2.12: Kode untuk Menginstal *Plugin* Google Maps

12    Maps SDK merender *native element* (MapView) di belakang aplikasi web (WebView). Maka  
 13    dari itu, dibutuhkan *boundaries* yang dirender. Maka dari itu Maps SDK menggunakan  
 14    *native element* dibandingkan HTMLElement. Sebelum dapat menggunakan *plugin* ini, harus  
 15    dilakukan impor terlebih dahulu. Setelah mengimpor *plugin*, sebuah instance Maps sederhana  
 16    dapat diinisialisasi (Kode 2.13).

```
17
18 1 const initializeMap = async () => {
19   await CapacitorGoogleMaps.initialize({
20     key: "YOUR_IOS_MAPS_API_KEY",
21     devicePixelRatio: window.devicePixelRatio, // this line is very important
22   });
23
24   7 const element = document.getElementById("container");
25   const boundingRect = element.getBoundingClientRect();
26   try {
27     const result = await CapacitorGoogleMaps.createMap({
28       boundingRect: {
29         width: Math.round(boundingRect.width),
30         height: Math.round(boundingRect.height),
31         x: Math.round(boundingRect.x),
32         y: Math.round(boundingRect.y),
33       },
34     });
35     element.style.background = "";
36     element.setAttribute("data-maps-id", result.googleMap.mapId);
37
38     alert("Map loaded successfully");
39   } catch (e) {
40     alert("Map failed to load");
41   }
42 };
43
44 27 (function () {
45   initializeMap();
46 })();
```

Kode 2.13: Contoh Kode Penggunaan *Plugin* Google Maps

1 Capacitor juga dapat menambahkan platform *native* ke dalam proyek Ionic, baik untuk sistem  
 2 operasi Android seperti pada kode 2.14 maupun iOS seperti pada kode 2.15. Setelah itu, Capacitor  
 3 akan melakukan instalasi *package* platform Capacitor, dan menyalin *template native* platform ke  
 4 dalam proyek. Kemudian, Capacitor juga dapat melakukan pembuatan aplikasi *native* dengan  
 5 menyalin aset web ke dalam *native* platform dengan perintah seperti pada kode 2.16 untuk perangkat  
 6 berbasis iOS dan perintah pada kode 2.17 untuk perangkat berbasis Android. Setelah perintah  
 7 tersebut dieksekusi, maka Capacitor akan membuka IDE dari proyek *native*, seperti Xcode untuk  
 8 perangkat berbasis iOS dan Android Studio untuk perangkat berbasis Android. Dengan begitu,  
 9 maka aplikasi dapat berjalan secara *native* di dalam perangkat terkait.

```
10
11 ionic capacitor add android
12
```

Kode 2.14: Kode untuk Menambahkan Platform Android dengan Capacitor

```
13
14 ionic capacitor add ios
15
```

Kode 2.15: Kode untuk Menambahkan Platform iOS dengan Capacitor

```
16
17 ionic capacitor build ios
18
```

Kode 2.16: Kode untuk Membuat Aplikasi Capacitor Untuk Perangkat iOS

```
19
20 ionic capacitor build android
21
```

Kode 2.17: Kode untuk Membuat Aplikasi Capacitor Untuk Perangkat Android

### 22 2.3.1.2 Cordova

23 Cordova merupakan *framework open source* yang dapat membuat pengembang untuk menggunakan  
 24 teknologi seperti HTML, JavaScript, dan CSS untuk membangun aplikasi untuk perangkat bergerak  
 25 yang dapat berjalan pada beberapa sistem operasi *mobile* [11]. Cordova menyediakan antarmuka  
 26 antara WebView dan lapisan *native* pada perangkat [3]. Selain dapat bekerja pada dua platform  
 27 seluler Android dan iOS, Cordova juga dapat digunakan pada platform seluler seperti Windows Pho-  
 28 ne, Blackberry, dan FireOS.

29 Untuk mengonfigurasi proyek Cordova, saat ini dapat menggunakan *Command Line Tool* (CLI).  
 30 CLI membuat proyek dasar dan mengonfigurasinya agar berfungsi dengan platform seluler apa pun  
 31 yang didukung yang dapat digunakan. Cordova CLI juga dapat membuat pengembang memiliki  
 32 integrasi dan pengelolaan *plug-in*. Selain itu, Cordova CLI juga dapat mengkompilasi aplikasi  
 33 untuk berjalan pada simulator atau pada perangkat *native*. Serupa dengan Capacitor, Cordova  
 34 membuat pengembang dapat mengakses fitur *native* dari sebuah perangkat, seperti kamera, papan  
 35 ketik, dan geolokasi, menggunakan *plugin*. *Framework* Ionic telah terdapat berbagai macam Type-  
 36 Script *wrapper* untuk *plugins* Cordova. Untuk dapat menggunakan Cordova Plugins, yaitu dengan  
 37 memasang Cordova Plugins terlebih dahulu (Kode 2.18), dan memperbaruiunya ke versi terakhir  
 38 (Kode 2.19) yang dapat dilakukan melalui CLI. Setiap *plugins* memiliki dua komponen, yaitu kode  
 39 *native* (Cordova), dan kode TypeScript (Ionic Native).

```

1 npm install cordova-plugin-name
2 npx cap sync

```

Kode 2.18: Kode untuk Memasang Cordova Plugins

```

5 npm install cordova-plugin-name@2
6 npx cap update

```

Kode 2.19: Kode untuk Memperbarui Cordova Plugins

9 Sama seperti Capacitor, Cordova juga memiliki *plugins* yang menyediakan antarmuka JavaScript  
 10 ke komponen *native*. *Plugin* memungkinkan aplikasi untuk menggunakan kemampuan *native* dari  
 11 suatu perangkat. Salah satu *plugin* yang tersedia yaitu Google Maps. *Plugin* ini menghasilkan  
 12 MapView secara *native*, dan menempatkannya di bawah browser. MapView yang dihasilkan bukan  
 13 merupakan HTMLElements, maka tidak terkait dengan HTML. Sebelum dapat menggunakan *plugin*  
 14 Google Maps, dilakukan instalasi *plugin* untuk pertama kali pada *command line* 2.20. Selanjutnya  
 15 atur Google Maps API Key di dalam *file config.xml* 2.21.

```

16 cordova plugin add cordova-plugin-googlemaps
17
18

```

Kode 2.20: Kode untuk Menginstal *Plugin* Cordova Google Maps

```

19 <widget ...>
20   <preference name="GOOGLE_MAPS_ANDROID_API_KEY" value="(api key)" />
21   <preference name="GOOGLE_MAPS_IOS_API_KEY" value="(api key)" />
22 </widget>
23
24

```

Kode 2.21: Kode untuk Mengatur API Key untuk *Plugin* Cordova Google Maps

25 Cordova juga dapat menambahkan platform *native* ke dalam proyek Ionic dengan mengetikan  
 26 perintah seperti pada kode 2.22 untuk menambahkan platform Android, dan kode 2.23 untuk  
 27 menambahkan platform iOS. Untuk menjalankan proyek Ionic dengan Cordova dengan mengetikan  
 28 perintah seperti pada kode 2.24 untuk perangkat Android, dan kode 2.25 untuk perangkat iOS.  
 29 Dengan begitu, Cordova akan membuka IDE sesuai dengan perintah yang dijalankan.

```

30 ionic cordova platform add android
31
32

```

Kode 2.22: Kode untuk Menambahkan Platform Android dengan Capacitor

```

33 ionic cordova platform add iOS
34
35

```

Kode 2.23: Kode untuk Menambahkan Platform iOS dengan Capacitor

```

36 ionic cordova platform run android
37
38

```

Kode 2.24: Kode untuk Membuat Aplikasi Capacitor Untuk Perangkat Android

```

39 ionic cordova platform run iOS
40
41

```

Kode 2.25: Kode untuk Membuat Aplikasi Capacitor Untuk Perangkat iOS

### 2.3.2 UI Component

Dengan Angular, *framework* Ionic dapat menggunakan kemampuan Angular dalam memperluas kosakata HTML, yaitu menyertakan *tag* khusus untuk menciptakan seluruh rangkaian komponen [3]. Semua komponen memiliki awalan ion, sehingga dapat dikenali dalam markup. Sama seperti *tag* HTML standar, komponen Ionic juga dapat menerima berbagai macam atribut sebagai pengaturan dari *tag* tersebut, seperti mengatur id atau mendefinisikan kelas CSS tambahan. Terdapat beberapa komponen yang ada pada *framework* Ionic versi 6 [1]. Komponen-komponen tersebut yaitu:

- Action Sheet

Merupakan dialog yang menampilkan serangkaian opsi, yang muncul di atas konten aplikasi dan harus ditutup secara manual oleh pengguna sebelum pengguna dapat melanjutkan interaksi dengan aplikasi. Untuk menutup Action Sheet terdapat beberapa cara, termasuk mengetuk latar belakang atau menekan tombol escape di desktop.

- Alert

Alert merupakan dialog yang menampilkan informasi kepada pengguna, atau mengumpulkan informasi dari pengguna menggunakan input. Alert muncul di atas konten aplikasi, dan harus ditutup secara manual oleh pengguna sebelum pengguna dapat melanjutkan interaksi dengan aplikasi. Secara opsional, terdapat header, sub header, dan pesan yang ada pada Alert.

- Badge

Merupakan elemen *inline block* yang biasanya muncul di dekat elemen lain, berisi angka atau karakter lain, yang digunakan sebagai pemberitahuan bahwa ada item tambahan yang terkait dengan suatu elemen dan menunjukkan berapa banyak item yang ada. Penggunaan Badge dengan menggunakan *tag* `<ion-badge>` (Kode 2.26).

```
1 <ion-badge>99</ion-badge>
```

Kode 2.26: Potongan Kode Program dari Badge Component

- Button

Merupakan elemen yang dapat diklik, biasanya digunakan dalam formulir atau di mana pun yang membutuhkan fungsionalitas tombol. Button biasanya menampilkan teks, ikon, atau bisa juga keduanya. Button dapat pula menggunakan atribut untuk menampilkannya dengan penampilan tertentu. Penggunaan Button dengan menggunakan *tag* `<ion-button>` (Kode 2.27).

```
1 <ion-button>Default</ion-button>
```

Kode 2.27: Potongan Kode Program dari Button Component

- Card

Merupakan bagian standar dari tampilan antarmuka yang berfungsi sebagai titik masuk ke dalam informasi yang lebih detail. Card dapat menjadi satu komponen, tetapi sering kali terdiri dari beberapa header, judul, sub judul, dan konten. Penggunaan Card dengan menggunakan *tag* `<ion-card>` yang dapat berisi *header*, *subtitle*, *title*, dan *content* (Kode 2.28).

```

1 1 <ion-card>
2   <ion-card-header>
3     <ion-card-subtitle>Card Subtitle</ion-card-subtitle>
4     <ion-card-title>Card Title</ion-card-title>
5   </ion-card-header>
6
7   <ion-card-content>
8     Card Content
9   </ion-card-content>
10 </ion-card>
11

```

Kode 2.28: Potongan Kode Program dari Card Component

- Content

Komponen content merupakan penyedia area konten yang bisa digunakan untuk mengontrol area yang dapat digulir. Dalam satu tampilan, setidaknya terdapat satu buah content. Content juga dapat dimodifikasi padding, margin, dan lainnya menggunakan *global style* yang berada di CSS Utilites atau mengubahnya secara individual dengan menggunakan CSS. Penggunaan Content dengan menggunakan tag `<ion-content>` (Kode 2.29).

```

1 <ion-content
2   [scrollEvents]="true"
3   (ionScrollStart)="logScrollStart()"
4   (ionScroll)="logScrolling($event)"
5   (ionScrollEnd)="logScrollEnd()">
6     <h1>Main Content</h1>
7
8     <div slot="fixed">
9       <h1>Fixed Content</h1>
10    </div>
11 </ion-content>

```

Kode 2.29: Potongan Kode Program dari Content Component

- Date and Time Pickers

Datetime merupakan penampil antarmuka untuk pengguna memilih tanggal dan waktu. Terdapat kolom yang dapat digulir yang dapat digunakan untuk memilih tahun, bulan, hari, jam, dan menit secara individual. Komponen ini menampilkan nilai di dua tempat, yaitu di komponen `<ion-datetime>` (Kode 2.43), dan di antarmuka pemilih yang ditampilkan dari bawah layar.

```

1 <ion-datetime displayFormat="MM DD YY" placeholder="Select Date"></ion-
40 41 </ion-datetime>

```

Kode 2.30: Kode Program dari Datetime Component dengan Format Bulan-Hari-Tahun

- Grid

Grid digunakan untuk membuat tata letak kustom pada tampilan. Grid terdiri dari tiga bagian, yaitu *grid*, baris, dan kolom. Masing-masing kolom dapat diubah ukurannya menggunakan CSS. Grid digunakan dengan tag `<ion-grid>` (Kode 2.31).

```

1 1 <ion-row>
2   <ion-col size="6">
3     ion-col [size="6"]
4   </ion-col>
5   <ion-col>
6     ion-col
7   </ion-col>
8 </ion-row>

```

Kode 2.31: Potongan Kode Program dari Grid Component

- Infinite Scroll

Komponen Infinite Scroll memanggil sebuah action yang akan dilakukan ketika pengguna menggulir dengan jarak tertentu dari bawah atau atas halaman. Penggunaan Infinite Scroll dengan menggunakan tag `<ion-infinite-scroll>` (Kode 2.32).

```

15 1 <ion-infinite-scroll threshold="100px" (ionInfinite)="loadData($event)">
16 2   <ion-infinite-scroll-content
17 3     loadingSpinner="bubbles"
18 4     loadingText="Loading more data...">
19 5   </ion-infinite-scroll-content>
20 6 </ion-infinite-scroll>
21

```

Kode 2.32: Potongan Kode Program dari Infinite Scroll Component

- Icon

Icon merupakan komponen yang berupa gambar kecil, yang merepresentasikan sebuah berkas, dan folder di dalam aplikasi. Penggunaan Icon adalah dengan menggunakan tag `<ion-icon>` (Kode 2.33).

```

23 1 <ion-icon name="home"></ion-icon>
24

```

Kode 2.33: Potongan Kode Program dari Icon Home

- Item

Item merupakan elemen yang dapat berisi teks, ikon, avatar, gambar, masukan, dan elemen asli atau kustom lainnya. Biasanya, item ditempatkan di dalam sebuah *list* bersamaan dengan item lainnya dengan tag `<ion-item>` (Kode 2.34). Dapat dilakukan *swipe*, dihapus, disusun ulang, dan diedit.

```

35 1 <ion-item>
36 2   <ion-label>
37 3     Item
38 4   </ion-label>
39 5 </ion-item>
40

```

Kode 2.34: Potongan Kode Program dari Item Component

- List

Komponen List terdiri dari beberapa baris *item* yang dapat berisi teks, tombol, *toggles*, ikon, thumbnails, dan komponen-komponen lainnya menggunakan tag `<ion-list>` (Kode 2.35). List mendukung untuk pengguna melakukan *swiping*, *dragging*, dan penghapusan *item*.

```

1 1 <ion-list>
2 2   <ion-item>
3 3     <ion-label>Pokemon Yellow</ion-label>
4 4   </ion-item>
5 5   <ion-item>
6 6     <ion-label>Mega Man X</ion-label>
7 7   </ion-item>
8 8   <ion-item>
9 9     <ion-label>The Legend of Zelda</ion-label>
10 10 </ion-item>
11 11 <ion-item>
12 12   <ion-label>Pac-Man</ion-label>
13 13 </ion-item>
14 14 <ion-item>
15 15   <ion-label>Super Mario World</ion-label>
16 16 </ion-item>
17 17 </ion-list>
18

```

Kode 2.35: Potongan Kode Program dari List Component

- Menu

Komponen Menu merupakan panel navigasi samping yang dapat dilakukan *slides* dari sisi pada tampilan halaman saat ini menggunakan tag `<ion-menu>` (Kode 2.36). Pada dasarnya, Menu muncul dari kiri, tetapi sisi kemunculan menu dapat diganti.

```

25 1 <ion-menu side="start" menuId="first" contentId="main">
26 2   <ion-header>
27 3     <ion-toolbar color="primary">
28 4       <ion-title>Start Menu</ion-title>
29 5     </ion-toolbar>
30 6   </ion-header>
31 7   <ion-content>
32 8     <ion-list>
33 9       <ion-item>Menu Item</ion-item>
34 10      <ion-item>Menu Item</ion-item>
35 11      <ion-item>Menu Item</ion-item>
36 12      <ion-item>Menu Item</ion-item>
37 13      <ion-item>Menu Item</ion-item>
38 14     </ion-list>
39 15   </ion-content>
40 16 </ion-menu>

```

Kode 2.36: Potongan Kode Program dari Menu Component

- Modal

Modal merupakan kotak dialog yang muncul diatas konten aplikasi lain, dan harus diutup secara manual oleh pengguna sebelum pengguna dapat melanjutkan menggunakan aplikasi. Modal berguna sebagai komponen pilihan ketika ada banyak opsi untuk dipilih, atau melakukan penyaringan isi di dalam daftar, serta beberapa kasus serupa lainnya. Modal dapat digunakan dengan tag `<ion-modal>` (Kode 2.37).

```

1 1 <ion-modal [isOpen]="true">
2   <ng-template>
3     <ion-content>Modal Content</ion-content>
4   </ng-template>
5 </ion-modal>

```

Kode 2.37: Kode Program dari Modal

- Navigation

Navigation adalah komponen mandiri yang digunakan untuk membuat komponen baru ke dalam *stack*. Navigation tidak terikat kepada *router* tertentu, mengakibatkan jika kita membuat komponen Navigation dan melakukan *push* komponen lain ke dalam *stack*, komponen tersebut tidak akan mempengaruhi router aplikasi secara keseluruhan. Sesuai dengan kasus penggunaan dimana ketika pengguna bisa memilih modal, yang membutuhkan sub-navigasinya sendiri, tanpa membuatnya terikat ke URL aplikasi.

- Refresher

Refresher merupakan komponen yang menyediakan fungsionalitas *pull-to-refresh* pada komponen konten dengan tag `<ion-refresher>` (Kode 2.38). Refresher digunakan pada saat pengguna menarik layar dari atas ke bawah, maka Refresher akan menyegarkan data atau mendapatkan daftar data untuk mengambil lebih banyak data.

```

1 <ion-content>
2   <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
3     <ion-refresher-content></ion-refresher-content>
4   </ion-refresher>
5 </ion-content>

```

Kode 2.38: Kode Program dari Refresher

- Segment

Segment berfungsi untuk menampilkan pilihan tombol bagi pengguna untuk beralih di antara tampilan berbeda di dalam satu halaman yang sama. Segment menampilkan sekelompok tombol-tombol yang dapat diklik, dalam baris horizontal. Penggunaan Segment dengan menggunakan tag `<ion-segment>` (Kode 2.39).

```

1 <ion-segment (ionChange)="segmentChanged($event)">
2   <ion-segment-button value="friends">
3     <ion-label>Friends</ion-label>
4   </ion-segment-button>
5   <ion-segment-button value="enemies">
6     <ion-label>Enemies</ion-label>
7   </ion-segment-button>
8 </ion-segment>

```

Kode 2.39: Kode Program dari Segment

- Slides

Komponen Slides merupakan sebuah *multi-section container* yang setiap bagianya dapat dilakukan *swipe* atau *drag*. Untuk menggunakan komponen ini dengan menggunakan tag `<ion-slides>` (Kode 2.40).

```

1 1 <ion-slides pager="true" [options]="slideOpts">
2   2   <ion-slide>
3     3     <h1>Slide 1</h1>
4   4   </ion-slide>
5   5   <ion-slide>
6     6     <h1>Slide 2</h1>
7   7   </ion-slide>
8   8   <ion-slide>
9     9     <h1>Slide 3</h1>
10 10 </ion-slide>
11 11 </ion-slides>
12

```

Kode 2.40: Kode Program dari Slides

- Tabs

Tabs merupakan navigasi *top-level* yang mengimplementasi sebuah *tab-based navigation*. Tabs dapat digunakan dengan tag `<ion-tabs>` (Kode 2.41) yang tidak memiliki *styling* apapun dan bekerja sebagai *router outlet* untuk menangani navigasi.

```

18
19 1 <ion-tabs>
20 2   <ion-tab-bar slot="bottom">
21 3     <ion-tab-button tab="schedule">
22 4       <ion-icon name="calendar"></ion-icon>
23 5       <ion-label>Schedule</ion-label>
24 6       <ion-badge>6</ion-badge>
25 7     </ion-tab-button>
26 8
27 9     <ion-tab-button tab="speakers">
28 10       <ion-icon name="person-circle"></ion-icon>
29 11       <ion-label>Speakers</ion-label>
30 12     </ion-tab-button>
31 13   </ion-tab-bar>
32 14 </ion-tabs>

```

Kode 2.41: Kode Program dari Tabs

- Toast

Komponen Toast merupakan sebuah notifikasi yang digunakan di aplikasi modern untuk memberikan umpan balik atau menampilkan pesan sistem. Komponen ini muncul diatas konten aplikasi, dan dapat ditutup untuk melanjutkan penggunaan aplikasi. Komponen ini menggunakan `ToastController` yang dimiliki oleh *library* Ionic Angular (Kode 2.42)

```

39
40 1 async presentToast() {
41 2   const toast = await this.toastController.create({
42 3     message: 'Your settings have been saved.',
43 4     duration: 2000
44 5   });
45 6   toast.present();
46 7 }

```

Kode 2.42: Kode Program dari Toast

1     • Toolbar

2     Toolbar dapat diposisikan di atas ataupun di bawah konten. Ketika toolbar ditempatkan  
 3     di header <ion-header> akan muncul di bagian atas konten, sedangkan ketika ditempatkan  
 4     di footer <ion-footer> akan muncul tetap di bagian bawah. Toolbar menggunakan tag  
 5     <ion-toolbar>, yang di dalamnya dapat berisi button, dan dapat menggunakan border  
 6     (Kode 2.43).

```
7   1 <ion-toolbar>
8     2   <ion-buttons slot="start">
9       3     <ion-back-button></ion-back-button>
10      4   </ion-buttons>
11      5   <ion-title>Back Button</ion-title>
12      6 </ion-toolbar>
```

Kode 2.43: Kode Program dari Toolbar dengan Button di Dalamnya

15 Selain komponen-komponen yang telah disebutkan, tertapat beberapa komponen lainnya yang tidak  
 16 disebutkan disini. Komponen-komponen tersebut yaitu Checkbox, Chip, Floating Action Button,  
 17 Grid, Input, Popover, Progress Indicator, Radio, Reorder, Routing, Searchbar, Select, dan Toggle<sup>1</sup>.

18 **2.3.3 Migrasi Ionic 3 ke Ionic 6**

19 Untuk melakukan migrasi dari Ionic 3 ke Ionic 6 memerlukan tiga tahap, yaitu migrasi dari Ionic 3  
 20 ke Ionic 4, migrasi Ionic 4 ke Ionic 5, dan migrasi dari Ionic 5 ke Ionic 6. Tahapan migrasi tersebut  
 21 adalah sebagai berikut:

22 1. Migrasi Ionic 3 ke Ionic 4

23 Ada beberapa langkah untuk melakukan migrasi dari Ionic 3 ke dalam Ionic 4, yaitu:

24 (a) Membuat Proyek Ionic Baru

25 Untuk membuat projek Ionic baru tanpa *template* apapun dengan menggunakan perintah  
 26 **ionic start myApp blank** dan memilih Angular sebagai *frameworknya* 2.44.

```
27 1 ionic start myApp blank
```

Kode 2.44: Perintah Membuat Proyek Ionic Baru

30 (b) Menyalin Angular Services yang pada Ionic 3 berada di **src/providers**, menjadi **src/app-**  
 31 **p/services** pada Ionic 4.

32 (c) Menyalin *Root-level Items*

33 Menyalin seluruh *Root-level Items* pada Ionic versi 3, seperti *pipes* dan *components*.  
 34 Terdapat perubahan struktur direktori dengan perubahan direktori yang semula **sr-**  
 35 **c/components** pada Ionic 3, menjadi **src/app/components** pada Ionic 4.

36 (d) Menyalin Global Scss dari **src/app/app.scss** pada Ionic 3, menjadi **src/global.scss**  
 37 pada Ionic 4.

38 (e) Menyalin Bagian-bagian Aplikasi

39 Menyalin keseluruhan bagian yang ada pada aplikasi, baik itu halaman maupun fitur  
 40 yang ada, dengan ketentuan sebagai berikut :

<sup>1</sup> ‘UI Components’ <https://ionicframework.com/docs/components>, Diakses pada 17 April 2022.

- 1 • Shadow DOM sudah aktif secara *default*.
- 2 • Page atau Components Sass tidak lagi dibungkus dengan *tag page* atau *components*
- 3 dan harus menggunakan opsi styleUrls milik Angular dari dekorator @Component.
- 4 • Perubahan RxJS yang digunakan. Pada ionic 3 adalah versi 5, sedangkan pada Ionic
- 5 4 RxJS yang digunakan adalah versi 6.
- 6 • Lifecycle Hooks tertentu harus digantikan dengan Angular Hooks.
- 7 • Perubahan markup yang mungkin saja dibutuhkan.

8 Sejak Ionic 4 dipindahkan ke elemen kustom, terdapat perubahan yang signifikan  
 9 terkait dengan markup untuk setiap komponen. Semua perubahan ini dibuat untuk  
 10 mengikuti spesifikasi dari elemen kustom. Komponen-komponen yang berubah  
 tersebut yaitu :

12 – *Button*

13 Terdapat perbedaan pada *tag* untuk membuat Button, yang semula pada Ionic 3  
 14 adalah <button> menjadi <ion-button> pada Ionic 4 (Kode 2.45).

```
1 <ion-button (click)="doSomething()">
2   Default Button
3 </ion-button>
```

Kode 2.45: Penggunaan Button pada Ionic 4

20 – Floating Action Button (FAB)

21 Terdapat perbedaan pada *tag* di dalam <ion-fab>, yang semula pada Ionic 3  
 22 adalah <button> menjadi <ion-fab-button> pada Ionic 4 (Kode 2.46).

```
1 <ion-fab>
2   <ion-fab-button>
3     <ion-icon name="add"></ion-icon>
4   </ion-fab-button>
5   <ion-fab-list>
6     <ion-fab-button>
7       <ion-icon name="logo-facebook"></ion-icon>
8     </ion-fab-button>
9   </ion-fab-list>
10 </ion-fab>
```

Kode 2.46: Penggunaan Floating Action Button pada Ionic 4

35 – Item

36 Terdapat perbedaan pada *tag* <ion-item>, dimana pada Ionic 3, *tag* <ion-label>  
 37 akan secara otomatis ditambahkan ke dalam <ion-item>. Sedangkan pada Ionic  
 38 4 diharuskan untuk menambahkan *tag* <ion-label> secara manual ke dalam  
 39 komponen item (Kode 2.47).

```
1 <ion-item>
2   <ion-label>
3     Default Item
4   </ion-label>
5 </ion-item>
```

Kode 2.47: Penggunaan Item pada Ionic 4

1           – Label

2           Pada Ionic 4, atribut untuk mengatur posisi dari label digabungkan dengan  
3           atribut *position* (Kode 2.48).

```
4
5   1 <ion-item>
6   2   <ion-label position="floating">Floating Label</ion-label>
7   3   <!-- input -->
8   4 </ion-item>
```

Kode 2.48: Penggunaan Atribut *Position* pada Ionic 4

10          – Menu

11          Terdapat beberapa perubahan nama pada Ionic 4, yaitu:

- 12          \* Perubahan Nama Properti Terdapat perubahan nama properti pada Ionic 4.  
13           Perubahan-perubahan tersebut adalah sebagai berikut:

- 14           · swipeEnable

15          Terdapat perubahan swipeEnable pada Ionic 4. Perubahan tersebut adalah  
16          sebagai berikut:

17          Pada Ionic 3: swipeEnabled

18          Sedangkan pada Ionic 4 menjadi: swipeGesture

- 19           · content

20          Terdapat perubahan content pada Ionic 4. Perubahan tersebut adalah  
21          sebagai berikut:

22          Pada Ionic 3: content

23          Sedangkan pada Ionic 4 menjadi: contentId

- 24          \* Perubahan Nama Events Terdapat perubahan nama *events* pada Ionic 4.  
25           Perubahan-perubahan tersebut adalah sebagai berikut :

- 26           · ionClose

27          Terdapat perubahan ionClose pada Ionic 4. Perubahan tersebut adalah  
28          sebagai berikut:

29          Pada Ionic 3: ionClose

30          Sedangkan pada Ionic 4 menjadi: ionDidClose

- 31           · ionOpen

32          Terdapat perubahan ionOpen pada Ionic 4. Perubahan tersebut adalah  
33          sebagai berikut:

34          Pada Ionic 3: ionOpen

35          Sedangkan pada Ionic 4 menjadi: ionDidOpen

36          – Nav

37          Terdapat perubahan Nav pada Ionic 4. Perubahan-perubahan tersebut adalah  
38          sebagai berikut:

- 39          \* Perubahan Nama Method Terdapat perubahan nama *method* pada Ionic 4.  
40           Perubahan-perubahan tersebut adalah sebagai berikut:

1 · remove

2 Terdapat perubahan remove pada Ionic 4. Perubahan tersebut adalah  
3 sebagai berikut:

4 Pada Ionic 3: remove

5 Sedangkan pada Ionic 4 untuk menghindari konflik dengan HTML, berubah  
6 menjadi: removeIndex

7 · getActiveChildNavs

8 Terdapat perubahan getActiveChildNavs pada Ionic 4. Perubahan tersebut  
9 adalah sebagai berikut:

10 Pada Ionic 3: getActiveChildNavs

11 Sedangkan pada Ionic 4 menjadi: getChildNavs

12 \* Perubahan Nama Prop

13 Terdapat perubahan nama prop pada Ionic 4. Perubahan tersebut adalah  
14 sebagai berikut:

15 Pada Ionic 3: swipeBackEnabled

16 Sedangkan pada Ionic 4 menjadi: swipeGesture

17 – Navbar

18 Pada Ionic 4, terdapat penghapusan terhadap komponen `<ion-navbar>` karena  
19 untuk menjaga agar selalu menggunakan `<ion-toolbar>` dengan *back button*  
20 yang eksplisit (Kode 2.49).

```
21 1 <ion-toolbar>
22 2   <ion-buttons slot="start">
23 3     <ion-back-button></ion-back-button>
24 4   </ion-buttons>
25 5   <ion-title>My Navigation Bar</ion-title>
26 6 </ion-toolbar>
```

Kode 2.49: Penggunaan Navbar pada Ionic 4 dengan *Back Button*

29 – Overlays

30 Pada Ionic 4, semua overlay harus menggunakan `async/await`. Overlay tersebut  
31 adalah Action Sheet, Alert, Loading, Modal, Popover, and Toast (Kode 2.50).

32 Selain itu, terjadi perubahan nama propert `enableBackdropDismiss` menjadi  
33 `backdropDismiss`.

```
34 1 async presentToast() {
35 2   const toast = await this.toastController.create({
36 3     message: 'Your settings have been saved.',
37 4     duration: 2000
38 5   });
39 6   toast.present();
40 7 }
```

Kode 2.50: Penggunaan Overlay untuk Toast pada Ionic 4

43 – Scroll

44 *Tag* `<ion-scroll>` sudah dihapus pada Ionic 4 agar penggunaan *tag* `<ion-content>`  
45 menjadi lebih maksimal.

1           – Segment Button

2           Pada Ionic 4, teks pada Segment Button membutuhkan `<ion-label>` untuk  
3           membungkusnya (Kode 2.51).

```
4
5   1 <ion-segment-button>
6   2   <ion-label>Item One</ion-label>
7   3 </ion-segment-button>
```

Kode 2.51: Penggunaan Segment Button untuk Toast pada Ionic 4

9           Selain yang telah disebutkan, terdapat beberapa perubahan lainnya yang tidak  
10          ditulis seperti Action Sheet, Alert, Colors, Content, Datetime, Dynamic Mode, Fixed  
11          Content, Grid, Icon, Infinite Scroll, Item Divider, Item Options, Item Sliding, List  
12          Header, Loading, Modal, Option, Popover, Radio, Range, Refresher, Select, Show  
13          When, Hide When, Spinner, Tabs, Typography, Theming, dan Toolbar <sup>2</sup>.

14        2. Migrasi Ionic 4 ke Ionic 5

15        Migrasi aplikasi dari Ionic 4 ke Ionic 5 memerlukan beberapa pembaruan mengenai properti  
16          API, CSS, dan *package dependencies* yang terpasang. Perubahan-perubahan tersebut yaitu :

- 17        • CSS

- 18        – *Activated, Focused, Hover States*

19        Kelas `.activated` secara otomatis ditambahkan ke komponen yang dapat diklik,  
20          mengalami perubahan nama menjadi `.ion-activated`. Selain itu juga memperbarui  
21          komponen Action Sheet sehingga variabel akan diawali dengan `button`. Hal ini  
22          dapat memungkinkan aplikasi tetap memiliki kontrol atas *opacity* jika diinginkan,  
23          tetapi saat memperbarui status, hanya perlu mengatur variabel utama, yaitu -  
24          background-activated, -background-focused, -background-hover. Hal tersebut penting  
25          saat mengubah tema global, karena memperbarui warna `toolbar` akan secara otomatis  
26          memperbarui *hover states* untuk semua `buttons` di `toolbar` (Kode 2.52).

```
27
28   1 /* Setting the button background on hover to solid red */
29   2 ion-button {
30   3   --background-hover: red;
31   4   --background-hover-opacity: 1;
32   5 }
33   6
34   7 /* Setting the action sheet button background on focus to an opaque
35   8   green */
36   9 ion-action-sheet {
37  10   --button-background-focus: green;
38  11   --button-background-focus-opacity: 0.5;
39  12 }
40  13 /*
41  14 * Setting the fab button background on hover to match the text color
42  15 * with
43  16 * the default --background-hover-opacity on md
44  17 */
```

<sup>2</sup> ‘Breaking Changes’ <https://github.com/ionic-team/ionic-framework/blob/main/angular/BREAKING.md>, Diakses pada 13 November 2021.

```

1   17 .md ion-fab-button {
2     18   --color: #222;
3     19   --background-hover: #222;
4     20 }

```

Kode 2.52: Contoh Kode *Hover States* pada Ionic 5

- CSS *Utilities*

Karena pada versi sebelumnya, yaitu Ionic versi 4, terdapat masalah dengan menggunakan atribut CSS dengan *framework* yang menggunakan JSX dan TypeScript, Ionic *Framework* menambahkan dukungan untuk beberapa *framework*, dan pada Ionic 5 menambahkan kelas CSS. Ionic versi 5 menghapus atribut CSS dan mendukung konsistensi. Selain itu, Ionic versi 5 juga mengubah ke kelas dengan diawali ion untuk menghindari konflik dengan atribut asli dan CSS dari pengguna (Kode 2.53).

```

1 <ion-header class="ion-text-center"></ion-header>
2 <ion-content class="ion-padding"></ion-content>
3 <ion-label class="ion-text-wrap"></ion-label>
4 <ion-item class="ion-wrap"></ion-item>

```

Kode 2.53: Contoh Kode Kelas CSS *Utility* pada Ionic 5

- *Display Classes*

Kelas dari *responsive display* yang ditemukan di dalam berkas display.css memiliki kueri media yang diperbarui untuk lebih mencerminkan bagaimana cara kerjanya.

- *Distributed Scss*

Berkas scss telah dihapus dari dist/. Sebagai gantinya, variabel CSS harus digunakan untuk tema.

- Komponen

Terdapat perubahan beberapa komponen pada Ionic 5, yaitu :

- Back Button dan Button

Perubahan terdapat pada penambahan penamaan kelas .activated yang secara otomatis ditambahkan ke komponen yang dapat diklik, menjadi .ion-activated.

- Controllers

Terdapat beberapa komponen yang dihapus dari Ionic sebagai elemen, yaitu ion-action-sheet-controller, ion-alert-controller, ion-loading-controller, ion-menu-controller, ion-modal-controller, ion-picker-controller, ion-popover-controller, dan ion-toast-controller. Sebagai gantinya, maka harus diimpor dari @ionic/core.

- Header dan Footer

Atribut no-border dihapus, dan sebagai gantinya yaitu dengan menggunakan kelas ion-no-border.

- List Header

Konten berupa teks apa pun di dalam <ion-list-header> harus dibungkus dengan <ion-label> sesuai dengan gaya desain yang baru (Kode 2.54). Jika label tidak ada, maka perataan tombol di header bisa saja terlihat tidak aktif.

```

1 1 <ion-list-header>
2   2   <ion-label>New This Week</ion-label>
3   3   <ion-button>See All</ion-button>
4 4 </ion-list-header>

```

Kode 2.54: Kode Program untuk List Header

## – Menu

Fungsi swipeEnable() telah dihapus di Angular, sebagai gantinya menggunakan swipeGesture(). Nilai *left* dan *right* telah dihapus, dan menggunakan *start* dan *end* sebagai gantinya. Selain itu ada penghapusan atribut utama, sebagai gantinya yaitu dengan menggunakan content-id (untuk vanila JS atau Vue) dan contentId (untuk Angular atau React) (Kode 2.55).

```

1 1 <ion-menu content-id="main"></ion-menu>
2 2 <ion-content id="main">...</ion-content>

```

Kode 2.55: Kode Program untuk Menu

## – Select Option

Properti selected telah dihapus. Sebagai gantinya harus mengatur properti nilai pada ion-select induk agar sesuai dengan opsi terpilih yang diinginkan (Kode 2.56).

```

1 1 <ion-select value="two">
2 2   <ion-select-option value="one">One</ion-select-option>
3 3   <ion-select-option value="two">Two</ion-select-option>
4 4 </ion-select>

```

Kode 2.56: Kode Program untuk Select Option

## – Toast

Properti close button seperti showCloseButton dan closeButtonText telah dihapus. Sebagai gantinya, gunakan buttons array untuk fungsi batal (Kode 2.57).

```

1 1 async presentToast() {
2 2   const toast = await this.toastController.create({
3 3     message: 'Your settings have been saved.',
4 4     buttons: [
5 5       {
6 6         text: 'Close',
7 7         role: 'cancel',
8 8         handler: () => {
9 9           console.log('Close clicked');
10 10      }
11 11    }
12 12  ]
13 13  });
14 14 toast.present();
15 15 }

```

Kode 2.57: Kode Program untuk Toast

Selain yang sudah disebutkan, terdapat beberapa komponen lain yang mendapat perubahan di Ionic 5, namun tidak ditulis di dalam dokumen skripsi ini. Komponen-komponen tersebut antara lain Action Sheet, Anchor, Card, FAB, Item, Menu Button, Nav Link, Radio, Segment, Segment Button, Skeleton Text, Split Pane, dan Tabs <sup>3</sup>.

- Warna

Terdapat perubahan terhadap warna bawaan milik ionic (Tabel 2.1).

Nama Warna	Kode HEX
primary	#3880ff
secondary	#3dc2ff
tertiary	#5260ff
success	#2dd36f
warning	#ffc409
danger	#eb445a
light	#f4f5f8
medium	#92949c
dark	#222428

Tabel 2.1: Tabel Warna Bawaan di Ionic 5

- Events

Pada Ionic 5, Events services di @ionic/angular telah dihapus. Sebagai gantinya gunakan Observables untuk arsitektur pub/sub, dan Redux untuk *advanced state management*.

- Package dan Dependencies

Untuk memasang package dan dependencies pada Angular, dapat memanfaatkan npm pada CLI, dengan menjalankan pemasangan pada package ionic-angular (Kode 2.58). Namun jika ingin membuat proyek baru, dapat dibuat dari CLI dan aplikasi yang ada dapat dimigrasikan secara manual.

```
1  npm install @ionic/angular@latest @ionic/angular-toolkit@latest --save
```

Kode 2.58: Kode untuk Memasang Package dan Dependencies pada Angular

### 3. Migrasi Ionic 5 ke Ionic 6

Berikut merupakan perubahan-perubahan pada Ionic 6, diantaranya yaitu:

- Pembaruan Ionic dan Angular

Ionic 6 mendukung penggunaan Angular versi 12 dan yang lebih baru, dengan begitu perlu dilakukan perbaruan Angular ke versi yang terbaru (Kode 2.59).

```
1  npm install @ionic/angular@6
```

Kode 2.59: Kode untuk Memperbarui Versi Ionic 6 dengan versi Angular Terbaru

- Mengganti penggunaan Config.set() menjadi IonicModule.forRoot().

- Icon

Penghapusan properti ariaLabel dan ariaHidden, dan diganti dengan menggunakan aria-label dan aria-hidden.

<sup>3</sup> ‘Breaking Changes’ <https://github.com/ionic-team/ionic-framework/blob/main/BREAKING.md>, Diakses pada 20 November 2021.

1     • *Input, Select, dan Textarea*

2       Menggunakan “undefined” sebagai nilai untuk diteruskan ke properti placeholder, dibandingkan dengan menggunakan “null”.

3

4     • *Modal dan Popover*

5       ion-modal (Kode 2.60) dan ion-popover (Kode 2.61) menggunakan Shadow DOM.

```
6
7 1 ion-modal::part(content) {
8 2   ...
9 3 }
10
11 5 ion-modal::part(backdrop) {
12 6   ...
13 7 }
```

Kode 2.60: Kode ion-modal menggunakan Shadow DOM pada CSS

```
15
16 1 ion-popover::part(arrow) {
17 2   ...
18 3 }
19
20 5 ion-popover::part(backdrop) {
21 6   ...
22 7 }
23
24 9 ion-popover::part(content) {
25 10   ...
26 11 }
```

Kode 2.61: Kode ion-popover menggunakan Shadow DOM pada CSS

28     • Radio

29       Menghapus semua penggunaan antarmuka RadioChangeEventDetail.

1

## BAB 3

2

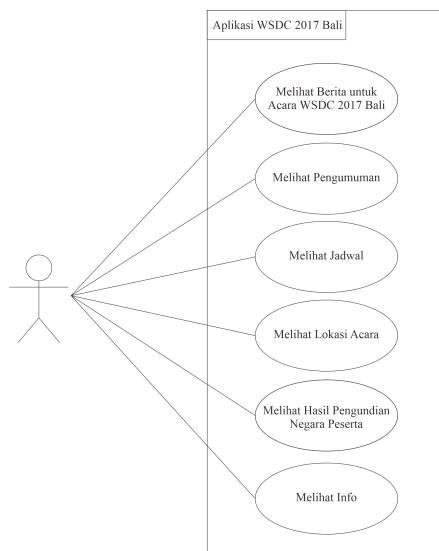
## ANALISIS

- 3 Pada bab ini akan dijelaskan analisis aplikasi WSDC 2017 Bali saat ini dan aplikasi WSDC yang  
4 akan dibangun. Analisis yang akan dibahas meliputi analisis *use case*, analisis kebutuhan sistem,  
5 dan analisis pembangunan aplikasi Android menggunakan Ionic.

### 6 3.1 Analisis Sistem Kini

- 7 Aplikasi WSDC 2017 Bali digunakan untuk menunjang keberlangsungan acara WSDC 2017 yang  
8 diselenggarakan di Bali, Indonesia. Pada halaman utama, pengguna dapat melihat berita-berita  
9 terkait acara WSDC 2017 Bali dan tombol *read more* yang apabila ditekan akan mengarahkan  
10 pengguna untuk melihat berita terkait acara WSDC 2017 Bali dengan format pdf. Aplikasi WSDC  
11 2017 Bali dapat digunakan untuk melihat berita acara, pengumuman, jadwal peserta, lokasi acara,  
12 hasil pengundian, info, serta pengumuman pemenang dari acara WSDC 2017 Bali (Gambar 3.1).

- 13 Aplikasi WSDC 2017 Bali dibangun menggunakan *framework* Ionic versi 3, dan Angular versi  
14 4.1.3. Dengan digunakannya Ionic Framework, maka memungkinkan aplikasi WSDC 2017 Ba-  
15 li menggunakan teknologi web seperti HTML, dan CSS. Aplikasi WSDC 2017 Bali dibangun  
16 menggunakan Cordova agar dapat berjalan secara *native*. Penggunaan Cordova memungkinkan  
17 aplikasi WSDC 2017 Bali kompatibel dengan perangkat berbasis Android dan IOS, tanpa perlu  
18 mengimplementasikannya kembali ke dalam bahasa masing-masing platform.



Gambar 3.1: *Use Case Diagram* Aplikasi WSDC 2017 Bali

### 1 3.1.1 Skenario Pengguna

2 Terdapat *sidemenu* untuk pengguna agar dapat bermavigasi ke dalam menu-menu yang terdapat  
 3 pada aplikasi WSDC 2017 Bali. Untuk mengakses *sidemenu*, pengguna dapat menekan tombol  
 4 navigasi berada di sebelah kiri atas aplikasi WSDC 2017 Bali. Selain itu dapat pula dengan cara  
 5 mengusap layar dari kiri ke kanan. Untuk menutup *sidemenu*, pengguna dapat menekan area di  
 6 luar *sidemenu*, atau dengan cara menekan tombol silang di sebelah kiri atas *sidemenu*. Terdapat  
 7 fitur-fitur yang ada pada aplikasi WSDC 2017 Bali yang dapat diakses melalui *sidemenu*. Fitur-fitur  
 8 tersebut adalah sebagai berikut :

9 1. *Home*

10 Pada halaman ini, pengguna dapat melihat halaman utama aplikasi WSDC 2017 Bali yang  
 11 berisi berita acara WSDC 2017 Bali, serta pemberitahuan terakhir terkait acara WSDC 2017  
 12 Bali. Halaman ini merupakan halaman awal yang ditampilkan saat aplikasi WSDC 2017 Bali  
 13 pertama kali dibuka (Tabel 3.1). Untuk mengakses halaman ini, dapat melalui *sidemenu*.

No	Aksi Aktor	Reaksi Sistem
1	Pengguna membuka aplikasi WSDC 2017 Bali	Aplikasi WSDC 2017 Bali menampilkan halaman selamat datang.
2		Aplikasi WSDC 2017 Bali menampilkan halaman <i>Home</i>
3	Pengguna mengklik <i>card Announcements</i>	Aplikasi WSDC 2017 Bali menampilkan halaman <i>Announcement</i> .

Tabel 3.1: Tabel Skenario dari Halaman *Home*

14 2. *Newsletter*

15 Pada *bagian newsletter* yang terdapat di *home*, pengguna dapat melihat berita-berita terkait  
 16 acara WSDC 2017 Bali dengan format pdf. Untuk mengakses halaman ini, dapat melalui  
 17 *sidemenu* (Tabel 3.2).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>read more</i> pada berita di halaman utama aplikasi WSDC 2017 Bali.	Aplikasi WSDC 2017 Bali menampilkan berita pada acara WSDC 2017 Bali

Tabel 3.2: Tabel Skenario dari *Newsletter*

18 3. *Announcement*

19 Pengguna dapat melihat berbagai pengumuman mengenai keberlangsungan acara WSDC 2017  
 20 Bali yang tersusun berdasarkan tanggal dirilisnya pengumuman tersebut. Untuk mengakses  
 21 halaman ini, dapat melalui *sidemenu* (Tabel 3.3).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Announcement</i> pada <i>sidemenu</i>	Aplikasi WSDC 2017 Bali menampilkan halaman <i>Announcement</i> .

Tabel 3.3: Tabel Skenario dari Halaman *Announcement*

1      4. *Schedule*

2      Pada halaman ini, pengguna dapat melihat jadwal acara WSDC 2017 Bali yang ditampilkan  
 3      berkelompok berdasarkan tanggal dan hari. Jadwal yang ditampilkan berupa waktu mulai dan  
 4      waktu selesai, lokasi acara, serta nama acara. Untuk mengakses halaman ini, dapat melalui  
 5      sidemenu (Tabel 3.4).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol Schedule pada sidemenu	Aplikasi WSDC 2017 Bali menampilkan halaman <i>Schedule</i> .
2	Pengguna menekan tanggal yang berada di atas halaman jadwal	Aplikasi WSDC 2017 Bali menampilkan jadwal berdasarkan tanggal yang dipilih oleh pengguna dengan detail waktu, lokasi, dan nama kegiatan.

Tabel 3.4: Tabel Skenario dari Halaman *Schedule*

6      5. *Venues*

7      Pada halaman ini, pengguna dapat melihat lokasi dari berlangsungnya acara WSDC 2017  
 8      Bali. Untuk mengakses halaman ini, dapat melalui sidemenu (Tabel 3.5).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol Venues pada sidemenu	Aplikasi WSDC 2017 Bali menampilkan halaman <i>Venues</i> yang berisi <i>Ceremony Venues</i> , <i>Competition Venues</i> , <i>Delegates Accommodation</i> , dan <i>Educational Tour</i> .
2	Pengguna menekan kategori <i>venues</i> yang diinginkan.	Aplikasi WSDC 2017 Bali menampilkan peta, nama lokasi acara dengan disertai penanda yang ada di dalam peta, dan jarak antara lokasi pengguna saat ini dan lokasi acara.

Tabel 3.5: Tabel Skenario dari Halaman *Venues*

9      6. *Draw*

10     Pada halaman ini, pengguna dapat melihat pembagian *venue* serta pembagian kubu proposisi  
 11    dan oposisi dari hasil pengundian untuk para negara peserta WSDC 2017 Bali. Untuk  
 12    mengakses halaman ini, dapat melalui sidemenu (Tabel 3.6).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol Draw pada sidemenu	Aplikasi WSDC 2017 Bali menampilkan halaman <i>Draw</i> yang dapat digulir kebawah untuk menampilkan keseluruhan tabel.

Tabel 3.6: Tabel Skenario dari Halaman *Draw*

1      7. *Result*

2      Pada halaman ini, pengguna dapat melihat pemenang dari kompetisi WSDC 2017 Bali. Untuk  
3      mengakses halaman ini, dapat melalui *sidemenu* (Tabel 3.7).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Result</i> pada <i>sidemenu</i>	Aplikasi WSDC 2017 Bali menampilkan halaman <i>Result</i> yang berisi pemenang dari babak semifinal, seperempatfinal, dan se-perdelapanfinal.

Tabel 3.7: Tabel Skenario dari Halaman *Result*

4      8. Info

5      Pada halaman ini, pengguna dapat melihat info-info seputar kontak-kontak penting yang  
6      dapat dihubungi, kosa kata dalam Bahasa Indonesia sehari-hari, serta *credits* kepada pembuat  
7      aplikasi WSDC 2017 Bali. Untuk mengakses halaman ini, dapat melalui *sidemenu* (Tabel 3.8).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>hamburger</i> di pojok kiri atas atau melakukan <i>swipe</i> dari kiri layar ke kanan layar aplikasi WSDC 2017 Bali.	Aplikasi WSDC 2017 Bali menampilkan <i>side bar</i>
2	Pengguna menekan tombol Info	Aplikasi WSDC 2017 Bali menampilkan halaman Info

Tabel 3.8: Tabel Skenario dari Halaman Info

8      3.1.2 Struktur Ionic 3

9      Aplikasi WSDC 2017 Bali saat ini menggunakan Ionic versi 3, Angular versi 4.0.0, dan Cordova.  
10     Dengan Ionic Framework yang disusun berdasarkan arsitektur Angular, maka aplikasi WSDC 2017  
11     memungkinkan untuk ditulis menggunakan bahasa pemrograman web seperti HTML, CSS, dan  
12     Javascript. Pada Ionic Framework versi 3 juga terdapat UI Component 2.3.2 yang digunakan dalam  
13     aplikasi WSDC 2017 Bali, diantarnya yaitu Badge, Button, Card, Content, Grid, Icons, Items, List,  
14     Menu, Segment, Slides, Tabs, dan Toolbar. Kemudian dengan digunakannya Cordova, maka seluruh  
15     kode program yang menggunakan bahasa pemrograman web tersebut, dapat hidup dan berjalan  
16     seperti halnya aplikasi *native* di dalam perangkat seluler.

17     Anatomi pada Ionic Framework memiliki struktur proyek Cordova. Pada saat pertama kali  
18     dijalankan, aplikasi WSDC 2017 Bali secara *default* akan membuka file index.html yang berada  
19     di folder src/index.html. File ini merupakan file pertama yang dijalankan untuk aplikasi WSDC  
20     2017 Bali. Tujuan dari file ini adalah untuk melakukan pengaturan terhadap script, CSS, serta  
21     menjalankan aplikasi. Di dalam file index.html ini terdapat sebuah tag <ion-app>. Tag ini yang  
22     pertama dicari dan dijalankan oleh Ionic untuk membuka komponen *root* dari aplikasi WSDC 2017  
23     Bali. Pada saat pertama menjalankan aplikasi, kode di dalam folder src akan ditranspilasikan  
24     ke versi JavaScript yang dapat dipahami browser. Dengan begitu, aplikasi dapat menjalankan  
25     TypeScript yang dikompilasi ke bentuk JavaScript.

Setelah index.html dijalankan, titik masuk ke dalam aplikasi WSDC 2017 Bali adalah file app.module.ts yang berada di src/app/app.module.ts. Di dalam file ini terdapat NgModule untuk mendeklarasi komponen apa saja yang akan digunakan, mengimpor module, bootstrap apa yang digunakan, dan menyediakan *services* apa yang akan digunakan oleh komponen (Kode 3.1).

```

1  @NgModule({
2    declarations: [
3      MyApp, HomePage, AnnouncementsPage, SchedulePage, VenuesPage, VenuesMapPage,
4      DrawPage, ResultPage, InfoPage
5    ],
6    imports: [
7      BrowserModule, HttpModule, IonicModule.forRoot(MyApp), IonicStorageModule.
8      forRoot(), CloudModule.forRoot(cloudSettings)
9    ],
10   bootstrap: [IonicApp],
11   entryComponents: [
12     MyApp, HomePage, AnnouncementsPage, SchedulePage, VenuesPage, VenuesMapPage,
13     DrawPage, ResultPage, InfoPage
14   ],
15   providers: [
16     StatusBar, SplashScreen, InAppBrowser, {provide: ErrorHandler, useClass:
17     IonicErrorHandler}, Geolocation,
18   ]
19 })
20 export class AppModule {}
```

Kode 3.1: NgModule pada app.module.ts

Komponen *root* diatur ke MyApp yang berada di folder src/app/app.component.ts. Karena pada file app.component.ts, *root* telah diatur ke dalam MyApp, maka komponen tersebut menjadi komponen pertama yang dibuka ke dalam aplikasi WSDC 2017 Bali. Di dalam komponen tersebut terdapat templateUrl yang digunakan sebagai template utama dari aplikasi WSDC 2017 Bali, yaitu file app.html (Kode 3.2). Di dalam template, terdapat tag <ion-menu> yang digunakan untuk menampilkan sidemenu, lalu tag <ion-nav> sebagai area koten utama, dengan properti [root] = “rootPage”. Properti tersebut yang nantinya akan diisi oleh halaman *root* dari aplikasi WSDC 2017 Bali, yaitu Home Page. Variabel rootPage telah diatur di file app.component.ts secara spesifik mengarah ke HomePage, yang akan menjadi halaman petama yang ditampilkan di nav controller.

```

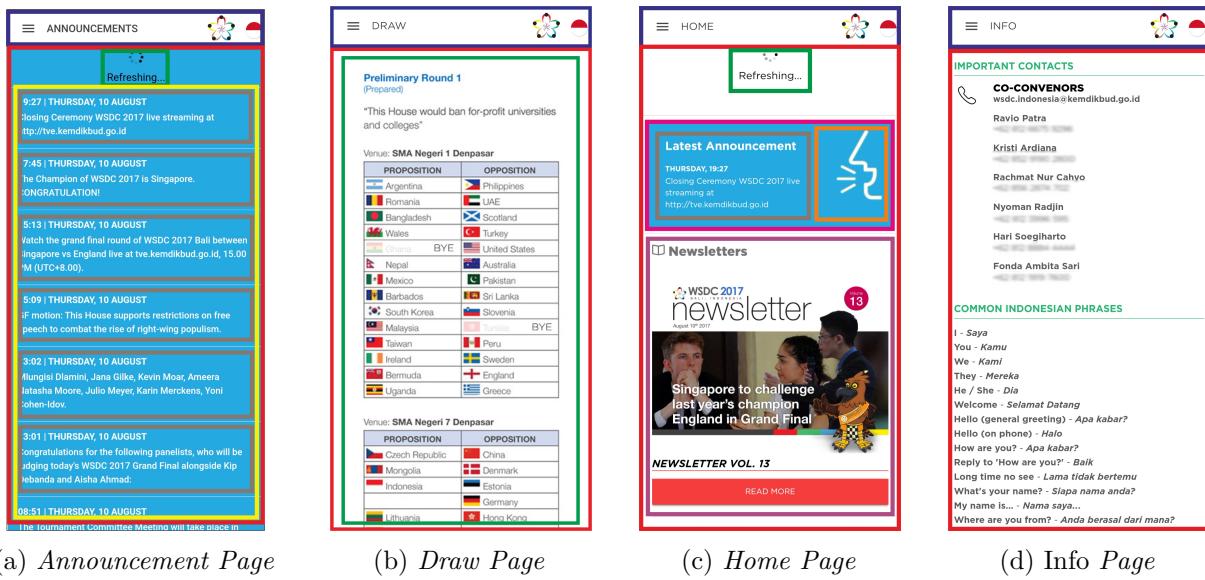
1 <ion-menu [content]="content">
2   <ion-header>
3     <ion-toolbar>
4       <ion-title>
5         <button menuClose id="menu-close-btn">
6           <ion-icon menu-close ios="ios-close-circle-outline" md="md-close-circle">
7         </ion-icon>
8       </button>
9       <span class="text">Menu</span>
10      </ion-title>
11    </ion-toolbar>
12  </ion-header>
13
14  <ion-content>
15    <ion-list>
16      <button class="title-sidemenu" menuClose ion-item *ngFor="let p of pages" (
17        click)="openPage(p)">
18        <ion-icon [ios]=p.iosicon [md]=p.mdicon></ion-icon>
19        <span class="text">{{p.title}}</span>
20      </button>
21    </ion-list>
22  </ion-content>
23
24 </ion-menu>
25
26 <!-- Disable swipe-to-go-back because it's poor UX to combine STGB with side menus
27 -->
28 <ion-nav [root]="rootPage" #content swipeBackEnabled="false"></ion-nav>
29
30

```

Kode 3.2: Source Code File app.html

31 Selain komponen *root*, terdapat pula beberapa komponen lain yang berisi halaman-halaman  
 32 yang ada di aplikasi WSDC 2017 Bali. Masing-masing komponen akan mengimpor Component  
 33 dari @angular/core, NavController dari ionic-angular, dan Storage dari @ionic/storage. Mengimpor  
 34 Component dari @angular/core berfungsi untuk menambahkan sebuah komponen ke dalam *module*.  
 35 Dengan begitu, komponen tersebut bisa terlihat di seluruh aplikasi, dan dapat digunakan oleh kom-  
 36 ponen lain. NavController merupakan *base class* untuk mengatur komponen navigasi. Ini berguna  
 37 agar aplikasi dapat berpindah antar halaman. Sedangkan Storage berfungsi untuk menyimpan  
 38 pasangan *key/value* dan sebuah objek JSON.

39 Setiap komponen memiliki tiga buah *file* utama, yaitu *file* HTML, CSS, dan TypeScript. *File*  
 40 HTML digunakan untuk menampilkan sebuah halaman ke dalam aplikasi dengan susunan kode  
 41 HTML. *File* CSS digunakan untuk mengatur desain, bentuk, dan tampilan dari sebuah halaman.  
 42 Sedangkan *file* TypeScript digunakan untuk mengontrol jalannya sebuah komponen.



Gambar 3.2: Wireframe Aplikasi WSDC 2017 Bali

Komponen-komponen yang ada pada aplikasi WSDC 2017 Bali adalah sebagai berikut:

### 1. Komponen *Announcement*

Komponen ini digunakan untuk menampilkan halaman *Announcement* pada aplikasi. Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam *file* announcement.ts terdapat sebuah *decorator* @Component untuk komponen (Kode 3.3). Di dalam decorator ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta **templateUrl** untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template* HTML yang digunakan adalah *file* announcement.html.

```

1  @Component({
2    selector: 'page-announcements',
3    templateUrl: 'announcements.html'
4  })

```

Kode 3.3: @Component pada announcement.ts

Pada komponen ini terdapat sebuah kelas AnnouncementsPage yang berisi beberapa *method* yang akan digunakan di dalam aplikasi. *Method* pada kelas ini diantaranya adalah sebagai berikut:

- ionViewDidLoad()

*Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman *announcement* telah dimuat. *Event* ini hanya berjalan satu kali, jadi jika sudah melakukan cache terhadap halaman ini, maka ionViewDidLoad() tidak akan berjalan lagi. *Lifecycle event* ini berguna untuk melakukan pengaturan awal, yaitu untuk memuat data *announcement* dari *storage*, dan menyimpannya di dalam variabel lokal.

- doRefresh(refresher)

*Method* ini berfungsi untuk melakukan penyegaran ulang pada halaman *announcement* untuk mendapatkan data *announcement* terbaru di dalam server, kemudian menyimpannya ke dalam penyimpanan Ionic. *Method* ini memiliki sebuah parameter refresher, yang berisi sebuah CustomEvent dari penyegaran ulang yang dilakukan.

- 1     • presentConnectionAlert()
- 2        Method ini digunakan ketika method doRefresh() mengalami *error*, yang kemudian
- 3        memunculkan *toast*.
- 4     • formatDate(sqlDatetime: string)
- 5        Method ini berfungsi untuk membuat format tanggal dan waktu. Method ini memiliki
- 6        sebuah parameter, yaitu sqlDatetime yang bertipe *string*, yang merupakan sebuah
- 7        *string* tanggal dengan format “tahun-bulan-hari jam-menit-detik”. Method ini akan
- 8        mengembalikan sebuah teks yang berisi waktu, tanggal dan bulan.

9     File announcement.html digunakan untuk menampilkan halaman *announcemnet*. Terdapat  
 10    beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan ke dalam  
 11    halaman *announcement*. Diantaranya adalah sebagai berikut:

- 12     • *Header*

13     Pada *header* dari halaman *announcement*, digunakan beberapa *tag* dari komponen yang  
 14    disediakan oleh Ionic Framework (Kode 3.4). Yaitu *tag* <ion-header> yang merupakan  
 15    komponen *parent* yang menampung komponen *toolbar* yang ditandai dengan warna biru  
 16    pada gambar 3.2a. Di dalam *tag* tersebut terdapat *tag* pendukung, seperti <ion-navbar>,  
 17    <button> sebagai tombol untuk membuka *sidemenu*, <ion-icon> untuk menampilkan  
 18    icon dari tombol pada *tag button*, dan <ion-title> untuk menampilkan judul dari  
 19    halaman, yaitu *Announcement*, pada *navbar*.

```
20
21 1 <ion-header>
22 2   <ion-navbar>
23 3     <button ion-button menuToggle>
24 4       <ion-icon name="menu"></ion-icon>
25 5     </button>
26 6     <ion-title>Announcements</ion-title>
27 7   </ion-navbar>
28 8 </ion-header>
```

Kode 3.4: *Header* pada Halaman *Announcement*

- 30     • *Content*

31     Konten pada halaman *announcement* yang ditandai dengan kotak berwarna merah pada  
 32    gambar 3.2a disusun menggunakan *tag* <ion-content> (Kode 3.5). Tag ini berisi beberapa  
 33    *tag* lain, yaitu *tag* <ion-refresher>, yang ditandai dengan kotak hijau, yang akan  
 34    menampilkan simbol *refresh* saat pengguna menyegarkan halaman dengan cara melakukan  
 35    *swipe* dari atas ke bawah layar. Kemudian terdapat *tag* <ion-list> yang ditandai  
 36    dengan kotak kuning, berfungsi untuk menampilkan baris. Baris-baris tersebut diisi  
 37    menggunakan *tag* <ion-item> yang ditandai dengan kotak berwarna hitam, digunakan  
 38    untuk menyimpan teks yang berisi tanggal, dan pesan pengumuman.

```

1   1 <ion-content>
2   2   <ion-refresher (ionRefresh)="doRefresh($event)">
3   3     <ion-refresher-content pullingIcon="arrow-dropdown" pullingText="Pull
4   4       to refresh" refreshingSpinner="circles" refreshingText="Refreshing...
5   5     ">
6   6   </ion-refresher-content>
7   7 </ion-refresher>
8   8 <ion-list>
9   9   <ion-item text-wrap *ngFor="let announcement of announcements">
10 10     <h3>{{formatDatetime(announcement.localtime)}}</h3>
11 11     <p>{{announcement.message}}</p>
12 12   </ion-item>
13 13 </ion-list>
14 14 </ion-content>
15 15
16 16

```

Kode 3.5: Content pada Halaman Announcement

## 2. Komponen Draw

Komponen *draw* digunakan untuk menampilkan halaman *Draw* pada aplikasi. Terdapat *file* TypeScript, draw.ts, yang berfungsi untuk mengatur keseluruhan halaman. Di dalam *file* tersebut terdapat *decorator* @Component (Kode 3.6) dan *decorator* @ViewChild (Kode 3.7). Pada *decorator* @Component, terdapat CSS *selector* untuk memilih CSS mana yang akan digunakan, serta **templateUrl** untuk mendefinisikan eksternal HTML *template* halaman *Draw* yang akan digunakan, yaitu draw.html. @ViewChild digunakan untuk memanggil elemen dari DOM untuk memanggil komponen API ke dalam TypeScript, yaitu pada komponen *draw* adalah drawIFrame yang berada di *file* draw.html.

```

1 @Component({
2   selector: 'page-draw',
3   templateUrl: 'draw.html'
4 })

```

Kode 3.6: @Component pada draw.ts

```

32 1 @ViewChild('drawIFrame') drawIFrame: ElementRef;
33
34

```

Kode 3.7: @ViewChild pada draw.ts

Terdapat kelas DrawPage yang berisi beberapa *method* yang akan digunakan di dalam aplikasi, diantaranya adalah sebagai berikut:

- ionViewDidLoad()

*Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman *draw* telah dimuat. *Event* ini hanya berjalan satu kali, jadi jika sudah melakukan cache terhadap halaman ini, maka ionViewDidLoad() tidak akan berjalan lagi. *Lifecycle event* ini berguna untuk melakukan pengaturan awal, yaitu untuk memuat data *draw* dari *storage*, kemudian data tersebut dimasukkan ke dalam *child* drawIFrame. Terakhir, method ini akan memanggil method presentLoading().

1     • presentLoading()

2         *Method* ini berfungsi untuk menampilkan sebuah *overlay* yang menunjukkan sebuah pesan  
 3         dan indikator pemuatan saat pertama kali halaman *draw* dimuat. Karena *overlay* ini  
 4         muncul di atas konten aplikasi, maka aktivitas pengguna akan diblokir untuk sementara  
 5         sampai seluruh halaman dimuat, yaitu sampai *method* onDrawIframeLoad() selesai.

6     • onDrawIframeLoad()

7         *Method* ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam *tag*  
 8         <iframe> pada draw.html yaitu *event* (load). *Method* ini berfungsi untuk menampilkan  
 9         data yang telah diambil yang disimpan di dalam *child* drawIFrame.

10      Selain itu terdapat *file* draw.html yang digunakan untuk menampilkan tata letak dari ha-  
 11      laman *draw*. Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang  
 12      diimplementasikan ke dalam halaman *draw*. Diantaranya adalah sebagai berikut:

13     • *Header*

14         *Header* dari halaman *draw* seperti pada gambar 3.2b menggunakan *tag* <ion-header>  
 15         (Kode 3.8). *Tag* tersebut merupakan komponen *parent* yang menampung komponen  
 16         navbar yang ditandai dengan kotak berwarna biru pada gambar 3.2b. Di dalam navbar  
 17         tersebut, terdapat sebuah *tag* <button> untuk memunculkan sidemenu, <ion-icon>  
 18         untuk menampilkan icon dari tombol pada *tag* button, dan *tag* <ion-title> sebagai  
 19         judul dari halaman.

```
20
21 1 <ion-header>
22 2   <ion-navbar>
23 3     <button ion-button menuToggle>
24 4       <ion-icon name="menu"></ion-icon>
25 5     </button>
26 6     <ion-title>Draw</ion-title>
27 7   </ion-navbar>
28 8 </ion-header>
```

Kode 3.8: *Header* pada draw.html

30     • *Content*

31         *Content* dari halaman *draw* seperti pada gambar 3.2b menggunakan *tag* <ion-content>  
 32         (Kode 3.9) yang ditandai menggunakan kotak berwarna merah. Di dalam *tag* ini terdapat  
 33         sebuah *tag* <iframe> yang berisi hasil pengundian grup untuk peserta WSDC 2017 Bali,  
 34         ditandai menggunakan kotak berwarna hijau. *Tag* <iframe> menampilkan hasil dari  
 35         *method* onDrawIframeLoad() pada draw.ts.

```
36
37 1 <ion-content>
38 2   <iframe #drawIFrame (load)="onDrawIframeLoad()" class="iframe-
39 3     fullscreen"></iframe>
40 4 </ion-content>
```

Kode 3.9: *Content* pada draw.html

1     3. Komponen *Home*

2     Komponen ini digunakan untuk menampilkan halaman *Home* pada aplikasi. Komponen ini  
 3     memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam *file* home.ts  
 4     terdapat sebuah *decorator* @Component untuk komponen (Kode 3.10). Di dalam decorator  
 5     ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta templateUrl untuk  
 6     mendefinisikan eksternal HTML *template* yang akan digunakan. *Template* HTML yang  
 7     digunakan adalah *file* home.html.

```
8
9   1 @Component({
10   2   selector: 'page-home',
11   3   templateUrl: 'home.html'
12   4 })
```

Kode 3.10: @Component pada home.ts

14    Komponen *Home* merupakan komponen yang menjadi rootPage dari aplikasi ini, yang dimasukkan di dalam *file* app.component.ts. Maka dari itu, saat pertama kali aplikasi dijalankan,  
 15    komponen *home*-lah yang pertama kali ditampilkan di dalam layar. rootPage di dalam *file*  
 16    app.component.ts akan memanggil komponen *home*, yang kemudian *file* home.ts akan berjalan.  
 17    Di dalam *file* ini terdapat sebuah kelas HomePage yang berisi beberapa *method*, diantaranya  
 18    adalah sebagai berikut:

- 20     • ionViewDidLoad()

21     *Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman *home* telah  
 22     dimuat. *Event* ini hanya berjalan satu kali, jadi jika sudah melakukan cache terhadap  
 23     halaman ini, maka ionViewDidLoad() tidak akan berjalan lagi. *Lifecycle event* ini  
 24     berguna untuk melakukan pengaturan awal, yaitu untuk mengambil keseluruhan  
 25     data aplikasi yang ada di dalam penyimpanan. Dengan memanfaatkan fitur *storage* yang  
 26     dimiliki oleh Ionic Framework, *method* ini akan mengecek apakah sudah ada data dengan  
 27     format .json yang berisi keseluruhan data aplikasi di dalam penyimpanan. Data tersebut  
 28     berisi data *announcements*, *newsletters*, *schedule*, *venues*, *draws*, dan *info*. Jika data  
 29     tersebut tidak ditemukan, maka akan diambil dari *file* wsdc-data.json yang kemudian  
 30     dimasukan ke dalam penyimpanan.

31     *Method* ini akan melakukan penyegaran terhadap data yang sudah ada di dalam penyimpanan dengan mengambil data dari server dengan batas waktu maksimal untuk terhubung  
 32     dengan server. Jika sudah melewati batas waktu, dan aplikasi belum terhubung dengan  
 33     server, maka *method* ini akan memanggil *method* showToast() yang akan menampilkan  
 34     Toast yang berisi teks ‘Failed to refresh information’. Jika penyegaran berhasil, maka  
 35     data yang didapatkan dari server akan dimasukan ke dalam penyimpanan internal.

- 37     • launch(url: string)

38     *Method* ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam tag  
 39     <button> pada home.html yaitu *event* (click). *Method* ini memiliki sebuah parameter url  
 40     yang bertipe *string*. Parameter tersebut berisi url dari berita yang akan dilihat oleh pengguna. Untuk membuka berita pada url tersebut memanfaatkan *plugin* InAppBrowser  
 41     yang disediakan oleh Ionic.

- 43     • formatDatetime(sqlDatetime: string)

1       Method ini berfungsi untuk membuat format tanggal dan waktu. Method ini memiliki  
 2       sebuah parameter, yaitu sqlDatetime yang bertipe *string*, yang merupakan sebuah  
 3       *string* tanggal dengan format “tahun-bulan-hari jam-menit-detik”. Method ini akan  
 4       mengembalikan sebuah teks yang berisi waktu, tanggal dan bulan.

- 5       • doRefresh(refresher)

6       Method ini berfungsi untuk melakukan penyegaran ulang pada halaman *home* untuk  
 7       mendapatkan data *home* terbaru di dalam server, kemudian menyimpannya ke dalam  
 8       penyimpanan. Method ini memiliki sebuah parameter refresher, yang berisi sebuah  
 9       CustomEvent dari penyegaran ulang yang dilakukan. Method ini akan melakukan  
 10      pemanggilan kembali kepada server, dalam batas waktu tertentu. Jika batas waktu  
 11      maksimal telah tercapai, sedangkan server belum juga memberi tanggapan, maka akan  
 12      memanggil method *showToast()* yang akan menampilkan sebuah Toast yang berisi teks  
 13      ‘Failed to refresh information’. Jika berhasil untuk terhubung dengan server, method ini  
 14      akan menghapus data yang berada di penyimpanan, dan digantikan dengan data yang  
 15      telah didapatkan dari server.

- 16       • showToast(message: string, duration: number = 3000)

17       Method ini berfungsi untuk memunculkan sebuah native Toast, yaitu sebuah *popup* teks,  
 18       dengan memanfaatkan UI Component milik Ionic Framework. Method ini menerima  
 19       parameter berupa sebuah string, yang berisi pesan yang akan dimunculkan ke dalam  
 20       sebuah Toast, dan memiliki sebuah parameter *duration* yang berisi lama waktu

- 21       • onAnnouncementClick()

22       Method ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam *tag*  
 23       <ion-card> pada home.html yaitu *event* (click). Method ini berfungsi untuk berpindah  
 24       halaman menjadi halaman *announcement*.

25       File home.html digunakan untuk menampilkan tata letak dari halaman *home*. Terdapat  
 26       beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan ke dalam  
 27       halaman *home*. Diantaranya adalah sebagai berikut:

- 28       • *Header*

29       Halaman *home* memiliki *header* dengan *tag* <ion-header> (Kode 3.11) seperti pada gam-  
 30       bar 3.2c yang ditandai dengan kotak berwarna biru. Tag tersebut merupakan komponen  
 31       parent yang menampung komponen navbar yang ditandai dengan kotak berwarna biru.  
 32       Di dalam navbar tersebut, terdapat sebuah *tag* <button> untuk memunculkan *sidemenu*,  
 33       <ion-icon> untuk menampilkan icon, dan *tag* <ion-title> sebagai judul dari halaman.

```
34
35 1 <ion-header>
36 2   <ion-navbar>
37 3     <button ion-button menuToggle>
38 4       <ion-icon name="menu"></ion-icon>
39 5     </button>
40 6     <ion-title>Home</ion-title>
41 7   </ion-navbar>
42 8 </ion-header>
```

Kode 3.11: *Header* pada home.html

- 44       • *Content*

Content pada halaman *home* dengan tag `<ion-content>` (Kode 3.12 pada gambar 3.2c) ditandai dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat beberapa tag lainnya. Pertama yaitu sebuah tag `<ion-refresher>` yang digunakan untuk menampilkan simbol *refresh* saat pengguna menyegarkan halaman dengan cara melakukan *swipe* dari atas ke bawah layar, ditandai dengan kotak berwarna hijau. Lalu terdapat tag `<ion-card>` yang digunakan sebagai tempat untuk pengumuman terkait acara WSDC 2017 Bali disimpan. Penggunaan *card* ditantai dengan kotak berwarna merah muda. Di dalam tag `<ion-card>` terdapat tag `<ion-grid>` untuk mengatur tata letak dari penyusunan isi dari suatu *card*. Di dalam *grid* tersebut terdapat satu baris dengan tag `<ion-row>` dan dua kolom dengan tag `<ion-col>`. Kolom pertama ditandai dengan kotak berwarna coklat, berisi tanggal beserta pengumuman, dan kolom kedua ditandai dengan kotak berwarna oranye berisi gambar.

Selanjutnya terdapat sebuah tag `<ion-list>` untuk menyimpan berita-berita terkait acara WSDC 2017 Bali, yang ditandai dengan warna ungu. Di dalam *list* tersebut terdapat tag `<ion-list-header>` sebagai judul dari *list*, dan tag `<ion-item>` untuk menyimpan berita-berita terkait acara WSDC 2017 Bali. Di dalam tag `<ion-item>` terdapat tag `<button>` yang apabila ditekan oleh pengguna, maka akan mengarahkan pengguna untuk melihat berita tertentu sesuai dengan *item* yang dipilih dengan memanggil *method* `launch()` yang ada di *home.ts*.

```

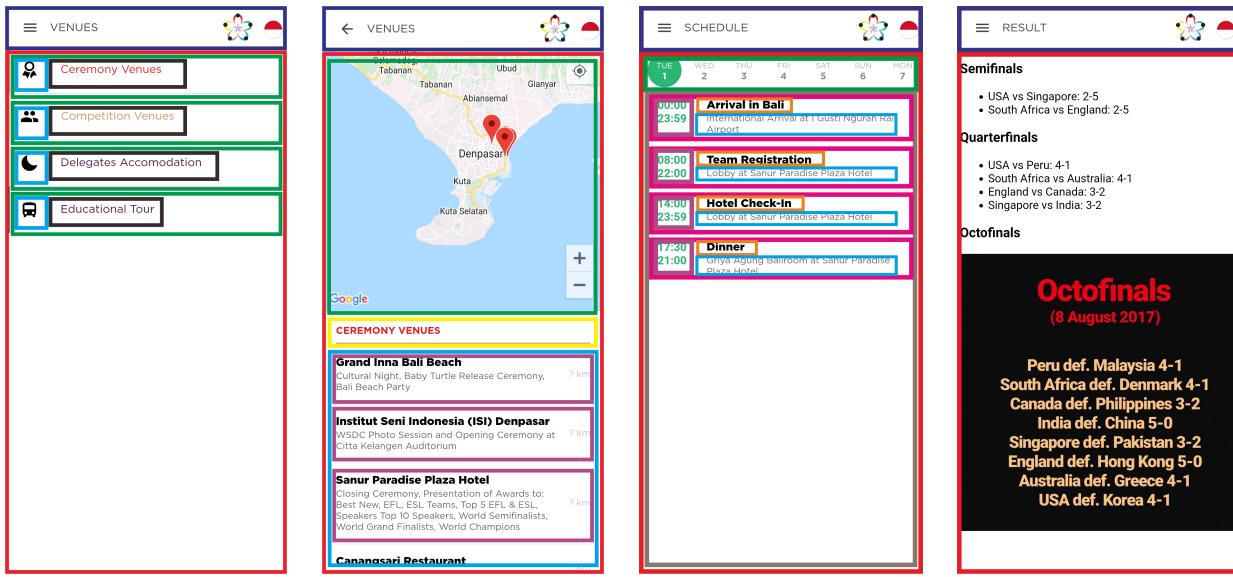
1 <ion-content>
2   <ion-refresher (ionRefresh)="doRefresh($event)">
3     <ion-refresher-content pullingIcon="arrow-dropdown" pullingText="Pull
4       to refresh" refreshingSpinner="circles" refreshingText="Refreshing...
5     ">
6     </ion-refresher-content>
7   </ion-refresher>
8   <ion-card (click)="onAnnouncementClick()">
9     <ion-grid>
10    <ion-row>
11      <ion-col col-9>
12        <ion-card-header text-wrap>
13          Latest Announcement
14        </ion-card-header>
15        <ion-card-content>
16          <h3>{{formatDatetime(wsdcData?.announcements[0].localtime)
17            }}</h3>
18          <p>{{wsdcData?.announcements[0].message}}</p>
19        </ion-card-content>
20      </ion-col>
21      <ion-col col-3>
22        
23      </ion-col>
24    </ion-row>
25  </ion-grid>
26 </ion-card>
27 <ion-list>
28   <ion-list-header>
29     <ion-icon ios="ios-book-outline" md="md-book"></ion-icon>
30   </ion-list-header>
31   <ion-item>
32     <ion-icon ios="ios-book-outline" md="md-book"></ion-icon>
33     Latest Announcement
34   </ion-item>
35 </ion-list>
36 </ion-content>
37 
```

```

1   27      Newsletters
2   28    </ion-list-header>
3   29    <ion-item *ngFor="let wsdcNews of wsdcData?.newsletters">
4   30      
6   32      <h2 text-wrap>{{wsdcNews.title}}</h2>
7   33      <button ion-button full block color="danger" (click)="launch(
8   34          wsdcNews.url)">Read More</button>
9   35    </ion-item>
10 36  </ion-list>
11 37  </ion-content>

```

Kode 3.12: Content pada home.html



Gambar 3.3: Wireframe Aplikasi WSDC 2017 Bali

#### 4. Komponen Info

Komponen ini digunakan untuk menampilkan halaman Info pada aplikasi. Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam *file* info.ts terdapat sebuah *decorator* @Component untuk komponen (Kode 3.13). Di dalam decorator ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta **templateUrl** untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template HTML* yang digunakan adalah *file* info.html.

```

1  @Component({
2    selector: 'page-info',
3    templateUrl: 'info.html'
4  })

```

Kode 3.13: @Component pada info.ts

Terdapat kelas InfoPage pada komponen info. Kelas ini hanya berisi *constructor* yang digunakan untuk menginisialisai halaman yang akan digunakan. *Constructor* sendiri berfungsi

untuk memuat data info dari penyimpanan dan memasukannya ke dalam sebuah variabel lokal. Selain itu, terdapat *file* info.html yang digunakan untuk menampilkan tata letak dari halaman info. Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan ke dalam halaman info. Diantaranya adalah sebagai berikut:

- *Header*

Halaman info memiliki *header* dengan tag `<ion-header>` (Kode 3.14) seperti pada gambar 3.2d. Tag tersebut merupakan komponen *parent* yang menampung komponen navbar yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat sebuah tag `<button>` untuk memunculkan sidemenu dan tag `<ion-icon>` untuk menampilkan icon dari tombol pada tag *button*. Terdapat tag `<ion-title>` sebagai judul dari halaman, yaitu “Info”.

```

1 <ion-header>
2   <ion-navbar>
3     <button ion-button menuToggle>
4       <ion-icon name="menu"></ion-icon>
5     </button>
6     <ion-title>Info</ion-title>
7   </ion-navbar>
8 </ion-header>

```

Kode 3.14: *Header* pada info.html

- *Content*

*Content* pada halaman info memiliki tag `<ion-content>` (Kode 3.15) yang pada gambar 3.2d dengan kotak berwarna merah. Di dalam tag info terdapat tag `<ion-grid>` untuk mengatur *layout* dari *content*. Di dalam tag `<ion-grid>` terdapat sebuah tag `<ion-row>` yang berisi sebuah tag `<div>`. Tag tersebut berisi info yang di dapatkan pada *constructor* di *file* info.ts.

```

1 <ion-content>
2   <ion-grid>
3     <ion-row>
4       <div [innerHTML]=wsdcInfoData>
5       </div>
6     </ion-row>
7   </ion-grid>
8 </ion-content>

```

Kode 3.15: *Content* pada info.html

## 5. Komponen *Result*

Komponen ini digunakan untuk menampilkan halaman *Result* pada aplikasi. Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam *file* result.ts terdapat sebuah *decorator* `@Component` untuk komponen (Kode 3.16) dan *decorator* `@ViewChild` (Kode 3.21). Di dalam *decorator* ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta `templateUrl` untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template HTML* yang digunakan adalah *file* info.html. `@ViewChild` digunakan untuk memanggil elemen dari DOM untuk memanggil komponen API ke dalam

1 TypeScript, yaitu pada komponen *result* adalah resultIFrame yang berada di file result.html.

```

2
3   1 @Component({
4     2   selector: 'page-result',
5     3   templateUrl: 'result.html'
6   4 })

```

Kode 3.16: @Component pada result.ts

```

8
9   1 @ViewChild('resultIFrame') resultIFrame: ElementRef;
10

```

Kode 3.17: @ViewChild pada result.ts

11 Terdapat kelas ResultPage dengan beberapa *method* yang digunakan, yaitu:

- 12 • ionViewDidLoad()

13     *Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman *result* telah  
 14     dimuat. *Event* ini hanya berjalan satu kali, jadi jika sudah melakukan cache terhadap  
 15     halaman ini, maka ionViewDidLoad() tidak akan berjalan lagi. *Lifecycle event* ini  
 16     berguna untuk melakukan pengaturan awal, yaitu untuk memuat data *result* yang sudah  
 17     disimpan di dalam penyimpanan internal. Setelah berhasil memuat data *result*, data  
 18     tersebut akan dimasukan ke dalam *child* resultIFrame. Terakhir, akan dipanggil *method*  
 19     presentLoading().

- 20 • presentLoading()

21     *Method* ini berfungsi untuk menampilkan sebuah *overlay* yang menunjukkan sebuah pesan  
 22     dan indikator pemuatan saat pertama kali halaman *draw* dimuat. Karena *overlay* ini  
 23     muncul di atas konten aplikasi, maka aktivitas pengguna akan diblokir untuk sementara  
 24     sampai seluruh halaman dimuat, yaitu sampai *method* onDrawIframeLoad() selesai.

- 25 • onResultIframeLoad()

26     *Method* ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam tag  
 27     <iframe> pada result.html yaitu *event* (load). *Method* ini berfungsi untuk menampilkan  
 28     data yang telah diambil yang disimpan di dalam *child* resultIFrame.

29 Selain itu, terdapat file result.html yang digunakan untuk menampilkan tata letak dari  
 30 halaman *result*. Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang  
 31 diimplementasikan ke dalam halaman *result*. Diantaranya adalah sebagai berikut:

- 32 • *Header*

33     Halaman *result* memiliki *header* dengan tag <ion-header> (Kode 3.18) seperti pada  
 34     gambar 3.3d. Tag tersebut merupakan komponen *parent* yang menampung komponen  
 35     navbar yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat  
 36     sebuah tag <button> untuk memunculkan *sidemenu*, dan <ion-icon> untuk menam-  
 37     pilkan icon dari tombol pada tag button. Terdapat tag <ion-title> sebagai judul dari  
 38     halaman, yaitu “Result”.

```

1 1 <ion-header>
2   2   <ion-navbar>
3     3     <button ion-button menuToggle>
4       4       <ion-icon name="menu"></ion-icon>
5     5     </button>
6       6     <ion-title>Result</ion-title>
7     7   </ion-navbar>
8   8 </ion-header>
9
10

```

Kode 3.18: *Header* pada result.html

11 • *Content*

12 *Content* pada halaman result memiliki tag `<ion-content>` (Kode 3.19) yang pada gam-  
 13 bar 3.3d dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat tag  
 14 `<iframe>`. Tag tersebut berisi informasi mengenai daftar pemenang acara WSDC 2017  
 15 bali yang di dapatkan pada method `onResultIframeLoad()` di kelas `ResultPage` pada file  
 16 `result.ts`.

```

17 1 <ion-content>
18 2   <iframe #resultIFrame (load)="onResultIframeLoad()" class="iframe-
19   2   fullscreen"></iframe>
20 3 </ion-content>
21
22

```

Kode 3.19: *Content* pada result.html

23 6. Komponen *Schedule*

24 Komponen ini digunakan untuk menampilkan halaman *Schedule* pada aplikasi. Komponen  
 25 ini memiliki sebuah file TypeScript untuk mengatur keseluruhan halaman. Di dalam file  
 26 `schedule.ts` terdapat sebuah *decorator* `@Component` untuk komponen (Kode 3.20) dan dua  
 27 *decorator* `@ViewChild` (Kode 3.21). Di dalam *decorator* ini terdapat CSS *selector* untuk  
 28 memilih CSS yang akan digunakan, serta `templateUrl` untuk mendefinisikan eksternal  
 29 HTML *template* yang akan digunakan. *Template* HTML yang digunakan adalah file `info.html`.  
 30 `@ViewChild` digunakan untuk memanggil elemen dari DOM untuk memanggil komponen API  
 31 ke dalam TypeScript, yaitu pada komponen `result` adalah `scheduleSlider` dan `segmentContainer`  
 32 yang berada di file `result.html`. `scheduleSlider` berfungsi untuk menyimpan konten dari sebuah  
 33 *slide*. Sedangkan `segmentContainer` berfungsi untuk menyimpan konten dari sebuah *segment*.

```

34 1 @Component({
35   2   selector: 'page-schedule',
36   3   templateUrl: 'schedule.html'
37 4 })
38

```

Kode 3.20: `@Component` pada schedule.ts

```

40 1 @ViewChild('scheduleSlider') slider: Slides;
41 2 @ViewChild('segmentContainer') segmentContainer: ElementRef;
42

```

Kode 3.21: `@ViewChild` pada schedule.ts

1 Terdapat kelas SchedulePage pada schedule.ts yang ini memiliki satu *constructor*. *Constructor*  
 2 kelas ini berfungsi untuk mengambil data *result* yang berada di dalam penyimpanan internal.  
 3 Data tersebut kemudian disimpan ke dalam variabel lokal schedules. Kemudian akan mengatur  
 4 *segment* yang aktif pada saat pertama kali halaman *result* dibuka, yaitu *segment* yang pertama  
 5 dan menampilkan *slide* pertama yang berisi jadwal pada hari pertama. Selain itu, terdapat  
 6 beberapa *method* yang digunakan, yaitu:

- 7 • onSegmentChanged(segmentButton)

8     *Method* ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam tag  
 9     `<ion-segment>` pada schedule.html yaitu *event* (`ionChange`). (`ionChange`) merupakan  
 10    *event* yang dimiliki oleh UI Component ion-segment milik Ionic Framework. *Method*  
 11    ini digunakan ketika pengguna memilih *segment* pada tag `<ion-segment>` di dalam file  
 12    schedule.html. *Method* ini memiliki sebuah parameter segmentButton yang berisi *event*  
 13    dari sebuah segment. *Child component* slides kemudian akan diubah sesuai dengan *value*  
 14    yang ada pada parameter, yaitu hari yang sedang aktif.

- 15 • onSlideChanged()

16     *Method* ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam tag  
 17     `<ion-slides>` pada schedule.html yaitu *event* (`ionSlideDidChange`). (`ionSlideDidChange`)  
 18    merupakan *event* yang dimiliki oleh UI Component ion-slides milik Ionic Framework.  
 19    *Method* ini berfungsi untuk berpindah antar *slides* saat pengguna menggeser *slides*  
 20    tersebut ke kanan atau ke kiri layar.

- 21 • getDayName(sqlDate: string)

22     *Method* ini berfungsi untuk mengembalikan nama hari dari parameter.

- 23 • getDate(sqlDate: string)

24     *Method* ini bergunsi untuk mengembalikan tanggal dari parameter.

25 Selain itu, terdapat file schedule.html yang digunakan untuk menampilkan halaman *schedule*.  
 26 Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan  
 27 ke dalam halaman *schedule*. Diantaranya adalah sebagai berikut:

- 28 • *Header*

29     Halaman *schedule* memiliki *header* dengan tag `<ion-header>` (Kode 3.22) seperti pada  
 30    gambar 3.3c. Tag tersebut merupakan komponen *parent* yang menampung komponen  
 31    navbar yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat  
 32    sebuah tag `<button>` untuk memunculkan *sidemenu* dan `ion-icon` untuk menampilkan  
 33    icon dari tombol pada tag *button*. Dan juga terdapat tag `<ion-title>` sebagai judul  
 34    dari halaman, yaitu “Schedule”.

```
35
36 1 <ion-header>
37 2   <ion-navbar>
38 3     <button ion-button menuToggle>
39 4       <ion-icon name="menu"></ion-icon>
40 5     </button>
41 6     <ion-title>Schedule</ion-title>
42 7   </ion-navbar>
43 8 </ion-header>
```

Kode 3.22: *Header* pada schedule.html

1     • *Content*

2     *Content* pada halaman result memiliki tag `<ion-content>` (Kode 3.23) yang pada gam-  
 3     bar 3.3c dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat dua buah  
 4     tag `<div>` yang masing masing berisi tag `<ion-segment>` dan tag `<ion-slides>`. Tag  
 5     `<ion-segment>` digunakan untuk tampilan hari, seperti pada gambar 3.3c yang ditandai  
 6     dengan kotak berwarna hijau. Lalu, tag `<ion-slides>` digunakan untuk tampilan jadwal  
 7     di dalam satu hari, seperti yang ditandai dengan kotak berwarna coklat.  
 8

9     Setiap jadwal yang berada di tag `<ion-slides>` dibungkus dengan tag `<ion-list>`  
 10    seperti pada kotak berwarna merah muda di gambar 3.3c. Dalam satu tag `<ion-list>`  
 11    terdapat tag `<ion-note>` yang berisi waktu mulai dan waktu selesai dari satu jadwal  
 12    seperti yang ditandai dengan kotak berwarna ungu, tag `<h3>` berisi nama acara seperti  
 13    yang ditandai dengan kotak berwarna oranye, dan tag `<p>` yang berisi tempat acara  
 tersebut diadakan ditandai dengan kotak berwarna biru muda.

```

14
15   1 <ion-content>
16   2   <div id="schedulesContainer">
17   3     <div id="schedulesSegments">
18   4       <ion-segment #segmentContainer *ngIf="schedules" [(ngModel)]="selectedSegmentIdx" (ionChange)="onSegmentChanged($event)">
19   5         <ion-segment-button *ngFor="let schedule of schedules; let i = index" [value]="i">
20   6           <div class="day">{{getDayName(schedule.date)}}</div>
21   7           <div class="date">{{getDate(schedule.date)}}</div>
22   8           </ion-segment-button>
23   9         </ion-segment>
24  10       </div>
25  11     <div id="schedulesSlides">
26  12       <ion-slides #scheduleSlider (ionSlideDidChange)="onSlideChanged()">
27  13         <ion-slide *ngFor="let schedule of schedules">
28  14           <ion-list>
29  15             <ion-item text-wrap *ngFor="let agenda of schedule.agenda">
30  16               <ion-note item-start>
31  17                 {{agenda.start}}<br/>
32  18                 {{agenda.end}}
33  19               </ion-note>
34  20               <h3>{{agenda.title}}</h3>
35  21               <p>{{agenda.subtitle}}</p>
36  22             </ion-item>
37  23           </ion-list>
38  24         </ion-slide>
39  25       </ion-slides>
40  26     </div>
41  27   </div>
42  28 </ion-content>
43
44
45

```

Kode 3.23: *Content* pada schedule.html

1    7. Komponen *Venues*

2    Komponen ini digunakan untuk menampilkan halaman *Venues* pada aplikasi. Komponen ini  
 3    memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam *file* *venues.ts*  
 4    terdapat sebuah *decorator* `@Component` untuk komponen (Kode 3.24). Di dalam decorator  
 5    ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta **templateUrl**  
 6    untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template* HTML yang  
 7    digunakan adalah *file* *venues.html*

```
8
9   @Component({
10     selector: 'page-venues',
11     templateUrl: 'venues.html'
12   })
```

Kode 3.24: `@Component` pada *venues.ts*

14   Terdapat kelas *VenuesPage* yang memiliki satu *constructor*. *Constructor* kelas ini berfungsi  
 15   untuk mengambil data *venues* yang berada di dalam penyimpanan internal. Data tersebut  
 16   kemudian disimpan ke dalam variabel lokal *valVenues*. Selain itu, terdapat sebuah *method*  
 17   *itemTapped()* yang berfungsi untuk berpindah halaman ke halaman *venues-map*, yang akan  
 18   menampilkan peta lokasi berlangsungnya acara, sesuai dengan *venues* yang dipilih.

19   Selain itu, terdapat *file* *venues.html* yang digunakan untuk menampilkan halaman *venues*.  
 20   Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan  
 21   ke dalam halaman *venues*. Diantaranya adalah sebagai berikut:

- 22   • *Header*

23   Halaman *venues* memiliki *header* dengan *tag* `<ion-header>` (Kode 3.25) seperti pada  
 24   gambar 3.3a. *Tag* tersebut merupakan komponen *parent* yang menampung komponen  
 25   *navbar* yang ditandai dengan kotak berwarna biru. Di dalam *navbar* tersebut, terdapat  
 26   sebuah *tag* `<button>` untuk memunculkan *sidemenu*. Lalu terdapat *tag* `<ion-title>`  
 27   sebagai judul dari halaman, yaitu “Venues”.

```
28
29   1 <ion-header>
30   2   <ion-navbar>
31   3     <button ion-button menuToggle>
32   4       <ion-icon name="menu"></ion-icon>
33   5     </button>
34   6     <ion-title>Venues</ion-title>
35   7   </ion-navbar>
36   8 </ion-header>
```

Kode 3.25: *Header* pada *venues.html*

- 38   • *Content*

39   *Content* pada halaman *venues* memiliki *tag* `<ion-content>` (Kode 3.26) yang pada gam-  
 40   bar 3.3a dengan kotak berwarna merah. Di dalam *tag* `<ion-content>` terdapat sebuah  
 41   *tag* `<ion-grid>` dan sebuah *tag* `<ion-row>`. Di dalam *tag* `<ion-row>` terdapat sebuah  
 42   *tag* `<ion-list>` yang berisi *tag* `<ion-button>` yang ditandai dengan kotak berwarna  
 43   hijau pada gambar 3.3a. Masing-masing *tag* `<ion-button>` berisi *tag* `<ion-icon>` yang  
 44   ditandai dengan kotak berwarna biru muda, dan *tag* `<span>` berisi nama *venues* yang  
 45   ditandai dengan kotak berwarna hitam.

```

1      1 <ion-content>
2          2   <ion-grid>
3              3     <ion-row>
4                  4       <ion-list style="width: 100%;" no-lines>
5                     5         <button ion-item id="{{wsdcVenue.id}}" *ngFor="let wsdcVenue
6                         of venuesData" (click)="itemTapped($event, wsdcVenue)">
7                         6             <ion-icon ios="ios-{{wsdcVenue.icon}}-outline" md="md-{{
8                             wsdcVenue.icon}}" item-start></ion-icon>
9                         7                 <span>{{wsdcVenue.name}}</span>
10                        8             </button>
11                    9         </ion-list>
12                10     </ion-row>
13            11   </ion-grid>
14        12 </ion-content>
15
16

```

Kode 3.26: *Content* pada venues.html

Terdapat kelas VenuesPage pada venues.ts. Kelas ini memiliki satu *constructor*. *Constructor* kelas ini berfungsi untuk mengambil data *venues* yang berada di dalam penyimpanan. Data tersebut kemudian disimpan ke dalam variabel lokal *venuesData*, yang berisi *id*, *name*, *icon*, *geojson*, dan *colorIdx*. Selain itu, terdapat sebuah *method* yaitu *itemTapped(event, wsdcVenue)*. *Method* ini memiliki dua buah parameter, *event* yang berisi *event* pada *tag button*, dan *wsdcVenue* yang merupakan data bertipe json yang berisi data lengkap sebuah venue yang ada di penyimpanan sesuai dengan data venue pada *event* di dalam *tag button*. Kemudian, dengan menggunakan NavController milik Ionic Framework, data *wsdcVenue* dikirimkan ke halaman Venues Map. Setelah itu halaman akan berpindah ke halaman Venues Map.

#### 8. Komponen *Venues Map*

Komponen ini digunakan untuk menampilkan halaman *Venues Map* pada aplikasi. Berbeda dengan komponen *venues*, komponen *Venues Map* menampilkan sebuah peta yang berisi lokasi dari acara WSDC 2017 Bali. Komponen ini memiliki sebuah file TypeScript untuk mengatur keseluruhan halaman. Di dalam file *venues\_map.ts* terdapat sebuah *decorator* `@Component` untuk komponen (Kode 3.27). Di dalam *decorator* ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta **templateUrl** untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template HTML* yang digunakan adalah *file* *venues\_map.html*.

```

1 @Component({
2   selector: 'page-venuesmap',
3   templateUrl: 'venues_map.html',
4 })

```

Kode 3.27: `@Component` pada *venues\_map.ts*

Terdapat kelas VenuesMapPage di dalam app.module.ts. Kelas ini memiliki sebuah *constructor*. *Constructor* kelas ini berfungsi untuk mengambil data *venues* yang berada di dalam penyimpanan internal, kemudian disimpan ke dalam variabel lokal *venuesData*. Selain itu, di dalam *constructor* juga mengambil data yang dikirimkan oleh halaman Venues. Data tersebut kemudian dimasukkan ke dalam variabel lokal beranam *items*.

1 Pada komponen ini, terdapat sebuah *plugin* Google Maps, yang digunakan untuk menampilkan  
2 peta yang berisi lokasi dari kegiatan WSDC 2017 Bali. *Plugin* yang digunakan adalah *plugin*  
3 Google Maps yang disediakan oleh Cordova. *Plugin* tersebut diinisialisasikan di dalam  
4 *constructor*. Selain itu, terdapat pula sebuah *plugin* Geolocation, yang berfungsi untuk  
5 menerima masukan posisi dari lokasi pengguna yang berisi *latitude* dan *longitude*, yang  
6 kemudian keseluruhan lokasi tersebut pada *constructor* akan dihitung jaraknya dari lokasi  
7 pengguna saat ini.

8 Terdapat beberapa *method* yang digunakan, diantaranya yaitu:

- 9 • ngAfterViewInit()

10 *Method* ini dipanggil hanya sekali ketika Angular menyelesaikan inisialisasi tampilan  
11 komponen. *Method* ini digunakan untuk menambahkan atribut ke dalam judul dari  
12 halaman, yaitu menambahkan warna pada teks judul.

- 13 • loadMap()

14 *Method* ini dipanggil di dalam *constructor*, dan berfungsi untuk menampilkan peta  
15 dengan bantuan *plugin* Google Maps. Pada *method* ini, peta pertama kali akan dibuat  
16 dengan pengaturan kamera yang mengarah ke lokasi latitude dan longitude dari kota  
17 Kuta, Bali. *Plugin* Google Maps sendiri akan memanfaatkan fitur-fitur *native* dari suatu  
18 perangkat. Fitur-fitur tersebut adalah untuk melakukan *gesture* seperti *scroll*, *tilt*, *rotate*,  
19 dan *zoom*. Selain itu terdapat fitur untuk menagkses kontrol pada Google Maps, seperti  
20 mengakses kompas, tombol lokasi pengguna saat ini, dan melihat peta di dalam ruangan.  
21 Saat Google Maps sudah tersedia untuk digunakan, *method* ini akan memanggil *method*  
22 *loadMarkers()* untuk membuat penanda pada peta.

- 23 • loadMarkers()

24 *Method* ini dipanggil oleh *method* *loadMap()*. *Method* ini digunakan untuk menampilkan  
25 *marker* dari setiap lokasi acara WSDC 2017 Bali yang sudah tersimpan di dalam variabel  
26 items. Google Maps menggunakan lokasi latitude dan longitude dari suatu lokasi yang  
27 berada di variabel items untuk membuat *marker*.

- 28 • featTapped(event, index)

29 *Method* ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam *tag*  
30 *<ion-item>* pada *venues\_map.html* yaitu *event* (*click*). Saat pengguna melakukan klik  
31 di dalam *tag* *<ion-item>*, maka peta akan melakukan *zoom* sesuai dengan lokasi yang  
32 ada pada *tag* *<ion-item>*.

- 33 • computeDistance(p1, p2)

34 *Method* ini digunakan untuk menghitung jarak antara pengguna ke lokasi *venues*. *Method*  
35 ini memanfaatkan fitur dari *plugin* Google Maps, yaitu *computeDistanceBetween* dengan  
36 parameter lokasi *venues* dan lokasi perangkat pengguna yang didapatkan dari paramter  
37 *method* ini.

38 Selain itu, terdapat *file* *venues\_map.html* yang digunakan untuk menampilkan halaman  
39 *venues map*. Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang  
40 diimplementasikan ke dalam halaman. Diantaranya adalah sebagai berikut:

1     • *Header*

2     Halaman *venues* memiliki *header* dengan tag `<ion-header>` (Kode 3.28) seperti pada  
 3     gambar 3.3b. Tag tersebut merupakan komponen *parent* yang menampung komponen  
 4     navbar seperti yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut,  
 5     terdapat sebuah tag `<button>` untuk memunculkan *sidemenu* dan `<ion-icon>` untuk  
 6     menampilkan icon. Lalu terdapat tag `<ion-title>` sebagai judul dari halaman, yaitu  
 7     “Venues”.

```
8
9   1 <ion-header>
10  2   <ion-navbar>
11  3     <button ion-button menuToggle>
12  4       <ion-icon name="menu"></ion-icon>
13  5     </button>
14  6     <ion-title>Venues</ion-title>
15  7   </ion-navbar>
16  8 </ion-header>
```

Kode 3.28: *Header* pada *venues\_map.html*

18     • *Content*

19     *Content* pada halaman *venues* dibungkus oleh tag `<ion-content>` (Kode 3.29) yang pada  
 20     gambar 3.3a dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat  
 21     sebuah tag `<div>` dengan id bernilai *map*, untuk menampilkan peta lokasi dari *venues*  
 22     seperti yang ditandai dengan kotak berwarna hijau pada gambar 3.3b. Untuk judul dari  
 23     *venues* ditandai dengan kotak berwarna kuning dengan menggunakan tag `<h3>`. Selain  
 24     itu terdapat sebuah tag `<ion-scroll>` seperti yang ditandai dengan kotak berwarna biru  
 25     muda, berfungsi untuk menampilkan sebuah konten yang dapat digulir. Di dalam tag  
 26     `<ion-scroll>` terdapat sebuah tag `<ion-list>` dan `<ion-item>` seperti yang ditandai  
 27     dengan kotak berwarna ungu, berisi nama, deskripsi, serta jarak pengguna ke tempat  
 28     *venues* berada. Tag `<ion-item>` akan melakukan perulangan dengan menggunakan  
 29     `*ngFor` yang tersedia pada Angular untuk menampilkan daftar *venues*.

```
30
31  1 <ion-content>
32  2   <div #map id="map"></div>
33  3   <h3 #pagetitle>
34  4     {{selectedItem.name}}
35  5   </h3>
36  6   <ion-scroll scrollY="true">
37  7     <ion-list>
38  8       <ion-item text-wrap *ngFor="let feature of items; let i=index" (
39  9         click)="featTapped($event, i)">
40 10         <h2>{{feature.name}}</h2>
41 11         <p>{{feature.description}}</p>
42 12         <ion-note item-end>
43 13           {{feature.distance}}
44 14         </ion-note>
45 15       </ion-item>
46 16     </ion-list>
47 17   </ion-scroll>
48 </ion-content>
```

1

---

Kode 3.29: *Content* pada venues\_map.html

2 **3.2 Analisis Sistem Usulan**

3 Aplikasi yang ada pada saat ini menggunakan Ionic Framework versi 3, yang sudah tidak lagi  
4 didukung oleh Ionic. Maka dari itu, aplikasi WSDC 2017 Bali akan dibangun ulang menggunakan  
5 Ionic Framework versi terbaru saat ini, yaitu Ionic Framework versi 6. Proses untuk melakukan  
6 pembangunan ulang aplikasi dari Ionic Framework versi 3 ke Ionic Framework versi 6 telah dijelaskan  
7 pada sub bab 2.3.3. Pada sub bab ini akan dijelaskan analisis untuk pengembangan kebutuhan  
8 aplikasi WSDC 2017 Bali agar aplikasi tersebut dapat berjalan menggunakan Ionic Framework versi 5.

9 **3.2.1 Analisis Kebutuhan**

10 Aplikasi WSDC 2017 Bali yang akan dibangun akan mengadopsi desain dan tata letak yang sama  
11 persis dengan aplikasi WSDC 2017 Bali saat ini. Namun dengan perubahan penggunaan Ionic  
12 Framework yang digunakan, yaitu versi 6, serta Angular versi 12. Pada Ionic Framework terbaru  
13 saat skripsi ini dibuat, aplikasi WSDC 2017 Bali yang akan dibangun akan memanfaatkan fasilitas  
14 yang disediakan oleh Ionic Framework, yaitu UI Component, dan CSS Utilities.

15 Struktur yang dibuat akan menggunakan struktur Ionic Framework versi 6, namun menggunakan  
16 komponen-komponen yang sama dengan Ionic Framework versi 3. Tapi, karena terdapat beberapa  
17 perubahan, maka perubahannya di dalam komponen seperti UI Component, dan CSS akan mengikuti  
18 Ionic Framework versi 6.

19 UI Component yang akan digunakan akan mengikuti perkembangan pada Ionic Framework  
20 versi 6. Pada setiap komponen, akan terdapat sebuah *header* dengan tag `<ion-header>`. Tag  
21 tersebut akan membungkus tag `<ion-toolbar>` sebagai *toolbar* dari aplikasi. Tag ini akan meng-  
22 gantikan tag `<ion-navbar>` yang telah dihapus sejak Ionic versi 4. Didalamn `<ion-toolbar>`  
23 terdapat tag `<ion-buttons>` sebagai pengganti tag `<button>` yang dihapus sejak Ionic versi 4,  
24 dan tag `<ion-title>`. Dibandingkan dengan aplikasi sistem kini, terdapat perubahan pada tag  
25 `<ion-toolbar>` yang semula bernama `<ion-navbar>` pada Ionic Framework versi 3. Selain itu  
26 ada tag `<ion-buttons>` yang semula bernama `<button>`. Di dalam tag `<ion-buttons>` akan ada  
27 sebuah tag `<ion-menu-button>`.

28 Selain itu, terdapat UI Component lain yang akan diterapkan ke dalam masing-masing komponen  
29 di dalam aplikasi yang akan dibangun, diantaranya adalah sebagai berikut :

30 1. *Announcement*

31 Pada komponen *announcement*, terdapat beberapa UI Component yang akan diimplementa-  
32 sikan, diantaranya adalah sebagai berikut:

33 • *Content*

34 Komponen ini akan digunakan sebagai penyedia area konten yang digunakan untuk meng-  
35 ontrol area yang dapat digulir dan menampilkan isi konten dari halaman *announcement*.

36 UI Component *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak  
37 mengalami perubahan dari Ionic Framework versi 3.

1     • *Refresher*

2       *Refresher* menyediakan fungsionalitas pull-to-refresh pada komponen *content*. UI Component *Refresher* dengan tag `<ion-refresher>` dan `<ion-refresher-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

5     • *List*

6       *List* dengan tag `<ion-list>` akan terdiri dari beberapa baris item `<ion-item>` yang  
7       berisi label `<ion-label>`. UI Component *List* dengan tag `<ion-list>`, `<ion-item>`  
8       dan `<ion-label>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic  
9       Framework versi 3.

10    • *Item*

11       *Item* dengan tag `<ion-item>` sejak Ionic 4 mengalami perubahan dibandingkan pada  
12       Ionic 3, yaitu wajib menambahkan *label* dengan tag `<ion-label>`. Pada aplikasi WSDC  
13       2017 Bali saat ini yang menggunakan Ionic 3, tidak terdapat tag `<ion-label>` pada  
14       `<ion-item>` (Kode 3.30). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi  
15       yang akan dibangun yang menggunakan Ionic 6, akan menggunakan tag `<ion-label>` di  
16       dalam `<ion-item>` (Kode 3.31).

```
1 <ion-item text-wrap *ngFor="let announcement of announcements">
2   <h3>{{formatDatetime(announcement.localtime)}}</h3>
3   <p>{{announcement.message}}</p>
4 </ion-item>
```

Kode 3.30: Tag `<ion-item>` dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```
23 1 <ion-item color="wsdc-blue" *ngFor="let announcement of announcements;">
24   2   let i = index">
25   3     <ion-label>
26   4       <h3>{{ formatDatetime(announcement.localtime) }}</h3>
27   5       <p>{{ announcement.message }}</p>
28   6     </ion-label>
29   7   </ion-item>
```

Kode 3.31: Tag `<ion-item>` dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan dibuat

32 2. *Draw*

33       Pada komponen *draw*, terdapat sebuah UI Component, yaitu *Content*. Komponen *content*  
34       akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area yang  
35       dapat digulir dan menampilkan isi konten dari halaman *draw*. UI Component *Content* dengan  
36       tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic  
37       Framework versi 3.

38 3. *Home*

39       Pada komponen *home*, terdapat beberapa file yaitu:

40     • Content

41       Komponen ini akan digunakan sebagai penyedia area konten yang digunakan untuk  
42       mengontrol area yang dapat digulir dan menampilkan isi konten dari halaman *home*.  
43       UI Component *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak  
44       mengalami perubahan dari Ionic Framework versi 3.

1     • *Refresher*

2       *Refresher* menyediakan fungsionalitas pull-to-refresh pada komponen *content*. UI Component *Refresher* dengan tag `<ion-refresher>` dan `<ion-refresher-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

3     • *Card*

4       Komponen ini akan digunakan sebagai tampilan antar muka, yang dapat menjadi titik masuk ke dalam informasi yang lebih detail. UI Component *Card* dengan tag `<ion-card>`, `<ion-card-title>` dan `<ion-card-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

5     • *Grid*

6       Komponen ini akan digunakan untuk membangun tata letak kustom pada halaman *home* bagian *announcement*, yang terdiri dari baris dan kolom. UI Component *Grid* dengan tag `<ion-grid>`, `<ion-row>` dan `<ion-col>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3

7     • *List Header*

8       *List Header* dengan tag `<ion-list-header>` sejak Ionic 4 diwajibkan untuk selalu menambahkan tag `<ion-label>`. Pada aplikasi WSDC 2017 Bali saat ini yang menggunakan Ionic 3, tidak terdapat tag `<ion-label>` pada `<ion-list-header>` (Kode 3.32). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi yang akan dibangun yang menggunakan Ionic 6, akan menggunakan tag `<ion-label>` di dalam `<ion-item-header>` (Kode 3.33).

```
23 1 <ion-list-header>
24 2   <ion-icon ios="ios-book-outline" md="md-book"></ion-icon>
25 3   Newsletters
26 4 </ion-list-header>
```

Kode 3.32: Tag `<ion-list-header>` dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```
28 1 <ion-list-header>
29 2   <ion-icon name="book-outline"></ion-icon>
30 3   <ion-label>Newsletters</ion-label>
31 4 </ion-list-header>
```

Kode 3.33: Tag `<ion-list-header>` dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan dibuat

34     • *Icon*

35       Komponen ini akan digunakan untuk menampilkan ikon pada halaman *home*. UI Component *List* dengan tag `<ion-icon>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

36     • *Button*

37       Di dalam halaman *home*, komponen ini merupakan sebuah komponen yang dapat diklik untuk mengarahkan pengguna ke URL yang berisi berita terkait WSDC 2017 Bali. Pada aplikasi WSDC 2017 Bali saat ini yang menggunakan Ionic 3, komponen ini ditulis menggunakan tag `<button>` (Kode 3.34). Sejak Ionic Framework versi 4, terjadi perubahan dengan mengganti tag tersebut menjadi `<ion-button>` pada aplikasi yang akan dibangun yang menggunakan Ionic 6 (Kode 3.35).

```

1   1 <button ion-button full block color="danger" (click)="launch(wsdcNews.url)
2     )">Read More</button>
3

```

Kode 3.34: *Button* dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```

5   1 <ion-button full block color="danger" (click)="launch(wsdcNews.url)">Read
6     More</ion-button>
7

```

Kode 3.35: *Button* dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan dibuat

#### 4. Info

Pada komponen *info*, terdapat beberapa UI Component yang akan diimplementasikan, diantaranya adalah sebagai berikut:

- *Content*

Komponen ini akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area yang dapat digulir dan menampilkan isi konten dari halaman *info*. UI Component *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

- *Grid*

Komponen ini akan digunakan untuk membangun tata letak kustom pada halaman *info*, yang terdiri dari baris. UI Component *Grid* dengan tag `<ion-card>` dan `<ion-row>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3

#### 5. Result

Pada komponen *Result*, terdapat sebuah UI Component, yaitu *Content*. Komponen *content* akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area konten dan menampilkan isi konten dari halaman *result*. UI Component *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

#### 6. Schedule

Pada komponen *schedule*, terdapat beberapa UI Component yang akan diimplementasikan, diantaranya adalah sebagai berikut:

- *Content*

Komponen ini akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area yang dapat digulir dan menampilkan isi konten dari halaman *schedule*. UI Component *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

- *Item*

*Item* dengan tag `<ion-item>` sejak Ionic 4 mengalami perubahan dibandingkan pada Ionic 3, yaitu wajib menambahkan *label* dengan tag `<ion-label>`. Pada aplikasi WSDC 2017 Bali saat ini yang menggunakan Ionic 3, tidak terdapat tag `<ion-label>` pada `<ion-item>` (Kode 3.36). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi yang akan dibangun yang menggunakan Ionic 6, akan menggunakan tag `<ion-label>` di dalam `<ion-item>` (Kode 3.37).

```

1   1 <ion-item text-wrap *ngFor="let agenda of schedule.agenda">
2     2   <ion-note item-start>
3     3     {{agenda.start}}<br/>
4     4       {{agenda.end}}
5     5   </ion-note>
6     6   <h3>{{agenda.title}}</h3>
7     7   <p>{{agenda.subtitle}}</p>
8     8 </ion-item>

```

Kode 3.36: Tag &lt;ion-item&gt; dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```

11 1 <ion-item *ngFor="let agenda of schedule.agenda;" >
12 2   <ion-note item-start>
13 3     {{agenda.start}}<br/>
14 4       {{agenda.end}}
15 5   </ion-note>
16 6   <ion-label class="ion-text-wrap">
17 7     <h3>{{agenda.title}}</h3>
18 8     <p>{{agenda.subtitle}}</p>
19 9   </ion-label>
20 10 </ion-item>
21

```

Kode 3.37: Tag &lt;ion-item&gt; dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan dibuat

- *List*

*List* berfungsi untuk menyimpan konten yang terdiri dari beberapa baris. *List* dengan tag <ion-list> akan terdiri dari beberapa baris item <ion-item> dan akan memiliki sebuah *header*. UI Component *List* dengan tag <ion-list>, dan <ion-item> pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

- *Segment*

Komponen ini akan digunakan untuk pengguna agar dapat berpindah tampilan di dalam halaman yang sama. Seperti pada tampilan halaman jadwal yang ada pada aplikasi WSDC 2017 Bali saat ini, dimana pengguna dapat berpindah hari untuk mengetahui jadwal kegiatan pada hari tertentu yang dipilih oleh pengguna, namun masih berada di halaman yang sama, yaitu halaman Schedule. UI Component *Segment* dengan tag <ion-segment> pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3. Sedangkan tag <ion-segment-button> yang berada di dalam tag <ion-segment> sejak Ionic 4 mengalami perubahan yaitu wajib menambahkan *label* dengan tag <ion-label>. Pada aplikasi WSDC 2017 Bali saat ini yang menggunakan Ionic 3, tidak terdapat tag <ion-label> pada <ion-item> (Kode 3.38). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi yang akan dibangun yang menggunakan Ionic 6, akan menggunakan tag <ion-label> di dalam <ion-item> (Kode 3.39).

```

42 1 <ion-segment-button *ngFor="let schedule of schedules; let i = index" [
43 2   value]="i">
44 3   <div class="day">{{getDayName(schedule.date)}}</div>
45 4   <div class="date">{{getDate(schedule.date)}}</div>
46 </ion-segment-button>

```

Kode 3.38: Tag &lt;ion-segment-button&gt; dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```

1 1 <ion-segment-button *ngFor="let schedule of schedules; let i = index" [
2   value]="i">
3   <ion-label>
4     <div class="day">{{getDayName(schedule.date)}}</div>
5     <div class="date">{{getDate(schedule.date)}}</div>
6   </ion-label>
7 </ion-segment-button>
8

```

Kode 3.39: Tag `<ion-segment-button>` dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan Dibuat

- *Slides*

Komponen ini akan digunakan sebagai wadah dari *multi-section*. Penggunaan slide di halaman *schedule* yaitu untuk berpindah jadwal perhari dengan cara melakukan *swipe* dari kanan ke kiri layar atau sebaliknya. UI Component *Slides* dengan tag `<ion-slides>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

## 7. *Venues*

Pada komponen *venues*, terdapat beberapa UI Component yang akan diimplementasikan, diantaranya adalah sebagai berikut:

- *Content*

Komponen Content dengan tag `<ion-content>` akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area yang dapat digulir dan menampilkan isi konten dari halaman *venues*. UI Component *Content* pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

- *Grid*

Komponen ini akan digunakan untuk membangun tata letak kustom pada halaman info. UI Component *Grid* dengan tag `<ion-card>` dan `<ion-row>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3

- *List*

*List* dengan tag `<ion-list>`, yang terdiri dari baris yang setiap barisnya berisi kategori *venues* yang disusun menggunakan *button*. UI Component *List* dengan tag `<ion-list>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

- *Button*

*Button* digunakan untuk berpindah ke halaman *Venues Map* sesuai dengan tombol apa yang dipilih. Pada aplikasi WSDC 2017 Bali saat ini yang menggunakan Ionic 3, komponen ini ditulis menggunakan tag `<button>` (Kode 3.40). Sejak Ionic Framework versi 4, terjadi perubahan dengan mengganti tag tersebut menjadi `<ion-button>` pada aplikasi yang akan dibangun yang menggunakan Ionic 6 (Kode 3.41).

```

1 <button ion-item id="{{wsdcVenue.id}}" *ngFor="let wsdcVenue of
2   venuesData" (click)="itemTapped($event, wsdcVenue)">
3   <ion-icon ios="ios-{{wsdcVenue.icon}}-outline" md="md-{{wsdcVenue.icon
4   }}> item-start</ion-icon>
5   <span>{{wsdcVenue.name}}</span>
6 </button>

```

Kode 3.40: *Button* dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```

1   1 <ion-button ion-item id="{{wsdcVenue.id}}" *ngFor="let wsdcVenue of
2     2   venuesData" (click)="itemTapped($event, wsdcVenue)">
3     3     <ion-icon ios="ios-{{wsdcVenue.icon}}-outline" md="md-{{wsdcVenue.icon
4       }}> item-start></ion-icon>
5     4     <span>{{wsdcVenue.name}}</span>
6   5   </ion-button>
7

```

Kode 3.41: *Button* dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan Dibuat

- *Icon*

Komponen ini akan digunakan untuk menampilkan ikon pada halaman *venues*. UI Component *Icon* dengan tag `<ion-icon>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

#### 8. *Venues Map*

Pada komponen *venues\_map*, terdapat beberapa UI Component yang akan diimplementasikan, diantaranya adalah sebagai berikut:

- *Content*

Komponen ini akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area yang dapat digulir dan menampilkan isi konten dari halaman *venues\_map*. UI Component *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

- *Scroll*

Tag `<ion-scroll>` telah dihapus sejak Ionic Framework versi 4, dan digantikan penggunaannya dengan hanya cukup menggunakan tag `<ion-content>` sejak Ionic Framework versi 4.

- *List*

*List* dengan tag `<ion-list>`, yang terdiri dari baris yang setiap barisnya berisi nama dan lokasi *venues* yang disusun menggunakan `<ion-item>`. UI Component *List* dengan tag `<ion-list>` dan `<ion-item>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

Sebagai penyedia *interface* untuk mengakses SDK *native* dan API *native* pada perangkat, pada skripsi ini akan menggunakan Capacitor dibandingkan dengan Cordova. Capacitor mengelola *plugin* dengan cara yang berbeda dibandingkan dengan Cordova, yaitu dengan cara membangun semua *plugin* sebagai sebuah *libraries* di Android, dan diinstal menggunakan *management tool* android, yaitu Gradle. Selain itu, Capacitor didukung langsung oleh Ionic, dengan pengembangan yang lebih baru dibandingkan Cordova dapat lebih mendukung untuk *Web Apps* modern untuk membuka fungsionalitas *native* dari platform melalui API.

Dengan digunakannya Capacitor, maka akan digunakan sebuah *plugin* yang disediakan oleh komunitas Capacitor, yaitu Capacitor Google Maps yang menggantikan *plugin* Google Maps sebelumnya yang berjalan menggunakan Cordova. *Plugin* ini berfungsi untuk menampilkan peta Google Maps secara *native* pada perangkat pengguna. Peta tersebut berisi lokasi dari seluruh acara WSDC 2017 Bali dilaksanakan yang ditandai dengan *marker*, dan juga akan menampilkan posisi dari perangkat pengguna. Untuk mengetahui posisi dari pengguna, akan digunakan sebuah *plugin* yang disediakan oleh Capacitor, yaitu Capacitor Geolocation. *Plugin*

1 Geolocation akan mendapatkan koordinat berupa *latitude* dan *longitude* dimana perangkat  
2 berada saat ini dengan menggunakan *Global Positioning System* (GPS). Selain itu, akan  
3 digunakan juga sebuah *plugin* untuk menampilkan *splash screen*, yaitu Capacitor *Splash Screen*,  
4 dimana pada aplikasi sebelumnya menggunakan Cordova. *Splash screen* akan ditampilkan saat  
5 pengguna membuka aplikasi. *Splash screen* menampilkan logo WSDC, logo WSDC 2017 Bali,  
6 dan logo Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi Republik Indonesia.  
7 Digunakan juga Browser API untuk menjalankan kemampuan in-app-browser pada aplikasi  
8 WSDC 2017 Bali, yaitu untuk membuka berita terkait acara WSDC 2017 Bali. Browser  
9 API yang disediakan oleh Capacitor, digunakan untuk menggantikan In App Browser milik  
10 Cordova.

### 11 **3.2.2 Tantangan Pengembangan Sistem Usulan**

12 Saat sedang melakukan proses migrasi aplikasi WSDC 2017 Bali dari Ionic Framework versi 3 ke  
13 Ionic Framework versi 6, terdapat beberapa kendala yang dialami. Kendala-kendala tersebut adalah  
14 sebagai berikut :

- 15 • Seperti yang disebutkan pada landasan teori (Sub Bab 2.3.3) sebelum melakukan migrasi dari  
16 Ionic Framework versi 3 ke Ionic Framework versi 6 terlebih dahulu melakukan migrasi dari  
17 Ionic Framework versi 3 ke Ionic Framework versi 4, yang selanjutnya dari Ionic versi 4 ke  
18 Ionic versi 5 dan dari Ionic versi 5 ke Ionic versi 6. Namun karena tidak tersedianya perintah  
19 untuk membuat aplikasi dengan menggunakan Ionic Framework versi 4 dan 5, maka penulis  
20 langsung melakukan migrasi dari Ionic Framework versi 3 ke Ionic Framework versi 6. Dalam  
21 melakukan hal ini, penulis berlandaskan bahwa susunan kelas Ionic Framework versi 4, 5 dan  
22 6 tidaklah berubah sama sekali. Yang mengalami perubahan hanyalah pembaruan properti  
23 mengenai API, CSS, dan *package dependencies* yang terpasang, yang telah dijelaskan pada  
24 landasan teori (Sub Bab 2.3.3).
- 25 • Pada awal pengerjaan skripsi, halaman Draw dan Result pada aplikasi WSDC 2017 Bali  
26 tidak dapat diakses karena terjadi kesalahan konfigurasi pada server. Kemudian setelah  
27 menghubungi dan dibantu oleh pembuat dari aplikasi WSDC 2017 Bali, maka masalah ini  
28 telah terselesaikan, yaitu halaman Draw dan Result pada aplikasi WSDC 2017 Bali dapat  
29 diakses kembali sebagaimana mestinya.
- 30 • Pada saat pengerjaan skripsi ini, pada Desember 2021 Ionic meluncurkan generasi terbaru  
31 dari Ionic Framework, yaitu Ionic versi 6. Sedangkan Ionic versi 5 pengembangannya berhenti  
32 didukung pada tanggal 8 Juni 2022 <sup>1</sup>. Maka dari itu, atas saran dan masukan dosen  
33 pembimbing dan dosen penguji, maka peneliti melakukan migrasi tidak lagi sampai Ionic versi  
34 5, melainkan sampai dengan Ionic versi 6 untuk masa dukungan Ionic Framework yang lebih  
35 lama.

---

<sup>1</sup>Status pemeliharaan dan dukungan dari setiap versi Ionic dilihat pada <https://ionicframework.com/docs/reference/support>



1

## BAB 4

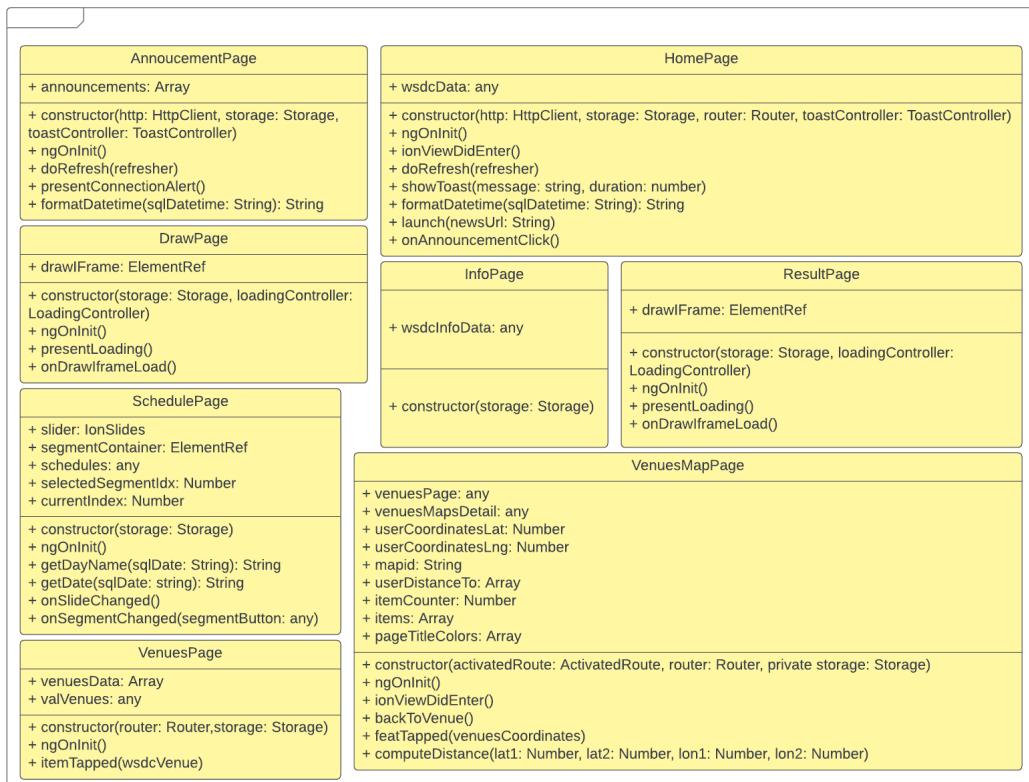
2

# PERANCANGAN

- 3 Pada bab ini akan dijelaskan mengenai perancangan aplikasi yang akan dibangun meliputi perancangan  
4 kelas pada masing-masing komponen beserta dengan deskripsi dan fungsinya, dan perancangan  
5 struktur HTML. Untuk antarmuka pada aplikasi yang akan dibangun akan memiliki antarmuka  
6 yang sama dengan aplikasi WSDC 2017 Bali terdahulu. Penjelasan terkait dengan antarmuka telah  
7 dibahas pada bagian [3.1.2](#).

## 8 4.1 Perancangan Kelas

- 9 Pada aplikasi WSDC 2017 Bali yang akan dibangun menggunakan struktur kelas yang sama dengan  
10 aplikasi WSDC 2017 Bali terdahulu, dengan beberapa penyesuaian terkait dengan pembaruan yang  
11 dilakukan. Diagram kelas secara keseluruhan dapat dilihat pada gambar [4.1](#).



Gambar 4.1: Diagram Kelas Keseluruhan

1 Perancangan kelas dari masing-masing komponen adalah sebagai berikut:

2 1. Komponen *Announcements*

3 Di dalam komponen *announcements* terdapat sebuah kelas *AnnouncementPage*. Kelas ini  
4 berfungsi untuk mengambil data *announcements* dari *storage*, dan menampilkan pengumuman  
5 ke halaman *announcements*. Kelas ini memiliki sebuah atribut, yaitu *announcements* bertipe  
6 *array* yang menyimpan localtime bertipe data *string* yang berisi tanggal dan waktu dari  
7 sebuah pengumuman dikeluarkan dengan format “Tahun-Bulan-Hari Jam:Menit:Detik”, dan  
8 message yang berisi pesan pengumuman yang bertipe data *string*. Kelas ini memiliki beberapa  
9 *method*, diantaranya adalah sebagai berikut:

- 10 • **constructor(private http: HttpClient, private storage: Storage, public toas-**
- 11 **tController: ToastController)**

12 *Method* constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat.

13 **Parameter:**

- 14 – **http**: Parameter ini digunakan untuk menyimpan API HttpClient.
- 15 – **storage**: Parameter ini digunakan untuk menyimpan API Storage.
- 16 – **toastController**: Parameter ini digunakan untuk menyimpan API ToastController.

17 **Kembalian:** tidak ada.

- 18 • **ngOnInit()**

19 *Method* ini berfungsi untuk mengambil data *announcements* yang terdapat di dalam  
20 *storage*, kemudian menyimpannya ke dalam atribut kelas, yaitu *announcements*.

21 **Parameter:** tidak ada.

22 **Kembalian:** tidak ada.

- 23 • **doRefresh(refresher)**

24 *Method* ini berfungsi untuk melakukan penyegaran ulang terhadap halaman *Announ-*  
25 *cements*. Saat melakukan penyegaran ulang, *method* ini akan mengambil data terbaru  
26 dari server, kemudian menyimpannya ke dalam atribut kelas, yaitu *announcements*.  
27 Selanjutnya akan dilakukan penghapusan terlebih dahulu terhadap data yang sudah ada  
28 di dalam *storage*, kemudian menyimpan data terbaru yang di dapatkan dari server ke  
29 dalam *storage*. Jika server tidak merespon dan data tidak dapat diambil, maka akan  
30 memanggil *method* presentConnectionAlert() untuk menampilkan *toast*.

31 **Parameter:** *Method* ini memiliki sebuah parameter yaitu *refresher*, yang berisi *event*  
32 *refresher*.

33 **Kembalian:** tidak ada.

- 34 • **presentConnectionAlert()**

35 *Method* ini berfungsi untuk menampilkan *toast* yang akan menampilkan sebuah tulisan  
36 “Failed to refresh information” selama tiga detik. *Method* ini hanya akan digunakan  
37 ketika *method* doRefresh tidak berhasil mengambil data terbaru dari server.

38 **Parameter:** tidak ada.

39 **Kembalian:** tidak ada.

1     • **formatDatetime(sqlDatetime: string)**

2       Method ini berfungsi untuk mengambil jam, menit, hari, dan bulan dari parameter.

3       **Parameter:** sqlDatetime: detail waktu dan tanggal pada sebuah pengumuman.

4       **Kembalian:** sebuah *string* yang berisi tanggal dan waktu dari sebuah pengumuman  
5       dengan format “Jam-Menit | Hari, Tanggal, Bulan”.

6     2. Komponen *Draw*

7       Di dalam komponen *announcements* terdapat sebuah kelas DrawPage. Kelas ini berfungsi  
8       untuk mengambil data *draw* dari *storage*, serta menampilkan data *draw* ke halaman *draw*.  
9       Kelas ini memiliki sebuah atribut, yaitu drawIFrame yang bertipe ElementRef. Atribut ini  
10      merupakan sebuah ViewChild yang digunakan untuk memanggil elemen dari komponen *draw*,  
11      yaitu drawIFrame pada file draw.page.html.

12      Kelas ini memiliki beberapa *method*, diantaranya adalah sebagai berikut:

13     • **constructor(private storage: Storage, public loadingController: LoadingCon-**

14       **troller)**

15       Method constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat.

16       **Parameter:**

17           – **storage:** Parameter ini digunakan untuk menyimpan API Storage.

18           – **loadingController:** Parameter ini digunakan untuk menyimpan API LoadingCon-

19           troller.

20       **Kembalian:** tidak ada.

21     • **ngOnInit()**

22       Method ini berfungsi untuk mengambil data *draws* yang terdapat di dalam *storage*. Data  
23       tersebut lalu disimpan ke dalam *child* drawIFrame. Kemudian *method* ini akan memanggil  
24       *method* presentLoading().

25       **Parameter:** tidak ada.

26       **Kembalian:** tidak ada.

27     • **async presentLoading()**

28       Method ini berfungsi untuk menampilkan sebuah indikator *loading* dengan pesan “Please  
29       wait...”.

30       **Parameter:** tidak ada.

31       **Kembalian:** tidak ada.

32     • **onDrawIframeLoad()**

33       Method ini merupakan sebuah *template statement* yang dipanggil oleh *event* di dalam  
34       tag <iframe> pada file draw.page.html. Method ini berfungsi untuk menampilkan data  
35       *draw* yang disimpan di dalam *storage*.

36       **Parameter:** tidak ada.

37       **Kembalian:** tidak ada.

38     3. Komponen *Home*

39       Di dalam komponen *home* terdapat sebuah kelas HomePage. Kelas ini menjadi sebagai  
40       kelas yang pertama kali diakses oleh aplikasi. Maka dari itu, kelas ini berfungsi untuk  
41       menginisialisasi sebuah *storage* yang diisi oleh data yang digunakan oleh seluruh kelas pada  
42       aplikasi. Kelas ini memiliki sebuah atribut, yaitu wsdcData yang bertipe any. Atribut ini

akan menyimpan data json berisi data yang akan digunakan untuk aplikasi.

Kelas ini memiliki beberapa *method*, diantaranya adalah sebagai berikut:

- **constructor(private http: HttpClient, private storage: Storage, private router: Router, public toastController: ToastController)**

*Method* constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat.

**Parameter:**

- **http:** Parameter ini digunakan untuk menyimpan API HttpClient.
- **storage:** Parameter ini digunakan untuk menyimpan API Storage.
- **router:** Parameter ini digunakan untuk menyimpan API Router.
- **toastController:** Parameter ini digunakan untuk menyimpan API ToastController.

**Kembalian:** tidak ada.

- **ionViewDidEnter()**

*Method* ini dijalankan ketika halaman sudah masuk sepenuhnya. *Method* ini berfungsi untuk menutup *splash screen*.

**Parameter:** tidak ada.

**Kembalian:** tidak ada.

- **ngOnInit()**

*Method* ini berfungsi untuk mengambil data json dari *storage*. Jika data di dalam *storage* tersebut belum pernah dibuat sebelumnya, maka *method* ini akan membuat sebuah data baru di dalam *storage* yang berisi data json yang dibutuhkan untuk aplikasi. Secara *default*, untuk mengantisipasi jika pengguna tidak memiliki koneksi internet, maka pertama kali *method* ini mengambil data adalah dari aset yang berada di *folder assets*. Setelah itu, *method* ini akan mengambil data terbaru dari server. Jika server tidak merespon dan data tidak berhasil didapatkan, maka akan memanggil *method showToast()*.

**Parameter:** tidak ada.

**Kembalian:** tidak ada.

- **doRefresh(refresher)**

*Method* ini berfungsi untuk melakukan penyegaran ulang terhadap halaman *Home*. Saat melakukan penyegaran ulang, *method* ini akan mengambil data terbaru dari server, kemudian menyimpannya ke dalam atribut kelas, yaitu *wsdcData*. Selanjutnya akan dilakukan penghapusan terlebih dahulu terhadap data yang sudah ada di dalam *storage*, kemudian menyimpan data terbaru yang di dapatkan dari server ke dalam *storage*. Jika server tidak memberi respon dan data tidak dapat diambil, maka akan memanggil *method showToast()* untuk menampilkan *toast*.

**Parameter:** tidak ada.

**Kembalian:** tidak ada.

- **async showToast(message: string, duration: number=3000)**

*Method* ini berfungsi untuk menampilkan *toast* yang akan menampilkan pesan dan durasi sesuai dengan yang ada pada parameter.

**Parameter:**

- **message**: pesan yang akan ditampilkan di dalam *toast*.
- **duration**: durasi seberapa lama *toast* berada di layar.

**Kembalian:** tidak ada.

- **formatDatetime(sqlDatetime: string)**

*Method* ini berfungsi untuk mengambil jam, menit, hari, dan bulan dari parameter.

**Parameter:** sqlDatetime: detail waktu dan tanggal pada sebuah pengumuman.

**Kembalian:** sebuah *string* yang berisi tanggal dan waktu dari sebuah pengumuman dengan format “Hari | Jam-Menit”.

- **launch(newsUrl: string)**

*Method* ini berfungsi untuk membuka url berita sesuai dengan url yang ada di dalam parameter.

**Parameter:** newsUrl: *string* url dari sebuah berita.

**Kembalian:** tidak ada.

- **onAnnouncementClick()**

*Method* ini berfungsi untuk berpindah halaman ke halaman *announcements*.

**Parameter:** tidak ada.

**Kembalian:** tidak ada.

#### 4. Komponen Info

Di dalam komponen info terdapat sebuah kelas InfoPage. Kelas ini berfungsi untuk mengambil data info dari *storage* dan menyimpannya ke atribut kelas yang kemudian data tersebut akan ditampilkan ke halaman info. Kelas ini memiliki sebuah atribut, yaitu wsdcInfoData yang bertipe any. Atribut ini digunakan untuk menyimpan data untuk halaman info. Kelas ini hanya memiliki sebuah *method* yaitu *method constructor*. *Method* ini memiliki sebuah parameter, yaitu storage yang bertipe Storage. Constructor pada kelas ini bertujuan untuk mengambil data info dari *storage* dan menyimpannya ke dalam atribut kelas, yaitu wsdcInfoData.

#### 5. Komponen Result

Di dalam komponen *result* terdapat sebuah kelas ResultPage. Kelas ini berfungsi untuk mengambil data *result* dari *storage* yang kemudian data tersebut akan ditampilkan ke halaman *result*. Kelas ini memiliki sebuah atribut yaitu resultIFrame yang bertipe ElementRef. Atribut ini merupakan sebuah @ViewChild yang digunakan untuk memanggil elemen dari komponen *result*, yaitu resultIFrame pada file result.page.html.

Kelas ini memiliki beberapa *method*, diantaranya adalah sebagai berikut:

- **constructor(private storage: Storage, public loadingController: LoadingController)**

*Method* constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat.

**Parameter:**

- **storage**: Parameter ini digunakan untuk menyimpan API Storage.
- **loadingController**: Parameter ini digunakan untuk menyimpan API LoadingController.

**Kembalian:** tidak ada.

1     • **ngOnInit()**

2       *Method* ini berfungsi untuk mengambil data *result* yang terdapat di dalam *storage*.  
3       Data tersebut lalu disimpan ke dalam *child resultIFrame*. Setelah itu *method* ini akan  
4       memanggil *method* presentLoading().

5       **Parameter:** tidak ada.

6       **Kembalian:** tidak ada.

7     • **async presentLoading()**

8       *Method* ini berfungsi untuk menampilkan sebuah indikator *loading* dengan pesan “Please  
9       wait...”.

10      **Parameter:** tidak ada.

11      **Kembalian:** tidak ada.

12     • **onResultIframeLoad()**

13       *Method* ini merupakan sebuah *template statement* yang dipanggil oleh *event* di dalam  
14       tag <iframe> pada *file result.page.html*. *Method* ini berfungsi untuk menampilkan data  
15       *result* yang disimpan di dalam *storage*.

16       **Parameter:** tidak ada.

17       **Kembalian:** tidak ada.

18     6. Komponen *Schedule*

19       Di dalam komponen *schedule* terdapat sebuah kelas SchedulePage. Kelas ini berfungsi untuk  
20       mengatur jadwal pada halaman *schedule*, seperti mengatur perpindahan *slides* dan *segment*  
21       pada jadwal.

22       Kelas ini memiliki beberapa atribut, diantaranya adalah sebagai berikut:

- 23       • **slider:** Atribut ini merupakan sebuah @ViewChild yang digunakan untuk memanggil  
24       elemen dari komponen *schedule*, yaitu scheduleSlider pada *file schedule.page.html*.
- 25       • **segmentContainer:** Atribut ini merupakan sebuah @ViewChild yang digunakan untuk  
26       memanggil elemen dari komponen *schedule*, yaitu segmentContainer pada *file schedule.page.html*.
- 27       • **schedules:** Atribut ini akan menyimpan data *schedules* yang diambil dari *storage*.
- 28       • **slideOpts:** Atribut ini berisi pengaturan dasar untuk *slides*. Berisi initialSlide untuk  
29       mengatur *slides* ke berapa saat pertama kali halaman *schedule* dibuka, dan speed untuk  
30       mengatur kecepatan transisi antar *slides*.
- 31       • **selectedSegmentIdx:** Atribut ini digunakan untuk menyimpan index dari *segment*  
32       yang sedang aktif.
- 33       • **currentIndex:** Atribut ini digunakan untuk menyimpan index dari *slides* yang sedang  
34       aktif.

35       Kelas ini memiliki beberapa *method*, diantaranya adalah sebagai berikut:

36     • **constructor(private storage: Storage)**

37       *Method* constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat.

38       **Parameter:** storage: Parameter ini digunakan untuk menyimpan API Storage.

39       **Kembalian:** tidak ada.

- 1     • **ngOnInit()**  
2        *Method* ini berfungsi untuk mengambil data *schedule* yang terdapat di dalam *storage*.  
3        Data tersebut lalu disimpan ke dalam atribut *schedules*.  
4        **Parameter:** tidak ada.  
5        **Kembalian:** tidak ada.
- 6     • **getDayName(sqlDate: string)**  
7        *Method* ini berfungsi untuk mengambil hari dari parameter.  
8        **Parameter:** sqlDate: Sebuah *string* yang berisi tahun, bulan, dan tanggal.  
9        **Kembalian:** *string* nama hari.
- 10    • **getDate(sqlDate: string)**  
11      *Method* ini berfungsi untuk mengambil tanggal dari parameter.  
12      **Parameter:** sqlDate: Sebuah *string* yang berisi tahun, bulan, dan tanggal.  
13      **Kembalian:** *string* tanggal.
- 14    • **onSlideChanged()**  
15      *Method* ini dipanggil saat *slides* dipindahkan dengan cara digeser ke kanan atau ke kiri.  
16      *Method* ini akan mengubah atribut *currentIndex* menjadi index *slides* saat ini, kemudian  
17      mengubah atribut *selectedSegmentIdx* menjadi index *slides* saat ini. Hal ini bertujuan  
18      agar indeks dari *segment* yang aktif dapat diganti sesuai dengan indeks *slides* yang aktif.  
19      Dengan begitu tampilan *segment* dan *slides* yang aktif akan sesuai.  
20      **Parameter:** tidak ada.  
21      **Kembalian:** tidak ada.
- 22    • **onSegmentChanged(segmentButton)**  
23      *Method* ini berfungsi untuk mengubah *slides* yang aktif sesuai dengan indeks dari *segment*  
24      yang sedang aktif.  
25      **Parameter:** segmentButton: Merupakan sebuah *event* dari *segment* yang akan diambil  
26      *value* yang berisi indeks dari *segment* yang aktif.  
27      **Kembalian:** tidak ada.

## 28   7. Komponen *Venues*

29   Di dalam komponen *venues* terdapat sebuah kelas *VenuesPage*. Kelas ini berfungsi untuk  
30   mengambil data *venues* dari *storage* dan menyimpannya ke dalam atribut kelas. Selain itu  
31   kelas ini juga berfungsi melakukan navigasi ke halaman *venues map*.

32   Kelas ini memiliki beberapa atribut, yaitu:

- 33     • **venuesData:** Atribut ini bertipe array. Atribut ini digunakan untuk menyimpan data  
34        dari *venues* yang berisi id, name, icon, geojson, dan colorIdx dari masing masing kategori  
35        *venues*. Data ini nantinya akan dikirimkan ke kelas *Venues Maps*.
- 36     • **valVenues:** Digunakan untuk menyimpan data *venues* yang didapatkan dari *storage*.

37   Kelas ini memiliki beberapa *method*, diantaranya yaitu:

- 38     • **constructor(private router: Router,private storage: Storage)**

39      *Method* constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat.

1           **Parameter:**

- 2           – **router:** Parameter ini berfungsi untuk menyimpan API Router.  
3           – **storage:** Parameter ini berfungsi untuk menyimpan API Storage.

4           **Kembalian:** tidak ada.

5           • **ngOnInit()**

6           *Method* ini berfungsi untuk mengambil data *venues* yang terdapat di dalam *storage*. Data  
7           tersebut lalu disimpan ke dalam atribut *valVenues*.

8           **Parameter:** tidak ada.

9           **Kembalian:** tidak ada.

10          • **itemTapped(wsdcVenue)**

11          *Method* ini berfungsi untuk melakuakn navigasi ke halaman *Venues Maps* dengan bantuan  
12          Router milik Angular. *Method* ini akan mengirimkan sebuah data array yang didapatkan  
13          dari parameter ke halaman *Venues Maps*.

14          **Parameter:** wsdcVenue: Parameter ini merupakan sebuah array yang berisi sama seperti  
15          atribut *venuesData*. Isi dari array ini adalah data untuk sebuah *venues* yang ingin dilihat.

16          **Kembalian:** tidak ada.

17          8. Komponen *Venues Maps*

18          Di dalam komponen *venues Maps* terdapat sebuah kelas *VenuesMapsPage*. Kelas ini berfungsi  
19          untuk mengambil data *venues* yang dikirimkan dari komponen *Venues*, menampilkan peta  
20          *venues*, serta melakukan kalkulasi jarak pengguna dengan *venues*. Kelas ini memiliki beberapa  
21          atribut, yaitu:

- 22           • **venuesPage** : Atribut ini digunakan untuk menyimpan data yang dikirimkan dari  
23           komponen *Venues*.
- 24           • **venuesMapsDetail** : Atribut ini digunakan untuk menyimpan data *venues* dari *storage*  
25           sesuai dengan kategori yang sedang dipilih.
- 26           • **userCoordinatesLat** : Atirbut ini digunakan untuk menyimpan koordinat latitude dari  
27           perangkat pengguna.
- 28           • **userCoordinatesLng** : Atirbut ini digunakan untuk menyimpan koordinat longitude  
29           dari perangkat pengguna.
- 30           • **mapid** : Atribut ini digunakan untuk menyimpan id dari peta.
- 31           • **userDistanceTo** : Atribut ini digunakan untuk menyimpan jarak dari posisi perangkat  
32           pengguna ke posisi *venues*.
- 33           • **itemCounter** : Atribut ini digunakan sebagai penghitung berapa banyak *venues* dalam  
34           sebuah kategori.
- 35           • **items** : Atribut ini digunakan untuk menyimpan id dan id warna dari sebuah katego-  
36           ri *venues*.
- 37           • **pageTitleColors** : Atribut ini merupakan sebuah *array* yang berisi kode warna un-  
38           tuk masing-masing judul kategori *venues*.

39          Kelas ini memiliki beberapa *method*, diantaranya adalah sebagai berikut:

- 40           • **constructor(private activatedRoute: ActivatedRoute, private router: Router,  
41           private storage: Storage)**

1        *Method constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat. Selain  
2        itu, constructor juga digunakan untuk mengambil data dari storage dan memasukannya  
3        ke atribut venuesMapsDetail.*

4        **Parameter:**

- 5        – **activatedRoute** : Parameter ini digunakan untuk menyimpan API ActivatedRoute.
- 6        – **router** : Parameter ini digunakan untuk menyimpan API Router.
- 7        – **storage** : Parameter ini digunakan untuk menyimpan API Storage.

8        **Kembalian:** tidak ada.

9        • **ngOnInit()**

10      *Method ini berfungsi untuk mengambil data venues yang terdapat di dalam storage. Data  
11      tersebut lalu disimpan ke dalam atribut venuesMapsDetail.*

12      **Parameter:** tidak ada.

13      **Kembalian:** tidak ada.

14      • **ionViewDidEnter()**

15      *Method ini dijalankan ketika halaman sudah masuk sepenuhnya. Method ini digunakan  
16      untuk menginisialisasi peta menggunakan plugin Google Maps yang disediakan oleh  
17      Capacitor. Peta tersebut menampilkan peta Pulau Bali, lebih tepatnya di Kecamatan  
18      Kuta dengan latitude -8.722396 dan longitude 115.17671. Method ini juga membuat  
19      marker yang menandai setiap lokasi venues pada satu kategori venues. Selain itu method  
20      ini juga digunakan untuk menyimpan jarak antara posisi perangkat pengguna ke masing-  
21      masing posisi venues, yang kemudian disimpan ke dalam atibut userDistanceTo.*

22      **Parameter:** tidak ada.

23      **Kembalian:** tidak ada.

24      • **backToVenue()**

25      *Method ini digunakan untuk bernavigasi kembali ke halaman Venues.*

26      **Parameter:** tidak ada.

27      **Kembalian:** tidak ada.

28      • **featTapped(venuesCoordinates)**

29      *Method ini digunakan untuk mengarahkan kamera map ke arah lokasi venues yang dituju  
30      sesuai dengan koordinat pada parameter.*

31      **Parameter:** venuesCoordinates: Parameter ini berisi koordinat latitude dan longitude  
32      dari lokasi venues yang dituju.

33      **Kembalian:** tidak ada.

34      • **computeDistance(lat1, lat2, lon1, lon2)**

35      *Method ini berfungsi untuk menghitung jarak dari posisi perangkat pengguna ke posisi  
36      venues menggunakan latitude dan longitude dari kedua posisi tersebut.*

37      **Parameter:**

- 38        – **lat1**: koordinat latitude dari pengguna.
- 39        – **lat2**: koordinat latitude dari *venues*.
- 40        – **lon1**: koordinat longitude dari pengguna.
- 41        – **lon2**: koordinat longitude dari *venues*.

42      **Kembalian:** *String* jarak dari posisi perangkat pengguna ke posisi *venues*.

## 4.2 Perancangan Struktur HTML

Struktur HTML pada masing-masing komponen mengambil struktur yang sama dengan aplikasi WSDC 2017 Bali terdahulu, namun dengan beberapa perubahan. Perubahan-perubahan tersebut telah dibahas pada bagian 2.3.3, serta analisis penggunaannya di sistem usulan pada bagian 3.2.1.

Masing-masing HTML yang terdapat pada setiap komponen memiliki struktur yang sama, yaitu terdapat sebuah *header* dan sebuah *content*. Penjelasan struktur pada *header* dan *content* adalah sebagai berikut:

### 1. Header

*Header* untuk setiap komponen pada umumnya memiliki struktur yang serupa namun dibedakan dengan judul dari setiap halaman. *Header* digunakan untuk menampilkan judul dari sebuah halaman, serta menyediakan sebuah *menu button* sebagai salah satu cara untuk membuat *sidemenu* untuk melakukan navigasi antar halaman. *Header* dibungkus oleh tag `<ion-header>` yang didalamnya terdapat tag `<ion-toolbar>` yang disediakan Ionic Framework. Kemudian untuk *menu button* dibuat oleh tag `<ion-menu-button>` pada tag `<ion-buttons>`. Selanjutnya untuk judul dari sebuah halaman dibungkus oleh tag `<ion-title>`. Judul akan berbeda beda tergantung dengan halamannya. Salah satu contoh dari penggunaan *header* adalah pada gambar 3.2c yang ditandai dengan kotak berwarna biru.

### 2. Content

*Content* untuk setiap komponen memiliki struktur yang berbeda-beda tergantung dengan isi dari halaman tersebut. Namun *content* untuk setiap komponen dibungkus oleh sebuah tag `<ion-content>`. Salah satu contoh dari penggunaan *content* adalah pada gambar 3.2c yang ditandai dengan warna merah. Struktur *content* untuk masing-masing halaman adalah sebagai berikut:

- Halaman *Announcements*

*Content* pada halaman *announcements* berisi sebuah *refresher* dengan tag `<ion-refresher>` untuk melakukan penyegaran ulang terhadap halaman *announcements* yaitu mengambil data terbaru dari server. Penggunaan tag `<ion-refresher>` seperti pada gambar 3.2a yang ditandai dengan kotak berwarna hijau. Selain itu terdapat sebuah list dengan tag `<ion-list>` yang ditandai dengan kotak berwarna kuning. List menampilkan pengumuman yang berisi waktu dan tanggal, serta pesan dari pengumuman tersebut. Setiap satu pengumuman dibungkus oleh sebuah tag `<ion-item>` yang ditandai dengan kotak berwarna hitam. Di dalamnya terdapat sebuah tag `<ion-label>` yang berisi tag `<h3>` untuk waktu dan tanggal, serta tag `<p>` untuk pesan pengumuman.

- Halaman *Draw*

*Content* pada halaman *draw* berisi sebuah tag `<iframe>` yang digunakan untuk menempatkan dokumen lain ke dalam dokumen HTML, yaitu sebuah data *draw* yang berisi pembagian grup proposisi dan oposisi bagi setiap negara peserta WSDC 2017 Bali. Data tersebut diambil dari server, kemudian dimasukan ke dalam tag `<iframe>`. Penggunaan tag `<iframe>` seperti pada gambar 3.2b yang ditandai dengan kotak berwarna hijau.

- Halaman *Home*

*Content* pada halaman *home* berisi sebuah *refresher* dengan tag `<ion-refresher>` untuk melakukan penyegaran ulang terhadap halaman *home*, yaitu mengambil data terbaru

dari server. Penggunaan *tag* <ion-refresher> seperti pada gambar 3.2c yang ditandai dengan kotak berwarna hijau. Selain itu terdapat sebuah *card* dengan *tag* <ion-card> seperti yang ditandai dengan kotak berwarna merah muda yang digunakan untuk menampilkan sebuah pengumuman terbaru, berikut dengan waktu dan tanggal serta pesan dari pengumuman tersebut. Di dalam *card* terdapat *grid* untuk *layout* dari *card*. Di dalam sebuah *grid* terdapat sebuah baris dengan *tag* <ion-row>. Di dalam baris tersebut terdapat dua buah kolom dengan *tag* <ion-col>. Masing-masing kolom memiliki ukuran, kolom pertama yang ditandai dengan kotak berwarna coklat berukuran sembilan digunakan untuk menyimpan *header* dari *card* dengan *title* yaitu “Latest Announcement”. *Header* ini dibungkus di dalam *tag* <ion-card-header>, dan *title* dengan *tag* <ion-card-title>. Selain *header* untuk *card*, terdapat pula *content* untuk *card* dengan *tag* <ion-card-content> yang berisi waktu dan tanggal, serta pesan dari pengumuman. Untuk kolom selanjutnya dengan ukuran tiga berisi sebuah gambar seperti yang ditandai dengan kotak berwarna jingga.

Selain *card* pengumuman, terdapat juga list dengan *tag* <ion-list> yang berisi *thumbnail* dari berita-berita terkait acara WSDC 2017 Bali seperti yang ditandai dengan kotak berwarna ungu pada gambar 3.2c. Di dalam list terdapat sebuah *header* dengan *tag* <ion-list-header> yang berisi judul dari list yaitu “Newsletters” yang berada di dalam *tag* <ion-label>. Untuk masing-masing *thumbnail* berita berada di dalam *tag* <ion-item>. Untuk masing-masing item terdapat sebuah gambar *thumbnail* dari sebuah berita, judul dari berita tersebut, serta sebuah tombol dengan *tag* <ion-button> yang jika ditekan akan mengarahkan pengguna untuk melihat berita tertentu sesuai dengan item yang dipilih.

• Halaman Info

*Content* pada halaman info berisi sebuah *grid* dengan *tag* <ion-grid>, yang berisi sebuah baris dengan *tag* <ion-row>. Baris ini menyimpan info-info seputar kontak-kontak penting yang dapat dihubungi, kosa kata dalam Bahasa Indonesia sehari-hari, serta credits kepada pembuat aplikasi WSDC 2017 Bali.

• Halaman Result

*Content* pada halaman result berisi sebuah *tag* <iframe> yang digunakan untuk menyematkan dokumen lain ke dalam dokumen HTML, yaitu sebuah data *result* hasil dari keseluruhan pertandingan WSDC 2017 Bali. Data tersebut diambil dari server, kemudian dimasukan ke dalam *tag* <iframe>.

• Halaman Schedule

*Content* pada halaman *schedule* berisi *segment* dengan *tag* <ion-segment> dan sebuah *slides* dengan *tag* <ion-slides>. *Segment* digunakan untuk menampilkan tanggal dan hari, serta berfungsi untuk memindahkan *slides* ke hari yang dipilih seperti yang ditandai dengan kotak berwarna hijau pada gambar 3.3c. Untuk melakukan hal tersebut, di dalam *segment* terdapat sebuah *button* dengan *tag* <ion-segment-button> yang berisi *tag* <ion-label> untuk menampung tanggal dan hari. Sedangkan *slides* digunakan untuk menampilkan jadwal acara WSDC 2017 Bali pada hari yang sesuai dengan *segment* yang terpilih seperti yang ditandai dengan kotak berwarna coklat. Untuk menampilkan jadwal

1 menggunakan *list* dengan *tag* `<ion-list>` yang ditandai dengan warna merah muda.  
2 Di dalam *list* terdapat sebuah *item* dengan *tag* `<ion-item>`. Untuk setiap *item* berisi  
3 waktu mulai dan selesai sebuah acara dengan *tag* `<ion-note>` yang ditandai dengan  
4 warna ungu, serta nama acara dengan *tag* `<h3>` yang ditandai dengan kotak berwarna  
5 jingga dan lokasi acara dengan *tag* `<p>` yang ditandai dengan kotak berwarna biru muda.  
6 Nama dan lokasi acara dibungkus dengan *tag* `<ion-label>`.

7 • Halaman *Venues*

8 *Content* pada halaman *venues* berisi *grid* dengan *tag* `<ion-grid>` dengan satu baris  
9 menggunakan *tag* `<ion-row>`. Di dalamnya terdapat sebuah *list* dengan *tag* `<ion-list>`.  
10 *List* tersebut berisi tombol dengan *tag* `<ion-button>` yang ditandai dengan kotak ber-  
11 warna hijau pada gambar 3.3a. Masing-masing tombol digunakan untuk melakukan  
12 navigasi ke halaman *venues map*. Setiap tombol berisi ikon dengan *tag* `<ion-icon>` yang  
13 ditandai dengan kotak berwarna biru muda pada gambar 3.3a dan sebuah *tag* `<span>`  
14 yang berisi nama dari *venues*, ditandai dengan kotak berwarna hitam.

15 • Halaman *Venues Map*

16 *Content* pada halaman *venues map* berisi *tag* `<div>` yang digunakan untuk menampung  
17 peta dari sebuah kategori *venues* seperti yang ditandai dengan kotak berwarna hijau  
18 pada gambar 3.3b. Selain itu terdapat sebuah label dengan *tag* `<ion-label>` yang berisi  
19 nama kategori *venues* yang ditandai dengan kotak berwarna kuning. Kemudian untuk  
20 menampilkan nama dan deskripsi sebuah *venues*, serta jarak antara pengguna dengan  
21 *venues*, menggunakan *list* dengan *tag* `<ion-list>` yang ditandai dengan kotak berwarna  
22 biru muda.

1

## BAB 5

2

### IMPLEMENTASI DAN PENGUJIAN

3 Pada bab ini akan dijelaskan mengenai implementasi perangkat lunak, dan pengujian perangkat  
4 lunak. Implementasi perangkat lunak berisi penjelasan lingkungan pengembangan perangkat lunak  
5 dan hasil implementasi. Sedangkan pengujian perangkat lunak berisi hasil pengujian fungsional  
6 dan eksperimental terhadap perangkat lunak yang telah dibangun.

7 **5.1 Implementasi**

8 **5.1.1 Lingkugan Implementasi**

9 Implementasi perangkat lunak ini dilakukan di komputer penulis dengan spesifikasi berikut:  
10 1. *Processor*: Intel Core i5 7200U  
11 2. *Random Access Memory (RAM)*: 16GB DDR4  
12 3. Sistem Operasi: Windows 10 version 21H2  
13 4. Versi Android Development Kit (SDK): API 30 (Android 11 (R))

14 **5.1.2 Hasil Implementasi**

15 Hasil implementasi berupa sebuah aplikasi android WSDC 2017 Bali. Sebelum halaman dimuat,  
16 ditampilkan sebuah *splash screen* terlebih dahulu yang menampilkan logo WSDC, logo WSDC 2017  
17 Bali, dan logo Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi Republik Indonesia.  
18 Tangkapan layar *splash screen* dapat dilihat pada Gambar 5.1a. Aplikasi WSDC 2017 Bali terdiri  
19 dari 8 halaman yang dapat diakses melalui *sidemenu*. Tangkapan layar *sidemenu* dapat dilihat  
20 pada Gambar 5.2a. Halaman-halaman yang ada pada aplikasi WSDC 2017 Bali tersebut yaitu:

21 1. Halaman *Home*

22 Halaman *home* menjadi halaman pertama yang dimasuki oleh pengguna di aplikasi WSDC  
23 2017 Bali. Pada halaman ini pengguna dapat melihat pengumuman terbaru terkait dengan  
24 acara WSDC 2017 Bali, yang berisi hari, jam, dan pesan dari pengumuman tersebut, yang  
25 dapat diklik dan mengarahkan pengguna ke halaman *announcements*. Selain itu, pengguna  
26 dapat melihat *headline* berita-berita terkait dengan acara WSDC 2017 Bali. Untuk melihat  
27 berita tersebut secara penuh, disediakan sebuah tombol yang akan mengarahkan pengguna  
28 untuk melihat dan mengunduh berita terkait acara WSDC 2017 Bali. Tangkapan layar  
29 halaman *home* dapat dilihat pada Gambar 5.3a. Sebagai perbandingan, tangkapan layar  
30 halaman *home* pada aplikasi WSDC 2017 Bali terdahulu dapat dilihat pada Gambar 5.3b.

1    2. Halaman *Announcements*

2    Halaman *announcements* berisi pengumuman-pengumuman terkait dengan acara WSDC 2017  
3    Bali yang disajikan terurut menurun dengan waktu terbaru yang pertama. Tangkapan layar  
4    halaman *announcements* dapat dilihat pada Gambar 5.4a. Sebagai perbandingan, tangkapan  
5    layar halaman *announcements* pada aplikasi WSDC 2017 Bali terdahulu dapat dilihat pada  
6    Gambar 5.4b.

7    3. Halaman *Draw*

8    Halaman *Draw* menampilkan hasil dari pembagian grup oposisi dan proposisi dari negara-  
9    negara peserta WSDC 2017 Bali. Tangkapan layar halaman *draw* dapat dilihat pada Gam-  
10   bar 5.5a. Sebagai perbandingan, tangkapan layar halaman *Draw* pada aplikasi WSDC 2017  
11   Bali terdahulu dapat dilihat pada Gambar 5.5b.

12   4. Halaman *Info*

13   Halaman *info* menampilkan info-info seperti kontak-kontak penting yang dapat dihubungi,  
14   kosa kata dalam Bahasa Indonesia sehari-hari, serta *credits* kepada pembuat aplikasi WSDC  
15   2017 Bali. Tangkapan layar dari halaman *info* dapat dilihat pada Gambar 5.6a. Sebagai  
16   perbandingan, tangkapan layar halaman *info* pada aplikasi WSDC 2017 Bali terdahulu dapat  
17   dilihat pada Gambar 5.6b.

18   5. Halaman *Result*

19   Halaman *result* menampilkan hasil dari pertandingan WSDC 2017 Bali pada babak seperde-  
20   lapan final, seperempat final, dan semifinal. Tangkapan layar dari halaman *result* dapat dilihat  
21   pada Gambar 5.7a. Sebagai perbandingan, tangkapan layar halaman *result* pada aplikasi  
22   WSDC 2017 Bali terdahulu dapat dilihat pada Gambar 5.7b.

23   6. Halaman *Schedule*

24   Halaman *schedule* berisi jadwal acara WSDC 2017 Bali yang ditampilkan berkelompok  
25   berdasarkan tanggal dan hari. Jadwal yang ditampilkan berupa waktu mulai dan waktu  
26   selesai, lokasi acara, serta nama acara. Pengguna dapat berpindah ke hari manapun untuk  
27   melihat jadwal yang ada pada hari tersebut dengan menggulir menyamping pada bagian  
28   tanggal dan hari, serta bagian jadwal. Tangkapan layar halaman *schedule* dapat dilihat pada  
29   Gambar 5.8a. Lalu sebagai perbandingan, tangkapan layar halaman *schedule* pada aplikasi  
30   WSDC 2017 Bali terdahulu dapat dilihat pada Gambar 5.8b.

31   7. Halaman *Venues*

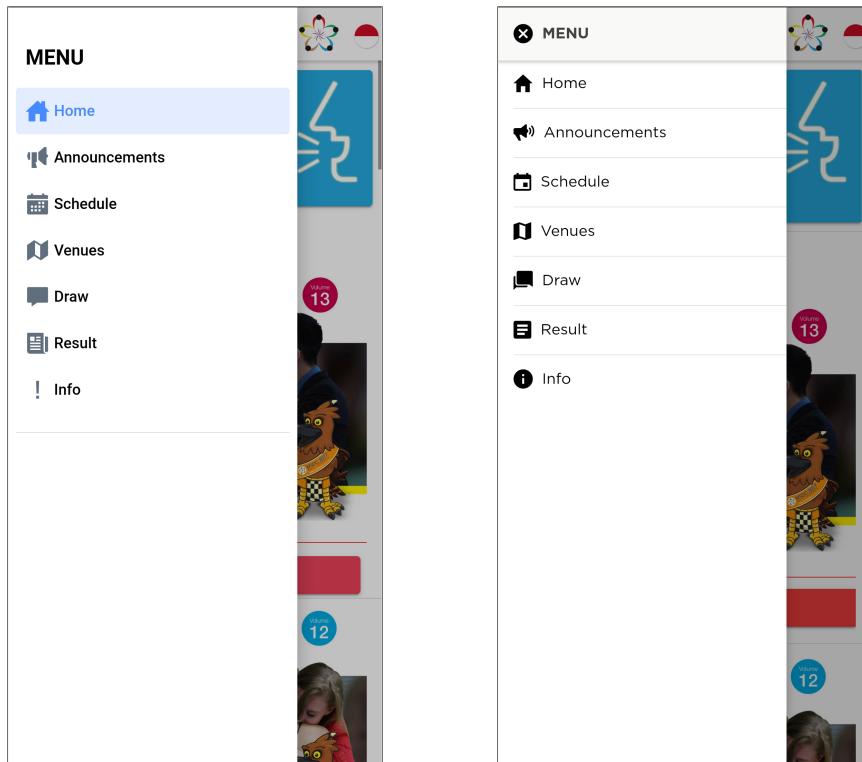
32   Halaman *venues* berisi kategori *venues* WSDC 2017 Bali. Setiap kategori yang ditampilkan  
33   merupakan sebuah tombol yang dapat diklik untuk mengarahkan pengguna ke halaman  
34   *venues map*. Tangkapan layar halaman *venues* dapat dilihat pada Gambar 5.9a. Lalu sebagai  
35   perbandingan, tangkapan layar halaman *venues* pada aplikasi WSDC 2017 Bali terdahulu  
36   dapat dilihat pada Gambar 5.9b.

37   8. Halaman *Venues Map*

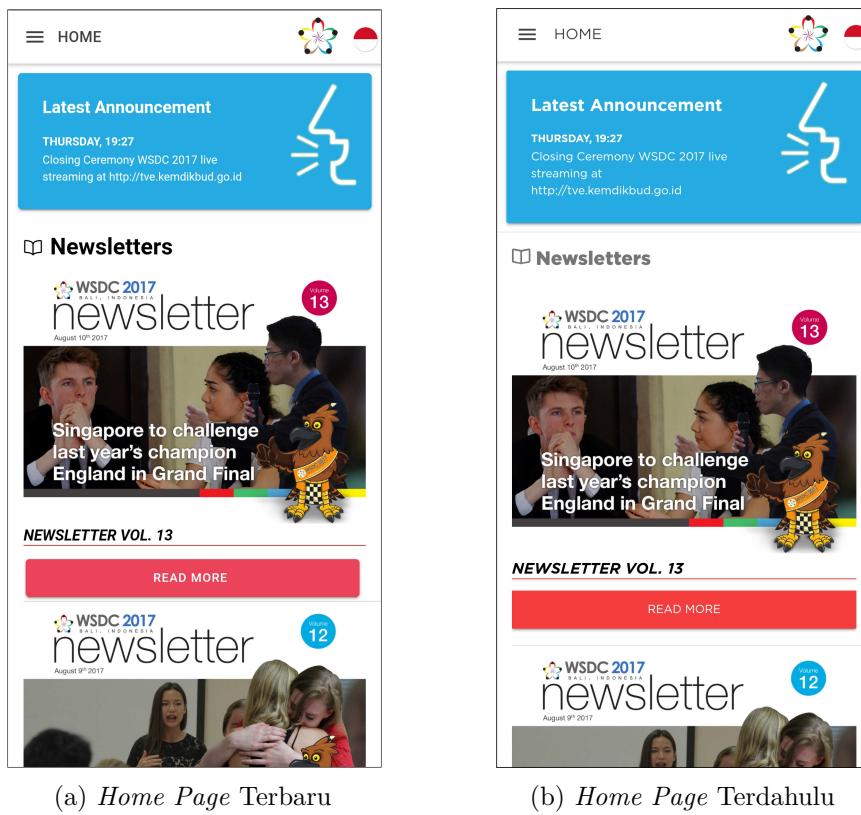
38   Halaman *venues map* berisi lokasi *venues* yang digunakan oleh WSDC 2017 Bali. Lokasi tersebut  
39   ditampilkan dengan peta, dan detail dari lokasi ditampilkan dengan *list* yang berisi nama  
40   dan lokasi *venues*, serta jarak dari pengguna ke lokasi *venues*. Tangkapan layar dari halaman  
41   *venues map* dapat dilihat pada Gambar 5.10a. Untuk perbandingan, tangkapan layar halaman  
42   *venues map* pada aplikasi WSDC 2017 Bali terdahulu dapat dilihat pada Gambar 5.10b.

(a) *Splash Screen Page Terbaru*(b) *Splash Screen Page Terdahulu*

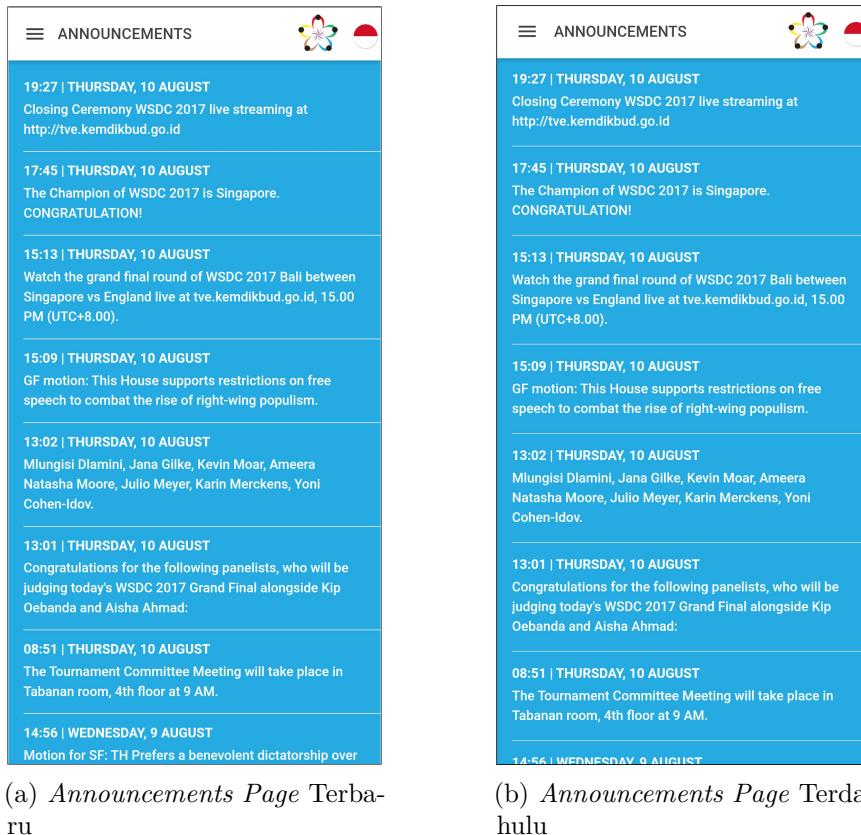
Gambar 5.1: Tangkapan Layar Halaman Splash Screen Aplikasi WSDC 2017 Bali

(a) *Sidemenu Terbaru*(b) *Sidemenu Terdahulu*

Gambar 5.2: Tangkapan Layar Sidemenu Aplikasi WSDC 2017 Bali



Gambar 5.3: Tangkapan Layar Halaman Home Aplikasi WSDC 2017 Bali



Gambar 5.4: Tangkapan Layar Halaman Announcements Aplikasi WSDC 2017 Bali

**(a) Draw Page Terbaru**

**Preliminary Round 1**  
(Prepared)

"This House would ban for-profit universities and colleges"

Venue: SMA Negeri 1 Denpasar

PROPOSITION	OPPOSITION
Argentina	Philippines
Romania	UAE
Bangladesh	Scotland
Wales	Turkey
Ghana	BYE United States
Nepal	Australia
Mexico	Pakistan
Barbados	Sri Lanka
South Korea	Slovenia
Malaysia	Tunisia BYE
Taiwan	Peru
Ireland	Sweden
Bermuda	England
Uganda	Greece

Venue: SMA Negeri 7 Denpasar

PROPOSITION	OPPOSITION
Czech Republic	China
Mongolia	Denmark
Indonesia	Estonia
Lithuania	Germany
	Hong Kong

**(b) Draw Page Terdahulu**

**Preliminary Round 1**  
(Prepared)

"This House would ban for-profit universities and colleges"

Venue: SMA Negeri 1 Denpasar

PROPOSITION	OPPOSITION
Argentina	Philippines
Romania	UAE
Bangladesh	Scotland
Wales	Turkey
Ghana	BYE United States
Nepal	Australia
Mexico	Pakistan
Barbados	Sri Lanka
South Korea	Slovenia
Malaysia	Tunisia BYE
Taiwan	Peru
Ireland	Sweden
Bermuda	England
Uganda	Greece

Venue: SMA Negeri 7 Denpasar

PROPOSITION	OPPOSITION
Czech Republic	China
Mongolia	Denmark
Indonesia	Estonia
Lithuania	Germany
	Hong Kong

Gambar 5.5: Tangkapan Layar Halaman Draw Aplikasi WSDC 2017 Bali

**(a) Info Page Terbaru**

**IMPORTANT CONTACTS**

**CO-CONVENORS**  
wsdc.indonesia@kemdikbud.go.id

- Ravio Patra
- Kristi Ardiana
- Rachmat Nur Cahyo
- Nyoman Radjin
- Hari Soegiharto
- Fonda Ambita Sari

**COMMON INDONESIAN PHRASES**

- I - *Saya*
- You - *Kamu*
- We - *Kami*
- They - *Mereka*
- He / She - *Dia*
- Welcome - *Selamat Datang*
- Hello (general greeting) - *Apa kabar?*
- Hello (on phone) - *Halo*
- How are you? - *Apa kabar?*
- Reply to 'How are you?' - *Baik*
- Long time no see - *Lama tidak bertemu*
- What's your name? - *Siapa nama anda?*
- My name is... - *Nama saya...*
- Where are you from? - *Anda berasal dari mana?*

**(b) Info Page Terdahulu**

**IMPORTANT CONTACTS**

**CO-CONVENORS**  
wsdc.indonesia@kemdikbud.go.id

- Ravio Patra
- Kristi Ardiana
- Rachmat Nur Cahyo
- Nyoman Radjin
- Hari Soegiharto
- Fonda Ambita Sari

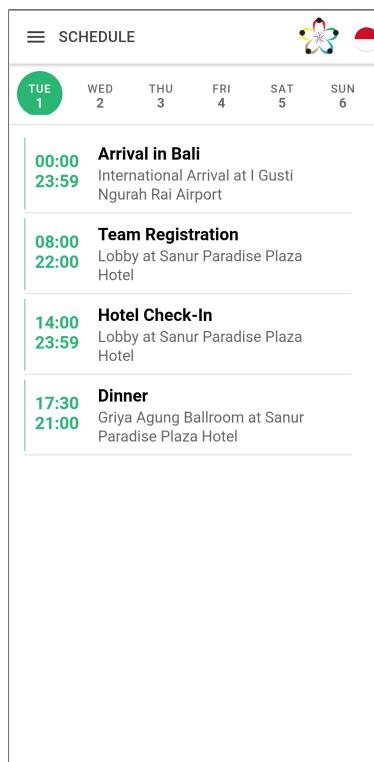
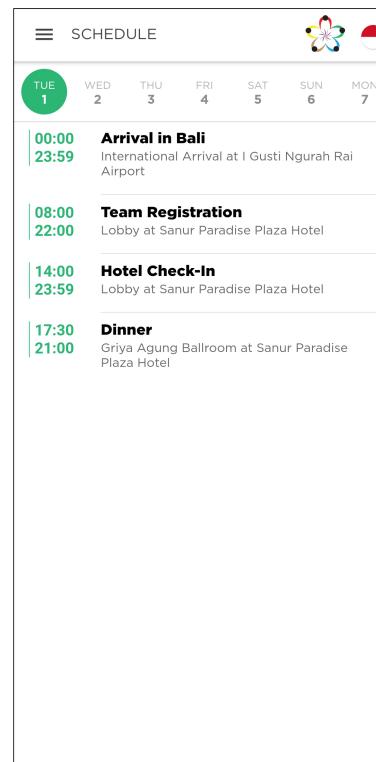
**COMMON INDONESIAN PHRASES**

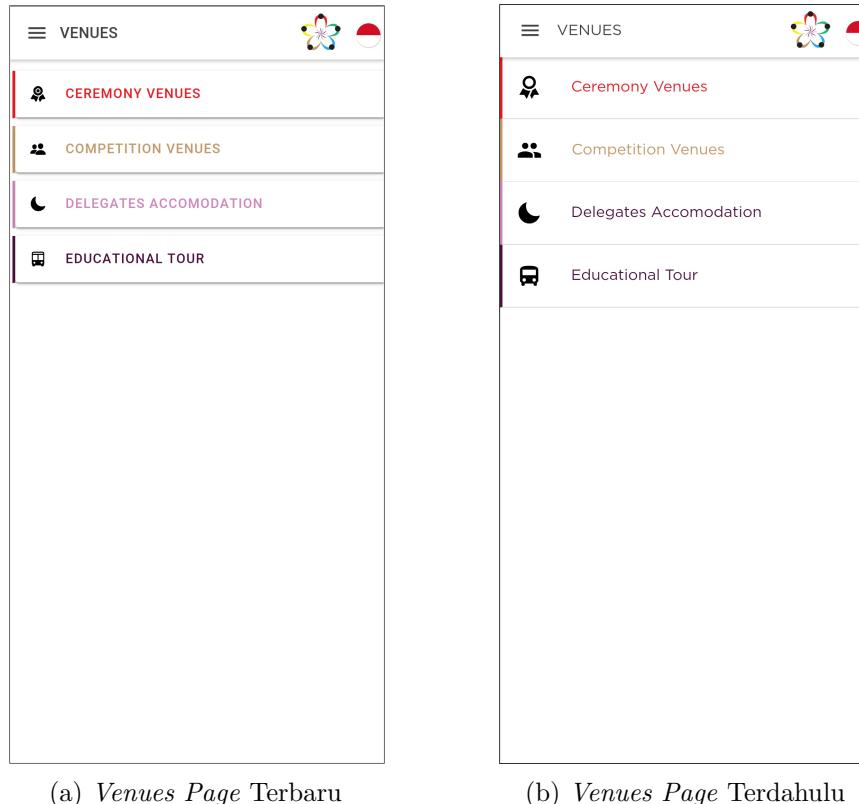
- I - *Saya*
- You - *Kamu*
- We - *Kami*
- They - *Mereka*
- He / She - *Dia*
- Welcome - *Selamat Datang*
- Hello (general greeting) - *Apa kabar?*
- Hello (on phone) - *Halo*
- How are you? - *Apa kabar?*
- Reply to 'How are you?' - *Baik*
- Long time no see - *Lama tidak bertemu*
- What's your name? - *Siapa nama anda?*
- My name is... - *Nama saya...*
- Where are you from? - *Anda berasal dari mana?*

(a) Info Page Terbaru

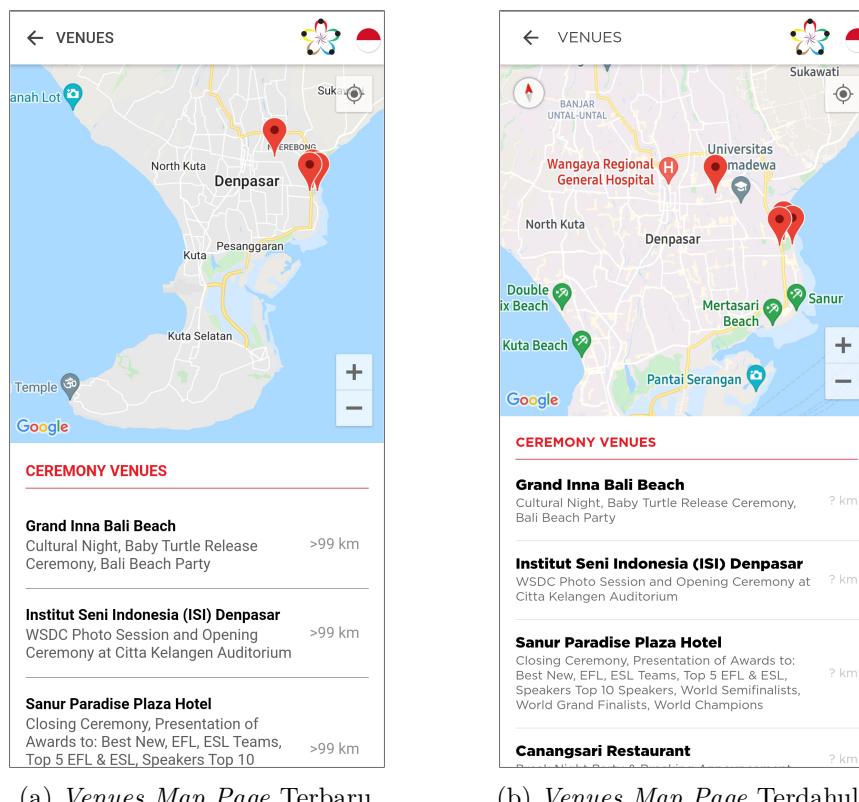
(b) Info Page Terdahulu

Gambar 5.6: Tangkapan Layar Halaman Info WSDC 2017 Bali

(a) *Result Page* Terbaru(b) *Result Page* TerdahuluGambar 5.7: Tangkapan Layar Halaman *Result* Aplikasi WSDC 2017 Bali(a) *Schedule Page* Terbaru(b) *Schedule Page* TerdahuluGambar 5.8: Tangkapan Layar Halaman *Schedule* Aplikasi WSDC 2017 Bali



Gambar 5.9: Tangkapan Layar Halaman *Venues* Aplikasi WSDC 2017 Bali



Gambar 5.10: Tangkapan Layar Halaman *Venues Map* Aplikasi WSDC 2017 Bali

## **5.2 Pengujian**

### **5.2.1 Pengujian Fungsional**

Pengujian fungsional dilakukan untuk mengetahui kesesuaian reaksi perangkat lunak dengan reaksi yang diharapkan berdasarkan aksi pengguna terhadap perangkat lunak. Perangkat yang digunakan untuk melakukan pengujian fungsional ini adalah sebuah perangkat emulator dari Android Studio yaitu Google Pixel 5 dengan versi Android 11, sebuah perangkat emulator Nox Player dengan versi Android 7.1.2, dan sebuah *smartphone* milik penulis yaitu Xiaomi Redmi Note 9 dengan versi Android 11. Tabel 5.2 merupakan hasil dari 20 tes kasus yang diujikan.

Tabel 5.1: Tabel Pengujian Fungsional

No	Aksi Pengguna	Reaksi yang diharapkan	Reaksi Perangkat Lunak
1	Pengguna menjalankan aplikasi	Splash Screen ditampilkan dan aplikasi menampilkan halaman home	Sesuai
2	Pengguna menekan tombol hamburger button di pojok kiri atas aplikasi	Sidemenu terbuka menampilkan menu	Sesuai
3	Pengguna melakukan swipe dari kiri layar ke kanan layar	Sidemenu terbuka menampilkan menu	Sesuai
4	Pengguna memilih menu Announcements pada Sidemenu	Aplikasi menampilkan halaman Announcements	Sesuai
5	Pengguna memilih menu Home pada Sidemenu	Aplikasi menampilkan halaman Home	Sesuai
6	Pengguna menekan tombol read more pada Home	Aplikasi mengarahkan pengguna untuk melihat newsletter	Sesuai
7	Pengguna menekan card Latest Announcements	Aplikasi mengarahkan pengguna ke halaman Announcements	Sesuai
8	Pengguna memilih menu Schedule pada Sidemenu	Aplikasi menampilkan halaman Schedule	Sesuai
9	Pengguna menekan tombol hari dan tanggal pada halaman Schedule	Aplikasi menampilkan jadwal yang ada pada hari dan tanggal yang dipilih	Sesuai
10	Pengguna melakukan swipe secara vertical baik dari kiri ke kanan maupun sebaliknya pada halaman Schedule	Aplikasi menampilkan jadwal yang ada pada hari dan tanggal sebelum maupun sesudahnya	Sesuai
11	Pengguna memilih menu Venues pada Sidemenu	Aplikasi menampilkan halaman Venues	Sesuai
12	Pengguna memilih kategori venues pada halaman Venues	Aplikasi menampilkan halaman Venues Map yang berisi peta dan lokasi venues	Sesuai
13	Pengguna menekan tombol lokasi pada map	Aplikasi menampilkan lokasi pengguna pada map dengan titik biru	Sesuai
14	Pengguna menekan tombol + pada map	Aplikasi melakukan zoom in pada map	Sesuai

Tabel 5.2: Lanjutan Tabel Pengujian Fungsional dari Halaman Sebelumnya

No	Aksi Pengguna	Reaksi yang diharapkan	Reaksi Perangkat Lunak
15	Pengguna menekan tombol – pada map	Aplikasi melakukan zoom out pada map	Sesuai
16	Pengguna menekan nama lokasi venues	Aplikasi melakukan zoom in mengarah ke lokasi yang dituju pada map	Sesuai
17	Pengguna memilih menu Draw pada Sidemenu	Aplikasi menampilkan halaman Draw	Sesuai
18	Pengguna memilih menu Result pada Sidemenu	Aplikasi menampilkan halaman Result	Sesuai
19	Pengguna memilih menu Info pada Sidemenu	Aplikasi menampilkan halaman Info	Sesuai
20	Pengguna menekan nomor telepon pada halama info	Aplikasi mengarahkan pengguna ke aplikasi pemanggilan	Sesuai

### **1 5.2.2 Pengujian Eksperimental**

2 Pengujian eksperimental dilakukan terhadap pengguna *smartphone* dengan sistem operasi Android.  
 3 Metode pengujian dilakukan dengan cara menyebarkan aplikasi yang dapat diunduh melalui Google  
 4 Drive <sup>1</sup>. Kemudian, pengguna diminta untuk mengunduh dan menjalankan aplikasi tersebut. Selain  
 5 itu, pengguna juga diminta untuk mengunduh aplikasi WSDC 2017 Bali terdahulu melalui Google  
 6 Play Store <sup>2</sup> dan menjalankannya. Setelah itu pengguna diminta untuk membandingkan kedua  
 7 aplikasi tersebut, dan mengisi beberapa pertanyaan terkait pengalaman menggunakan aplikasi  
 8 WSDC 2017 Bali melalui Google Form. Berikut ini merupakan pertanyaan dan rangkuman jawaban  
 9 dari hasil pengujian eksperimental terhadap sembilan responden sebagai berikut:

10 **1. Apa versi Android smartphone Anda?**

11 Seorang responden menjawab versi Android 5.1, seorang menjawab versi Android 8.0, seorang  
 12 menjawab versi Android 8.1, seorang menjawab versi Android 10, dan empat orang menjawab  
 13 versi Android 11.

14 **2. Saat pertama kali membuka aplikasi, apakah aplikasi WSDC 2017 Bali terbaru  
 15 menampilkan logo WSDC?**

16 Semua responden menjawab aplikasi WSDC 2017 Bali terbaru menampilkan logo WSDC.

17 **3. Apakah semua halaman memiliki isi nya masing-masing, dan tidak ada halaman  
 18 yang isinya kosong?**

19 Semua responden menjawab tidak ada halaman yang tidak memiliki isi.

20 **4. Apakah tombol GPS, *zoom in*, *zoom out*, dan lokasi *venues* yang terdapat pada  
 21 menu *Venues* dapat berfungsi dengan baik?**

22 Semua responden menjawab semua tombol berfungsi dengan normal.

<sup>1</sup>Tautan Google Drive aplikasi WSDC 2017 Bali dengan Ionic 6 yang diujikan kepada responden: <https://drive.google.com/file/d/1Np29U2dG58Pryp1cbrs-M3XfUcm39cSZ/view?usp=sharing>

<sup>2</sup>Tautan Google Play Store aplikasi WSDC 2017 Bali terdahulu: <https://play.google.com/store/apps/details?id=org.wsdc2017indonesia.app>

1   **5. Apakah Anda mengalami *crash*, *forced close*, atau kendala lain saat menggunakan**  
2   **aplikasi WSDC 2017 Bali terbaru?**

3   Sebanyak delapan responden menjawab tidak ada crash, forced close, atau kendala lain saat  
4   menggunakan aplikasi WSDC 2017 Bali terbaru, dan ada satu responden yang menjawab iya,  
5   namun tidak menjelaskan kendala apa yang terjadi.

6   **6. Apakah ada perbedaan positif yang signifikan dibandingkan dengan aplikasi**  
7   **terdahulu?**

8   Dua orang responden berpendapat bahwa tampilan *sidemenu* tampak lebih segar dan menarik.  
9   Lalu sebanyak satu responden berpendapat bahwa tampilan *icon* terlihat lebih beragam  
10   dan menarik. Kemudian sebanyak empat responden berpendapat bahwa aplikasi WSDC  
11   2017 Bali terbaru dapat dibuka dengan lebih cepat dibandingkan dengan aplikasi terdahulu.  
12   Lalu sebanyak satu responden berpendapat bahwa perubahan aplikasi menjadi lebih baik  
13   dari sebelumnya, satu responden menjawab perubahan yang terjadi hanya sedikit, dan satu  
14   responden menjawab tidak ada perubahan positif yang dirasakan.

15   **7. Apakah ada perbedaan negatif yang signifikan dibandingkan dengan aplikasi**  
16   **terdahulu?**

17   Sebanyak empat orang responden menjawab bahwa tidak ada perubahan negatif pada aplikasi  
18   WSDC 2017 Bali terbaru. Seorang responden menjawab *font* tulisan lebih kaku, dan halaman  
19   *draw* kualitasnya terlihat lebih rendah dibandingkan aplikasi terdahulu. Lalu seorang responden  
20   menjawab tampilan pada aplikasi yang terbaru dari menu yang ada di masing-masing *Venues*  
21   sedikit aneh, karena nama tempat dan alamatnya saling berdekatan tanpa ada jarak spasi.  
22   Dan seorang responden menjawab pemakaian aplikasi terbaru lebih boros.

23   **8. Secara keseluruhan, dengan skala 1-5, seberapa baik aplikasi WSDC 2017 Bali**  
24   **terbaru dibandingkan dengan aplikasi terdahulu?**

25   Seorang responden menjawab netral dengan skala 3, enam orang responden menjawab dengan  
26   skala 4, dan dua orang responden menjawab aplikasi WSDC 2017 Bali lebih baik dibandingkan  
27   aplikasi terdahulu dengan skala 5.

28   **9. Apakah Anda lebih memilih menggunakan aplikasi WSDC 2017 Bali terdahulu,**  
29   **atau yang terbaru?**

30   Sebanyak delapan responden memilih untuk menggunakan aplikasi WSDC 2017 Bali terbaru,  
31   sedangkan satu responden memilih untuk menggunakan aplikasi WSDC 2017 Bali terdahulu.

32   **10. Apakah terdapat kritik dan saran terhadap aplikasi WSDC 2017 Bali terbaru?**

33   Terdapat beberapa kritik dan saran dari responden sebagai berikut:

- 34   (a) Dapat memilih font yang lebih menarik dan nyaman untuk dibaca.
- 35   (b) Pada halaman *Result* diharapkan agar memiliki tampilan lebih menarik lagi.
- 36   (c) Ukuran aplikasi bisa dikecilkan.
- 37   (d) Pada halaman *venues* dimana tulisan headline dari list map dijauhkan sedikit dari tulisan  
38   body karena terlalu dekat.

<sup>1</sup>

## BAB 6

<sup>2</sup>

### KESIMPULAN DAN SARAN

#### <sup>3</sup> 6.1 Kesimpulan

<sup>4</sup> Dari hasil pembangunan aplikasi WSDC 2017 Bali menggunakan Ionic 6, didapatkan kesimpulan  
<sup>5</sup> sebagai berikut:

- <sup>6</sup> 1. Telah berhasil melakukan pembaruan aplikasi WSDC 2017 Bali dengan Ionic Framework versi  
<sup>7</sup> 6 yang sebelumnya menggunakan Ionic Framework versi 3.
- <sup>8</sup> 2. Aplikasi WSDC 2017 Bali telah dapat dijalankan pada perangkat dengan sistem operasi  
<sup>9</sup> Android <sup>1</sup>.

#### <sup>10</sup> 6.2 Saran

<sup>11</sup> Dari hasil penelitian dan pengujian termasuk dengan pengujian terhadap responden, berikut ini  
<sup>12</sup> merupakan beberapa saran untuk pengembangan lebih lanjut:

- <sup>13</sup> 1. Dapat memilih font yang lebih menarik dan nyaman untuk dibaca.
  - <sup>14</sup> 2. Dapat memperbaiki halaman *Result* agar memiliki tampilan lebih menarik.
  - <sup>15</sup> 3. Dapat mengecilkan ukuran aplikasi.
  - <sup>16</sup> 4. Pada halaman *venues* dimana tulisan headline dari list map dijauhkan sedikit dari tulisan  
<sup>17</sup> body karena terlalu dekat.
- <sup>18</sup> Namun menurut pengamatan penulis terhadap kritik dan saran pada poin 3, bahwa ukuran aplikasi  
<sup>19</sup> WSDC 2017 Bali terbaru sudah cukup kecil, yaitu 25MB. Sedangkan untuk poin 1 dan 2 bersifat  
<sup>20</sup> subjektif, sehingga tidak akan diimplementasikan oleh penulis.

---

<sup>1</sup>Selain dijalankan pada perangkat Android, aplikasi WSDC 2017 Bali dengan Ionic 6 juga telah berhasil diuji dan dijalankan pada perangkat iOS. Pengujian tersebut dilakukan oleh dosen pembimbing, dan dijalankan terpisah dari dokumen ini.



## DAFTAR REFERENSI

- [1] Yusuf, S. (2016) *Ionic Framework By Example*, 1st edition. Pact Publishing Ltd., Birmingham, UK.
- [2] Waranashiwar, J. dan Ukey, M. (2018) Ionic framework with angular for hybrid app development. *International Journal of New Technology and Research*, **4**, 01–02.
- [3] Griffith, C. (2017) *Mobile App Development with Ionic : Cross-Platform Apps with Ionic, Angular and Cordova*, 1st edition. O'Reilly Media, Inc., California, USA.
- [4] Grønli, T.-M., Biørn-Hansen, A., dan Majchrzak, T. A. (2019) Median trajectories using well-visited regions and shortest paths software development for mobile computing the internet of things and wearable devices: Inspecting the past to understand the future. *Proceedings of the 52nd Hawaii International Conference on System Sciences*, Grand Wailea, Hawaii, 8–11 January, pp. 7451–7460. University of Hawaii, Manoa.
- [5] Wohlgethan, E. (2018) Supporting web development decisions by comparing three major javascript frameworks: Angular, react and vue.js. Thesis. Hochschule für angewandte Wissenschaften Hamburg, Germany.
- [6] Emmit A. Scott, J. (2015) *SPA Design and Architecture: Understanding single-page web applications*, 1st edition. Manning Publications, New York, USA.
- [7] World Schools Debate Championship (2021) WSDC. <https://wsdcdebate.org/history>. [Online; diakses 8-Juli-2021].
- [8] Moiseev, A. dan Fain, Y. (2018) *Angular Development with TypeScript*, 2nd edition. Manning Publications, New York, USA.
- [9] Kunz, G. (2018) *Mastering Angular Components: Build component-based user interfaces using Angular*, 2nd edition. Pact Publishing Ltd., Birmingham, UK.
- [10] Huber, S., Demetz, L., dan Felderer, M. (2021) Pwa vs the others: A comparative study on the ui energy-efficiency of progressive web apps. *Web Engineering*, Switzerland, 11 May, pp. 464–479. Springer International Publishing.
- [11] Gonsalves, M. (2018) Evaluating the mobile development frameworks apache cordova and flutter and their impact on the development process and application characteristics. Thesis. California State University, Chico, California, USA.



# LAMPIRAN A

## KODE PROGRAM

### A.1 Komponen Announcements

```
1 import { HttpClient } from '@angular/common/http';
2 import { Component, OnInit } from '@angular/core';
3 import { Storage } from '@ionic/storage';
4 import { ToastController } from '@ionic/angular';
5
6 @Component({
7   selector: 'app-announcement',
8   templateUrl: './announcement.page.html',
9   styleUrls: ['./announcement.page.scss'],
10 })
11
12 export class AnnouncementPage implements OnInit {
13   announcements: Array<{ localtime: string, message: string }>;
14
15   constructor(private http: HttpClient, private storage: Storage, public
16     toastController: ToastController) { }
17
18   ngOnInit(): void {
19     this.storage.get('wsdcDataStorage').then((data) => {
20       this.announcements = data.announcements;
21     })
22   }
23
24   doRefresh(refresher) {
25     console.log('Begin async operation');
26     this.http.get('https://wsdc.dnartworks.com/wsdc_data.json').subscribe(
27       (data: any) => {
28         this.storage.clear();
29         this.storage.set('wsdcDataStorage', data);
30         this.announcements = data.announcements;
31         console.log("Data updated!");
32         if (refresher != 0)
33           refresher.target.complete();
34       },
35       (error) => {
36         this.presentConnectionAlert();
37         refresher.target.complete();
38         console.log('error in XMLHttpRequest JSON');
39       });
40     setTimeout(() => {
41       console.log('Async operation has ended');
42       refresher.target.complete();
43     }, 30000);
44   }
45
46   async presentConnectionAlert() {
47     let toast = await this.toastController.create({
```

```

47     message: 'Failed to refresh information',
48     duration: 3000
49   });
50   toast.present();
51 }
52
53 formatDatetime(sqlDatetime: string) {
54   var date = new Date(sqlDatetime.substring(0, 10));
55   var dayNames = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"];
56   var monthNames = ["January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"];
57   var dayOfWeek = date.getDay();
58   var day = date.getDate();
59   var monthIndex = date.getMonth();
60   var time=sqlDatetime.substring(16,4)
61   return time.substring(7) + ' | ' + dayNames[dayOfWeek] + ', ' + day + ' ' +
62   monthNames[monthIndex];
63 }

```

Kode A.1: annoncement.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Announcements</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
12     <ion-refresher-content></ion-refresher-content>
13   </ion-refresher>
14   <ion-list>
15     <ion-item color="wsdc-blue" *ngFor="let announcement of announcements; let i =
16       index">
17       <ion-label class="ion-text-wrap">
18         <h3>{{ formatDatetime(announcement.localtime) }}</h3>
19         <p>{{ announcement.message }}</p>
20       </ion-label>
21     </ion-item>
22   </ion-list>
23 </ion-content>

```

Kode A.2: annoncement.page.html

## A.2 Komponen Draw

```

1 import { Component, ElementRef, OnInit, ViewChild } from '@angular/core';
2 import { LoadingController } from '@ionic/angular';
3 import { Storage } from '@ionic/storage';
4
5 @Component({
6   selector: 'app-draw',
7   templateUrl: './draw.page.html',
8   styleUrls: ['./draw.page.scss'],
9 })
10

```

```

11 export class DrawPage implements OnInit {
12   @ViewChild('drawIFrame') drawIFrame: ElementRef;
13   constructor(private storage: Storage, public loadingController: LoadingController
14   ) { }
15 
16   ngOnInit() {
17     this.storage.get('wsdcDataStorage').then((data) => {
18       this.drawIFrame.nativeElement.contentWindow.location.assign(data.draws);
19     });
20     this.presentLoading();
21   }
22 
23   async presentLoading() {
24     const loading = await this.loadingController.create({
25       message: 'Please wait...',
26       backdropDismiss: true // If true, the loading indicator will be dismissed
27       when the backdrop is clicked.
28     });
29     await loading.present();
30     setTimeout(() => {
31       loading.dismiss();
32     }, 1000);
33   }
34 
35   onDrawIframeLoad(){
36     let doc = this.drawIFrame.nativeElement.contentWindow.document;
37     let elements = [
38       doc.getElementById('header'),
39       doc.getElementById('page_header'),
40       doc.getElementById('footer')
41     ];
42     elements.forEach(function (element) {
43       if (element) {
44         element.style.display = 'none';
45       }
46     })
47     this.drawIFrame.nativeElement.style.display = 'block';
48   }
49 }

```

Kode A.3: draw.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Draw</ion-title>
7   </ion-toolbar>
8 </ion-header>
9 
10 <ion-content>
11   <iframe #drawIFrame (load)="onDrawIframeLoad()"></iframe>
12 </ion-content>

```

Kode A.4: draw.page.html

### A.3 Komponen Home

```

1 import { Component, OnInit } from '@angular/core';
2 import { HttpClient } from '@angular/common/http';
3 import { Brower } from '@capacitor/browser';
4 import { Storage } from '@ionic/storage';
5 import { Router } from '@angular/router';
6 import { SplashScreen } from '@capacitor/splash-screen';
7 import { ToastController } from '@ionic/angular';
8
9 @Component({
10   selector: 'app-home',
11   templateUrl: './home.page.html',
12   styleUrls: ['./home.page.scss'],
13 })
14
15 export class HomePage implements OnInit {
16   wsdcData:any;
17
18   constructor(private http: HttpClient,private storage: Storage,private router: Router,public toastController: ToastController) { }
19
20   ionViewDidEnter(){
21     SplashScreen.hide()
22   }
23
24   ngOnInit(){
25     this.storage.get('wsdcDataStorage').then((data) => {
26
27       if(data == null){
28         //Default from asset
29         this.http.get('../assets/json/wsdc_data.json').subscribe((data: any) => {
30           this.wsdcData = data;
31           this.storage.set('wsdcDataStorage',data);
32         },
33         error => {
34           this.showToast('Failed to refresh information from local storage');
35         });
36       }else{
37         this.wsdcData = data;
38       }
39
40       // Refresh data
41       setTimeout(() => {
42         this.http.get('http://wsdc.dnartworks.com/wsdc_data.json')
43         .subscribe((data: any) => {
44           this.storage.set('wsdcDataStorage', data);
45           this.wsdcData = data;
46         },
47         error => {
48           this.showToast('Failed to refresh information');
49         });
50       }, 1000);
51     })
52   }
53
54   doRefresh(refresher) {
55     this.http.get('https://wsdc.dnartworks.com/wsdc_data.json').subscribe((data: any) => {
56       this.storage.clear();
57       this.storage.set('wsdcDataStorage',data);
58       this.wsdcData = data;
59     })
60   }
61
62   showToast(message) {
63     const toast = this.toastController.create({
64       message: message,
65       duration: 3000
66     });
67     toast.present();
68   }
69
70   ionViewWillLeave() {
71     SplashScreen.show();
72   }
73
74   ionViewDidEnter() {
75     SplashScreen.hide();
76   }
77
78   ionViewWillLeave() {
79     SplashScreen.show();
80   }
81
82   ionViewDidEnter() {
83     SplashScreen.hide();
84   }
85
86   ionViewWillLeave() {
87     SplashScreen.show();
88   }
89
90   ionViewDidEnter() {
91     SplashScreen.hide();
92   }
93
94   ionViewWillLeave() {
95     SplashScreen.show();
96   }
97
98   ionViewDidEnter() {
99     SplashScreen.hide();
100   }
101
102   ionViewWillLeave() {
103     SplashScreen.show();
104   }
105
106   ionViewDidEnter() {
107     SplashScreen.hide();
108   }
109
110   ionViewWillLeave() {
111     SplashScreen.show();
112   }
113
114   ionViewDidEnter() {
115     SplashScreen.hide();
116   }
117
118   ionViewWillLeave() {
119     SplashScreen.show();
120   }
121
122   ionViewDidEnter() {
123     SplashScreen.hide();
124   }
125
126   ionViewWillLeave() {
127     SplashScreen.show();
128   }
129
130   ionViewDidEnter() {
131     SplashScreen.hide();
132   }
133
134   ionViewWillLeave() {
135     SplashScreen.show();
136   }
137
138   ionViewDidEnter() {
139     SplashScreen.hide();
140   }
141
142   ionViewWillLeave() {
143     SplashScreen.show();
144   }
145
146   ionViewDidEnter() {
147     SplashScreen.hide();
148   }
149
150   ionViewWillLeave() {
151     SplashScreen.show();
152   }
153
154   ionViewDidEnter() {
155     SplashScreen.hide();
156   }
157
158   ionViewWillLeave() {
159     SplashScreen.show();
160   }
161
162   ionViewDidEnter() {
163     SplashScreen.hide();
164   }
165
166   ionViewWillLeave() {
167     SplashScreen.show();
168   }
169
170   ionViewDidEnter() {
171     SplashScreen.hide();
172   }
173
174   ionViewWillLeave() {
175     SplashScreen.show();
176   }
177
178   ionViewDidEnter() {
179     SplashScreen.hide();
180   }
181
182   ionViewWillLeave() {
183     SplashScreen.show();
184   }
185
186   ionViewDidEnter() {
187     SplashScreen.hide();
188   }
189
190   ionViewWillLeave() {
191     SplashScreen.show();
192   }
193
194   ionViewDidEnter() {
195     SplashScreen.hide();
196   }
197
198   ionViewWillLeave() {
199     SplashScreen.show();
200   }
201
202   ionViewDidEnter() {
203     SplashScreen.hide();
204   }
205
206   ionViewWillLeave() {
207     SplashScreen.show();
208   }
209
210   ionViewDidEnter() {
211     SplashScreen.hide();
212   }
213
214   ionViewWillLeave() {
215     SplashScreen.show();
216   }
217
218   ionViewDidEnter() {
219     SplashScreen.hide();
220   }
221
222   ionViewWillLeave() {
223     SplashScreen.show();
224   }
225
226   ionViewDidEnter() {
227     SplashScreen.hide();
228   }
229
230   ionViewWillLeave() {
231     SplashScreen.show();
232   }
233
234   ionViewDidEnter() {
235     SplashScreen.hide();
236   }
237
238   ionViewWillLeave() {
239     SplashScreen.show();
240   }
241
242   ionViewDidEnter() {
243     SplashScreen.hide();
244   }
245
246   ionViewWillLeave() {
247     SplashScreen.show();
248   }
249
250   ionViewDidEnter() {
251     SplashScreen.hide();
252   }
253
254   ionViewWillLeave() {
255     SplashScreen.show();
256   }
257
258   ionViewDidEnter() {
259     SplashScreen.hide();
260   }
261
262   ionViewWillLeave() {
263     SplashScreen.show();
264   }
265
266   ionViewDidEnter() {
267     SplashScreen.hide();
268   }
269
270   ionViewWillLeave() {
271     SplashScreen.show();
272   }
273
274   ionViewDidEnter() {
275     SplashScreen.hide();
276   }
277
278   ionViewWillLeave() {
279     SplashScreen.show();
280   }
281
282   ionViewDidEnter() {
283     SplashScreen.hide();
284   }
285
286   ionViewWillLeave() {
287     SplashScreen.show();
288   }
289
290   ionViewDidEnter() {
291     SplashScreen.hide();
292   }
293
294   ionViewWillLeave() {
295     SplashScreen.show();
296   }
297
298   ionViewDidEnter() {
299     SplashScreen.hide();
300   }
301
302   ionViewWillLeave() {
303     SplashScreen.show();
304   }
305
306   ionViewDidEnter() {
307     SplashScreen.hide();
308   }
309
310   ionViewWillLeave() {
311     SplashScreen.show();
312   }
313
314   ionViewDidEnter() {
315     SplashScreen.hide();
316   }
317
318   ionViewWillLeave() {
319     SplashScreen.show();
320   }
321
322   ionViewDidEnter() {
323     SplashScreen.hide();
324   }
325
326   ionViewWillLeave() {
327     SplashScreen.show();
328   }
329
330   ionViewDidEnter() {
331     SplashScreen.hide();
332   }
333
334   ionViewWillLeave() {
335     SplashScreen.show();
336   }
337
338   ionViewDidEnter() {
339     SplashScreen.hide();
340   }
341
342   ionViewWillLeave() {
343     SplashScreen.show();
344   }
345
346   ionViewDidEnter() {
347     SplashScreen.hide();
348   }
349
350   ionViewWillLeave() {
351     SplashScreen.show();
352   }
353
354   ionViewDidEnter() {
355     SplashScreen.hide();
356   }
357
358   ionViewWillLeave() {
359     SplashScreen.show();
360   }
361
362   ionViewDidEnter() {
363     SplashScreen.hide();
364   }
365
366   ionViewWillLeave() {
367     SplashScreen.show();
368   }
369
370   ionViewDidEnter() {
371     SplashScreen.hide();
372   }
373
374   ionViewWillLeave() {
375     SplashScreen.show();
376   }
377
378   ionViewDidEnter() {
379     SplashScreen.hide();
380   }
381
382   ionViewWillLeave() {
383     SplashScreen.show();
384   }
385
386   ionViewDidEnter() {
387     SplashScreen.hide();
388   }
389
390   ionViewWillLeave() {
391     SplashScreen.show();
392   }
393
394   ionViewDidEnter() {
395     SplashScreen.hide();
396   }
397
398   ionViewWillLeave() {
399     SplashScreen.show();
400   }
401
402   ionViewDidEnter() {
403     SplashScreen.hide();
404   }
405
406   ionViewWillLeave() {
407     SplashScreen.show();
408   }
409
410   ionViewDidEnter() {
411     SplashScreen.hide();
412   }
413
414   ionViewWillLeave() {
415     SplashScreen.show();
416   }
417
418   ionViewDidEnter() {
419     SplashScreen.hide();
420   }
421
422   ionViewWillLeave() {
423     SplashScreen.show();
424   }
425
426   ionViewDidEnter() {
427     SplashScreen.hide();
428   }
429
430   ionViewWillLeave() {
431     SplashScreen.show();
432   }
433
434   ionViewDidEnter() {
435     SplashScreen.hide();
436   }
437
438   ionViewWillLeave() {
439     SplashScreen.show();
440   }
441
442   ionViewDidEnter() {
443     SplashScreen.hide();
444   }
445
446   ionViewWillLeave() {
447     SplashScreen.show();
448   }
449
450   ionViewDidEnter() {
451     SplashScreen.hide();
452   }
453
454   ionViewWillLeave() {
455     SplashScreen.show();
456   }
457
458   ionViewDidEnter() {
459     SplashScreen.hide();
460   }
461
462   ionViewWillLeave() {
463     SplashScreen.show();
464   }
465
466   ionViewDidEnter() {
467     SplashScreen.hide();
468   }
469
470   ionViewWillLeave() {
471     SplashScreen.show();
472   }
473
474   ionViewDidEnter() {
475     SplashScreen.hide();
476   }
477
478   ionViewWillLeave() {
479     SplashScreen.show();
480   }
481
482   ionViewDidEnter() {
483     SplashScreen.hide();
484   }
485
486   ionViewWillLeave() {
487     SplashScreen.show();
488   }
489
490   ionViewDidEnter() {
491     SplashScreen.hide();
492   }
493
494   ionViewWillLeave() {
495     SplashScreen.show();
496   }
497
498   ionViewDidEnter() {
499     SplashScreen.hide();
500   }
501
502   ionViewWillLeave() {
503     SplashScreen.show();
504   }
505
506   ionViewDidEnter() {
507     SplashScreen.hide();
508   }
509
510   ionViewWillLeave() {
511     SplashScreen.show();
512   }
513
514   ionViewDidEnter() {
515     SplashScreen.hide();
516   }
517
518   ionViewWillLeave() {
519     SplashScreen.show();
520   }
521
522   ionViewDidEnter() {
523     SplashScreen.hide();
524   }
525
526   ionViewWillLeave() {
527     SplashScreen.show();
528   }
529
530   ionViewDidEnter() {
531     SplashScreen.hide();
532   }
533
534   ionViewWillLeave() {
535     SplashScreen.show();
536   }
537
538   ionViewDidEnter() {
539     SplashScreen.hide();
540   }
541
542   ionViewWillLeave() {
543     SplashScreen.show();
544   }
545
546   ionViewDidEnter() {
547     SplashScreen.hide();
548   }
549
550   ionViewWillLeave() {
551     SplashScreen.show();
552   }
553
554   ionViewDidEnter() {
555     SplashScreen.hide();
556   }
557
558   ionViewWillLeave() {
559     SplashScreen.show();
560   }
561
562   ionViewDidEnter() {
563     SplashScreen.hide();
564   }
565
566   ionViewWillLeave() {
567     SplashScreen.show();
568   }
569
570   ionViewDidEnter() {
571     SplashScreen.hide();
572   }
573
574   ionViewWillLeave() {
575     SplashScreen.show();
576   }
577
578   ionViewDidEnter() {
579     SplashScreen.hide();
580   }
581
582   ionViewWillLeave() {
583     SplashScreen.show();
584   }
585
586   ionViewDidEnter() {
587     SplashScreen.hide();
588   }
589
590   ionViewWillLeave() {
591     SplashScreen.show();
592   }
593
594   ionViewDidEnter() {
595     SplashScreen.hide();
596   }
597
598   ionViewWillLeave() {
599     SplashScreen.show();
600   }
601
602   ionViewDidEnter() {
603     SplashScreen.hide();
604   }
605
606   ionViewWillLeave() {
607     SplashScreen.show();
608   }
609
610   ionViewDidEnter() {
611     SplashScreen.hide();
612   }
613
614   ionViewWillLeave() {
615     SplashScreen.show();
616   }
617
618   ionViewDidEnter() {
619     SplashScreen.hide();
620   }
621
622   ionViewWillLeave() {
623     SplashScreen.show();
624   }
625
626   ionViewDidEnter() {
627     SplashScreen.hide();
628   }
629
630   ionViewWillLeave() {
631     SplashScreen.show();
632   }
633
634   ionViewDidEnter() {
635     SplashScreen.hide();
636   }
637
638   ionViewWillLeave() {
639     SplashScreen.show();
640   }
641
642   ionViewDidEnter() {
643     SplashScreen.hide();
644   }
645
646   ionViewWillLeave() {
647     SplashScreen.show();
648   }
649
650   ionViewDidEnter() {
651     SplashScreen.hide();
652   }
653
654   ionViewWillLeave() {
655     SplashScreen.show();
656   }
657
658   ionViewDidEnter() {
659     SplashScreen.hide();
660   }
661
662   ionViewWillLeave() {
663     SplashScreen.show();
664   }
665
666   ionViewDidEnter() {
667     SplashScreen.hide();
668   }
669
670   ionViewWillLeave() {
671     SplashScreen.show();
672   }
673
674   ionViewDidEnter() {
675     SplashScreen.hide();
676   }
677
678   ionViewWillLeave() {
679     SplashScreen.show();
680   }
681
682   ionViewDidEnter() {
683     SplashScreen.hide();
684   }
685
686   ionViewWillLeave() {
687     SplashScreen.show();
688   }
689
690   ionViewDidEnter() {
691     SplashScreen.hide();
692   }
693
694   ionViewWillLeave() {
695     SplashScreen.show();
696   }
697
698   ionViewDidEnter() {
699     SplashScreen.hide();
700   }
701
702   ionViewWillLeave() {
703     SplashScreen.show();
704   }
705
706   ionViewDidEnter() {
707     SplashScreen.hide();
708   }
709
710   ionViewWillLeave() {
711     SplashScreen.show();
712   }
713
714   ionViewDidEnter() {
715     SplashScreen.hide();
716   }
717
718   ionViewWillLeave() {
719     SplashScreen.show();
720   }
721
722   ionViewDidEnter() {
723     SplashScreen.hide();
724   }
725
726   ionViewWillLeave() {
727     SplashScreen.show();
728   }
729
730   ionViewDidEnter() {
731     SplashScreen.hide();
732   }
733
734   ionViewWillLeave() {
735     SplashScreen.show();
736   }
737
738   ionViewDidEnter() {
739     SplashScreen.hide();
740   }
741
742   ionViewWillLeave() {
743     SplashScreen.show();
744   }
745
746   ionViewDidEnter() {
747     SplashScreen.hide();
748   }
749
750   ionViewWillLeave() {
751     SplashScreen.show();
752   }
753
754   ionViewDidEnter() {
755     SplashScreen.hide();
756   }
757
758   ionViewWillLeave() {
759     SplashScreen.show();
760   }
761
762   ionViewDidEnter() {
763     SplashScreen.hide();
764   }
765
766   ionViewWillLeave() {
767     SplashScreen.show();
768   }
769
770   ionViewDidEnter() {
771     SplashScreen.hide();
772   }
773
774   ionViewWillLeave() {
775     SplashScreen.show();
776   }
777
778   ionViewDidEnter() {
779     SplashScreen.hide();
780   }
781
782   ionViewWillLeave() {
783     SplashScreen.show();
784   }
785
786   ionViewDidEnter() {
787     SplashScreen.hide();
788   }
789
790   ionViewWillLeave() {
791     SplashScreen.show();
792   }
793
794   ionViewDidEnter() {
795     SplashScreen.hide();
796   }
797
798   ionViewWillLeave() {
799     SplashScreen.show();
800   }
801
802   ionViewDidEnter() {
803     SplashScreen.hide();
804   }
805
806   ionViewWillLeave() {
807     SplashScreen.show();
808   }
809
810   ionViewDidEnter() {
811     SplashScreen.hide();
812   }
813
814   ionViewWillLeave() {
815     SplashScreen.show();
816   }
817
818   ionViewDidEnter() {
819     SplashScreen.hide();
820   }
821
822   ionViewWillLeave() {
823     SplashScreen.show();
824   }
825
826   ionViewDidEnter() {
827     SplashScreen.hide();
828   }
829
830   ionViewWillLeave() {
831     SplashScreen.show();
832   }
833
834   ionViewDidEnter() {
835     SplashScreen.hide();
836   }
837
838   ionViewWillLeave() {
839     SplashScreen.show();
840   }
841
842   ionViewDidEnter() {
843     SplashScreen.hide();
844   }
845
846   ionViewWillLeave() {
847     SplashScreen.show();
848   }
849
850   ionViewDidEnter() {
851     SplashScreen.hide();
852   }
853
854   ionViewWillLeave() {
855     SplashScreen.show();
856   }
857
858   ionViewDidEnter() {
859     SplashScreen.hide();
860   }
861
862   ionViewWillLeave() {
863     SplashScreen.show();
864   }
865
866   ionViewDidEnter() {
867     SplashScreen.hide();
868   }
869
870   ionViewWillLeave() {
871     SplashScreen.show();
872   }
873
874   ionViewDidEnter() {
875     SplashScreen.hide();
876   }
877
878   ionViewWillLeave() {
879     SplashScreen.show();
880   }
881
882   ionViewDidEnter() {
883     SplashScreen.hide();
884   }
885
886   ionViewWillLeave() {
887     SplashScreen.show();
888   }
889
890   ionViewDidEnter() {
891     SplashScreen.hide();
892   }
893
894   ionViewWillLeave() {
895     SplashScreen.show();
896   }
897
898   ionViewDidEnter() {
899     SplashScreen.hide();
900   }
901
902   ionViewWillLeave() {
903     SplashScreen.show();
904   }
905
906   ionViewDidEnter() {
907     SplashScreen.hide();
908   }
909
910   ionViewWillLeave() {
911     SplashScreen.show();
912   }
913
914   ionViewDidEnter() {
915     SplashScreen.hide();
916   }
917
918   ionViewWillLeave() {
919     SplashScreen.show();
920   }
921
922   ionViewDidEnter() {
923     SplashScreen.hide();
924   }
925
926   ionViewWillLeave() {
927     SplashScreen.show();
928   }
929
930   ionViewDidEnter() {
931     SplashScreen.hide();
932   }
933
934   ionViewWillLeave() {
935     SplashScreen.show();
936   }
937
938   ionViewDidEnter() {
939     SplashScreen.hide();
940   }
941
942   ionViewWillLeave() {
943     SplashScreen.show();
944   }
945
946   ionViewDidEnter() {
947     SplashScreen.hide();
948   }
949
950   ionViewWillLeave() {
951     SplashScreen.show();
952   }
953
954   ionViewDidEnter() {
955     SplashScreen.hide();
956   }
957
958   ionViewWillLeave() {
959     SplashScreen.show();
960   }
961
962   ionViewDidEnter() {
963     SplashScreen.hide();
964   }
965
966   ionViewWillLeave() {
967     SplashScreen.show();
968   }
969
970   ionViewDidEnter() {
971     SplashScreen.hide();
972   }
973
974   ionViewWillLeave() {
975     SplashScreen.show();
976   }
977
978   ionViewDidEnter() {
979     SplashScreen.hide();
980   }
981
982   ionViewWillLeave() {
983     SplashScreen.show();
984   }
985
986   ionViewDidEnter() {
987     SplashScreen.hide();
988   }
989
990   ionViewWillLeave() {
991     SplashScreen.show();
992   }
993
994   ionViewDidEnter() {
995     SplashScreen.hide();
996   }
997
998   ionViewWillLeave() {
999     SplashScreen.show();
1000   }
1001
1002   ionViewDidEnter() {
1003     SplashScreen.hide();
1004   }
1005
1006   ionViewWillLeave() {
1007     SplashScreen.show();
1008   }
1009
1010   ionViewDidEnter() {
1011     SplashScreen.hide();
1012   }
1013
1014   ionViewWillLeave() {
1015     SplashScreen.show();
1016   }
1017
1018   ionViewDidEnter() {
1019     SplashScreen.hide();
1020   }
1021
1022   ionViewWillLeave() {
1023     SplashScreen.show();
1024   }
1025
1026   ionViewDidEnter() {
1027     SplashScreen.hide();
1028   }
1029
1030   ionViewWillLeave() {
1031     SplashScreen.show();
1032   }
1033
1034   ionViewDidEnter() {
1035     SplashScreen.hide();
1036   }
1037
1038   ionViewWillLeave() {
1039     SplashScreen.show();
1040   }
1041
1042   ionViewDidEnter() {
1043     SplashScreen.hide();
1044   }
1045
1046   ionViewWillLeave() {
1047     SplashScreen.show();
1048   }
1049
1050   ionViewDidEnter() {
1051     SplashScreen.hide();
1052   }
1053
1054   ionViewWillLeave() {
1055     SplashScreen.show();
1056   }
1057
1058   ionViewDidEnter() {
1059     SplashScreen.hide();
1060   }
1061
1062   ionViewWillLeave() {
1063     SplashScreen.show();
1064   }
1065
1066   ionViewDidEnter() {
1067     SplashScreen.hide();
1068   }
1069
1070   ionViewWillLeave() {
1071     SplashScreen.show();
1072   }
1073
1074   ionViewDidEnter() {
1075     SplashScreen.hide();
1076   }
1077
1078   ionViewWillLeave() {
1079     SplashScreen.show();
1080   }
1081
1082   ionViewDidEnter() {
1083     SplashScreen.hide();
1084   }
1085
1086   ionViewWillLeave() {
1087     SplashScreen.show();
1088   }
1089
1090   ionViewDidEnter() {
1091     SplashScreen.hide();
1092   }
1093
1094   ionViewWillLeave() {
1095     SplashScreen.show();
1096   }
1097
1098   ionViewDidEnter() {
1099     SplashScreen.hide();
1100   }
1101
1102   ionViewWillLeave() {
1103     SplashScreen.show();
1104   }
1105
1106   ionViewDidEnter() {
1107     SplashScreen.hide();
1108   }
1109
1110   ionViewWillLeave() {
1111     SplashScreen.show();
1112   }
1113
1114   ionViewDidEnter() {
1115     SplashScreen.hide();
1116   }
1117
1118   ionViewWillLeave() {
1119     SplashScreen.show();
1120   }
1121
1122   ionViewDidEnter() {
1123     SplashScreen.hide();
1124   }
1125
1126   ionViewWillLeave() {
1127     SplashScreen.show();
1128   }
1129
1130   ionViewDidEnter() {
1131     SplashScreen.hide();
1132   }
1133
1134   ionViewWillLeave() {
1135     SplashScreen.show();
1136   }
1137
1138   ionViewDidEnter() {
1139     SplashScreen.hide();
1140   }
1141
1142   ionViewWillLeave() {
1143     SplashScreen.show();
1144   }
1145
1146   ionViewDidEnter() {
1147     SplashScreen.hide();
1148   }
1149
1150   ionViewWillLeave() {
1151     SplashScreen.show();
1152   }
1153
1154   ionViewDidEnter() {
1155     SplashScreen.hide();
1156   }
1157
1158   ionViewWillLeave() {
1159     SplashScreen.show();
1160   }
1161
1162   ionViewDidEnter() {
1163     SplashScreen.hide();
1164   }
1165
1166   ionViewWillLeave() {
1167     SplashScreen.show();
1168   }
1169
1170   ionViewDidEnter() {
1171     SplashScreen.hide();
1172   }
1173
1174   ionViewWillLeave() {
1175     SplashScreen.show();
1176   }
1177
1178   ionViewDidEnter() {
1179     SplashScreen.hide();
1180   }
1181
1182   ionViewWillLeave() {
1183     SplashScreen.show();
1184   }
1185
1186   ionViewDidEnter() {
1187     SplashScreen.hide();
1188   }
1189
1190   ionViewWillLeave() {
1191     SplashScreen.show();
1192   }
1193
1194   ionViewDidEnter() {
1195     SplashScreen.hide();
1196   }
1197
1198   ionViewWillLeave() {
1199     SplashScreen.show();
1200   }
1201
1202   ionViewDidEnter() {
1203     SplashScreen.hide();
1204   }
1205
1206   ionViewWillLeave() {
1207     SplashScreen.show();
1208   }
1209
1210   ionViewDidEnter() {
1211     SplashScreen.hide();
1212   }
1213
1214   ionViewWillLeave() {
1215     SplashScreen.show();
1216   }
1217
1218   ionViewDidEnter() {
1219     SplashScreen.hide();
1220   }
1221
1222   ionViewWillLeave() {
1223     SplashScreen.show();
1224   }
1225
1226   ionViewDidEnter() {
1227     SplashScreen.hide();
1228   }
1229
1230   ionViewWillLeave() {
1231     SplashScreen.show();
1232   }
1233
1234   ionViewDidEnter() {
1235     SplashScreen.hide();
1236   }
1237
1238   ionViewWillLeave() {
1239     SplashScreen.show();
1240   }
1241
1242   ionViewDidEnter() {
1243     SplashScreen.hide();
1244   }
1245
1246   ionViewWillLeave() {
1247     SplashScreen.show();
1248   }
1249
1250   ionViewDidEnter() {
1251     SplashScreen.hide();
1252   }
1253
1254   ionViewWillLeave() {
1255     SplashScreen.show();
1256   }
1257
1258   ionViewDidEnter() {
1259     SplashScreen.hide();
1260   }
1261
1262   ionViewWillLeave() {
1263     SplashScreen.show();
1264   }
1265
1266   ionViewDidEnter() {
1267     SplashScreen.hide();
1268   }
1269
1270   ionViewWillLeave() {
1271     SplashScreen.show();
1272   }
1273
1274   ionViewDidEnter() {
1275     SplashScreen.hide();
1276   }
1277
1278   ionViewWillLeave() {
1279     SplashScreen.show();
1280   }
1281
1282   ionViewDidEnter() {
1283     SplashScreen.hide();
1284   }
1285
1286   ionViewWillLeave() {
1287     SplashScreen.show();
1288   }
1289
1290   ionViewDidEnter() {
1291     SplashScreen.hide();
1292   }
1293
1294   ionViewWillLeave() {
1295     SplashScreen.show();
1296   }
1297
1298   ionViewDidEnter() {
1299     SplashScreen.hide();
1300   }
1301
1302   ionViewWillLeave() {
1303     SplashScreen.show();
1304   }
1305
1306   ionViewDidEnter() {
1307     SplashScreen.hide();
1308   }
1309
1310   ionViewWillLeave() {
1311     SplashScreen.show();
1312   }
1313
1314   ionViewDidEnter() {
1315     SplashScreen.hide();
1316   }
1317
1318   ionViewWillLeave() {
1319     SplashScreen.show();
1320   }
1321
1322   ionViewDidEnter() {
1323     SplashScreen.hide();
1324   }
1325
1326   ionViewWillLeave() {
1327     SplashScreen.show();
1328   }
1329
1330   ionViewDidEnter() {
1331     SplashScreen.hide();
1332   }
1333
1334   ionViewWillLeave() {
1335     SplashScreen.show();
1336   }
1337
1338   ionViewDidEnter() {
1339     SplashScreen.hide();
1340   }
1341
1342   ionViewWillLeave() {
1343     SplashScreen.show();
1344   }
1345
1346   ionViewDidEnter() {
1347     SplashScreen.hide();
1348   }
1349
1350   ionViewWillLeave() {
1351     SplashScreen.show();
1352   }
1353
1354   ionViewDidEnter() {
1355     SplashScreen.hide();
1356   }
1357
1358   ionViewWillLeave() {
1359     SplashScreen.show();
1360   }
1361
1362   ionViewDidEnter() {
1363     SplashScreen.hide();
1364   }
1365
1366   ionViewWillLeave() {
1367     SplashScreen.show();
1368   }
1369
1370   ionViewDidEnter() {
1371     SplashScreen.hide();
1372   }
1373
1374   ionViewWillLeave() {
1375     SplashScreen.show();
1376   }
1377
1378   ionViewDidEnter() {
1379     SplashScreen.hide();
1380   }
1381
1382   ionViewWillLeave() {
1383     SplashScreen.show();
1384   }
1385
1386   ionViewDidEnter() {
1387     SplashScreen.hide();
1388   }
1389
1390   ionViewWillLeave() {
1391     SplashScreen.show();
1392   }
1393
1394   ionViewDidEnter() {
1395     SplashScreen.hide();
1396   }
1397
1398   ionViewWillLeave() {
1399     SplashScreen.show();
1400   }
1401
1402   ionViewDidEnter() {
1403     SplashScreen.hide();
1404   }
1405
1406   ionViewWillLeave() {
1407     SplashScreen.show();
1408   }
1409
1410   ionViewDidEnter() {
1411     SplashScreen.hide();
1412   }
1413
1414   ionViewWillLeave() {
1415     SplashScreen.show();
1416   }
1417
1418   ionViewDidEnter() {
1419     SplashScreen.hide();
1420   }
1421
1422   ionViewWillLeave() {
1423     SplashScreen.show();
1424   }
1425
1426   ionViewDidEnter() {
1427     SplashScreen.hide();
1428   }
1429
1430   ionViewWillLeave() {
1431     SplashScreen.show();
1432   }
1433
1434   ionViewDidEnter() {
1435     SplashScreen.hide();
1436   }
1437
1438   ionViewWillLeave() {
1439     SplashScreen.show();
1440   }
1441
1442   ionViewDidEnter() {
1443     SplashScreen.hide();
1444   }
1445
1446   ionViewWillLeave() {
1447     SplashScreen.show();
1448   }
1449
1450   ionViewDidEnter() {
1451     SplashScreen.hide();
1452   }
1453
1454   ionViewWillLeave() {
1455     SplashScreen.show();
1456   }
1457
1458   ionViewDidEnter() {
1459     SplashScreen.hide();
1460   }
1461
1462   ionViewWillLeave() {
1463     SplashScreen.show();
1464   }
1465
1466   ionViewDidEnter() {
1467     SplashScreen.hide();
1468   }
1469
1470   ionViewWillLeave() {
1471     SplashScreen.show();
1472   }
1473
1474   ionViewDidEnter() {
1475     SplashScreen.hide();
1476   }
1477
1478   ionViewWillLeave() {
1479     SplashScreen.show();
1480   }
1481
1482   ionViewDidEnter() {
1483     SplashScreen.hide();
1484   }
1485
1486   ionViewWillLeave() {
1487     SplashScreen.show();
1488   }
1489
1490   ionViewDidEnter() {
1491     SplashScreen.hide();
1492   }
1493
1494   ionViewWillLeave() {
1495     SplashScreen.show();
1496   }
1497
1498   ionViewDidEnter() {
1499     SplashScreen.hide();
1500   }
1501
1502   ionViewWillLeave() {
1503     SplashScreen.show();
1504   }
1505
1506   ionViewDidEnter() {
1507     SplashScreen.hide();
1508   }
1509
1510   ionViewWillLeave() {
1511     SplashScreen.show();
1512   }
1513
1514   ionViewDidEnter() {
1515     SplashScreen.hide();
1516   }
1517
1518   ionViewWillLeave() {
1519     SplashScreen.show();
1520   }
1521
1522   ionViewDidEnter() {
1523     SplashScreen.hide();
1524   }
1525
1526   ionViewWillLeave() {
1527     SplashScreen.show();
1528   }
1529
1530   ionViewDidEnter() {
1531     SplashScreen.hide();
1532   }
1533
1534   ionViewWillLeave() {
1535     SplashScreen.show();
1536   }
1537
1538   ionViewDidEnter() {
1539     SplashScreen.hide();
1540   }
1541
1542   ionViewWillLeave() {
1543     SplashScreen.show();
1544   }
1545
1546   ionViewDidEnter() {
1547     SplashScreen.hide();
1548   }
1549
1550   ionViewWillLeave() {
1551     SplashScreen.show();
1552   }
1553
1554   ionViewDidEnter() {
1555     SplashScreen.hide();
1556   }
1557
1558   ionViewWillLeave() {
1559     SplashScreen.show();
1560   }
1561
1562   ionViewDidEnter() {
1563     SplashScreen.hide();
1564   }
1565
1566   ionViewWillLeave() {
1567     SplashScreen.show();
1568   }
1569
1570   ionViewDidEnter() {
1571     SplashScreen.hide();
1572   }
1573
1574   ionViewWillLeave() {
1575     SplashScreen.show();
1576   }
1577
1578   ionViewDidEnter() {
1579     SplashScreen.hide();
1580   }
1581
1582   ionViewWillLeave() {
1583     SplashScreen.show();
1584   }
1585
1586   ionViewDidEnter() {
1587     SplashScreen.hide();
1588   }
1589
1590   ionViewWillLeave() {
1591     SplashScreen.show();
1592   }
1593
1594   ionViewDidEnter() {
1595     SplashScreen.hide();
1596   }
1597
1598   ionViewWillLeave() {
1599     SplashScreen.show();
1600   }
1601
1602   ionViewDidEnter() {
1603     SplashScreen.hide();
1604   }
1605
1606   ionViewWillLeave() {
1607     SplashScreen.show();
1608   }
1609
1610   ionViewDidEnter() {
1611     SplashScreen.hide();
1612   }
1613
1614   ionViewWillLeave() {
1615     SplashScreen.show();
1616  
```

```

60     },
61     error => {
62       // Timeout or no connection
63       this.showToast('Failed to refresh information');
64       refresher.complete();
65     });
66   }
67   setTimeout(() => {
68     refresher.target.complete();
69   }, 2000);
70 }
71
72 async showToast(message: string, duration: number=3000) {
73   let toast = await this.toastController.create({
74     message: message,
75     duration: duration
76   });
77   toast.present();
78 }
79
80 formatDatetime(sqlDatetime: string) {
81   if (sqlDatetime === null) {
82     return null;
83   }
84   var time=sqlDatetime.substring(16,4)
85   var date = new Date(sqlDatetime.substring(0, 10));
86   var dayNames = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"];
87   var dayOfWeek = date.getDay();
88   return dayNames[dayOfWeek] + ', ' + time.substring(7);
89 }
90
91 // ionic 6 : use capacitor in app browser
92 launch(newsUrl: string) {
93   Browser.open({ url: newsUrl });
94 }
95
96 // ionic 6 : use ionic angular router for change page
97 onAnnouncementClick() {
98   this.router.navigate(['announcement']);
99 }
100

```

Kode A.5: home.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Home</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
12     <ion-refresher-content></ion-refresher-content>
13   </ion-refresher>
14   <ion-card (click)="onAnnouncementClick()">
15     <ion-grid>
16       <ion-row>
17         <ion-col size="9">
18           <ion-card-header>

```

```

19         <ion-card-title>Latest Announcement</ion-card-title>
20     </ion-card-header>
21
22     <ion-card-content>
23         <h3>{{ formatDatetime(wsdcData?.announcements[0].localtime) }}</h3>
24         <p>{{ wsdcData?.announcements[0].message }}</p>
25     </ion-card-content>
26 </ion-col>
27 <ion-col size="3">
28     
29 </ion-col>
30 </ion-row>
31 </ion-grid>
32 </ion-card>
33
34 <ion-list>
35     <ion-list-header>
36         <ion-icon name="book-outline"></ion-icon>
37         <ion-label class="label_newsletters"><h1><b> Newsletters</b></h1></ion-label>
38     </ion-list-header>
39     <ion-item *ngFor="let wsdcNews of wsdcData?.newsletters;">
40         <div class="newsletters" >
41             
42             <h2 text-wrap>{{wsdcNews.title}}</h2> <br>
43             <ion-button class="ion-padding-end" full block color="danger" (click)="launch(wsdcNews.url)">Read More</ion-button>
44         </div>
45     </ion-item>
46 </ion-list>
47 </ion-content>

```

Kode A.6: home.page.html

## A.4 Komponen Info

```

1 import { Component, ViewEncapsulation } from '@angular/core';
2 import { Storage } from '@ionic/storage';
3
4 @Component({
5   selector: 'app-info',
6   templateUrl: './info.page.html',
7   styleUrls: ['./info.page.scss'],
8   encapsulation: ViewEncapsulation.None,
9 })
10
11 export class InfoPage{
12   wsdcInfoData: any;
13
14   constructor(private storage: Storage) {
15
16     this.storage.get('wsdcDataStorage').then((data) => {
17       this.wsdcInfoData = data.info;
18       this.wsdcInfoData = this.wsdcInfoData.replace(new RegExp('icon-telephone', 'g'), '');
19     });
20   }
21 }
22

```

Kode A.7: info.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Info</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <ion-grid>
12     <ion-row>
13       <div [innerHTML]="wsdcInfoData"></div>
14     </ion-row>
15   </ion-grid>
16 </ion-content>
```

Kode A.8: info.page.html

## A.5 Komponen Result

```

1 import { Component, ElementRef, OnInit, ViewChild } from '@angular/core';
2 import { Storage } from '@ionic/storage';
3 import { LoadingController } from '@ionic/angular';
4
5 @Component({
6   selector: 'app-result',
7   templateUrl: './result.page.html',
8   styleUrls: ['./result.page.scss'],
9 })
10
11 export class ResultPage implements OnInit {
12   @ViewChild('resultIFrame') resultIFrame: ElementRef;
13
14   constructor(private storage: Storage, public loadingController: LoadingController) { }
15
16   ngOnInit() {
17     this.storage.get('wsdcDataStorage').then((data) => {
18       this.resultIFrame.nativeElement.contentWindow.location.assign(data.results);
19     });
20     this.presentLoading();
21   }
22
23   async presentLoading() {
24     const loading = await this.loadingController.create({
25       message: 'Please wait...',
26       backdropDismiss: true
27     });
28
29     await loading.present();
30
31     setTimeout(() => {
32       loading.dismiss();
33     }, 500);
34   }
35 }
```

```

36     onResultIframeLoad(){
37         let doc = this.resultIFrame.nativeElement.contentWindow.document;
38         let elements = [
39             doc.getElementById('header'),
40             doc.getElementById('page_header'),
41             doc.getElementById('footer')
42         ];
43         elements.forEach(function (element) {
44             if (element) {
45                 element.style.display = 'none';
46             }
47         })
48         this.resultIFrame.nativeElement.style.display = 'block';
49     }
50 }
51 }
```

Kode A.9: result.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Result</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <iframe #resultIFrame (load)="onResultIframeLoad()"></iframe>
12 </ion-content>
```

Kode A.10: result.page.html

## A.6 Komponen Schedule

```

1 import { Component, ElementRef } from '@angular/core';
2 import { ViewChild } from '@angular/core';
3 import { IonSlides } from '@ionic/angular';
4 import { Storage } from '@ionic/storage';
5
6 @Component({
7   selector: 'app-schedule',
8   templateUrl: './schedule.page.html',
9   styleUrls: ['./schedule.page.scss'],
10 })
11 export class SchedulePage implements OnInit{
12   @ViewChild('scheduleSlider', { static: false }) slider: IonSlides;
13   @ViewChild('segmentContainer', { static: false }) segmentContainer: ElementRef;
14   schedules: any;
15   slideOpts = {
16     initialSlide: 0,
17     speed: 400,
18   };
19   selectedSegmentIdx: any;
20   currentIndex: any;
21
22   constructor(private storage: Storage) {
23
24 }
```

```

26  ngOnInit() {
27    this.storage.get('wsdcDataStorage').then((val) => {
28      this.schedules = val.schedules;
29      this.selectedSegmentIdx = 0;
30    });
31  }
32
33  getDayName(sqlDate: string) {
34    var date = new Date(sqlDate);
35    var dayNames = ["Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"];
36    var dayOfWeek = date.getDay();
37    return dayNames[dayOfWeek];
38  }
39
40  getDate(sqlDate: string) {
41    var date = new Date(sqlDate);
42    return date.getDate();
43  }
44
45  onSlideChanged() {
46    this.slider.getActiveIndex().then((index: number) => {
47      this.currentIndex = index;
48      document.getElementById(this.currentIndex).scrollIntoView({
49        behavior: 'smooth',
50        block: 'center',
51        inline: 'center'
52      });
53      this.selectedSegmentIdx = index;
54    });
55  }
56
57  onSegmentChanged(segmentButton) {
58    this.slider.slideTo(segmentButton.detail.value);
59  }
60}

```

Kode A.11: schedule.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Schedule</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <div id="schedulesContainer">
12     <div id="schedulesSegments" slot="fixed">
13       <ion-segment #segmentContainer scrollable [(ngModel)]="selectedSegmentIdx"
14       (ionChange)="onSegmentChanged($event)">
15         <ion-segment-button *ngFor="let schedule of schedules; let i = index;" [
16           value]="i" [id]="#i">
17           <ion-label>
18             <div class="day">{{ getDayName(schedule.date) }}</div>
19             <div class="date">{{ getDate(schedule.date) }}</div>
20           </ion-label>
21         </ion-segment-button>
22       </ion-segment>
23     </div>
24     <div id="schedulesSlides">
25       <ion-slides #scheduleSlider [options]="slideOpts" (ionSlideDidChange)=""
26

```

```

onSlideChanged()">
    <ion-slide *ngFor="let schedule of schedules;">
        <ion-list>
            <ion-item *ngFor="let agenda of schedule.agenda;" >
                <ion-note item-start>
                    {{agenda.start}} <br>
                    {{agenda.end}}
                </ion-note>
                <ion-label class="ion-text-wrap">
                    <h3>{{agenda.title}}</h3>
                    <p>{{agenda.subtitle}}</p>
                </ion-label>
            </ion-item>
        </ion-list>
    </ion-slide>
</ion-slides>
</div>
</div>
</ion-content>

```

Kode A.12: schedule.page.html

## A.7 Komponen Venues

```

1 import { Component, OnInit } from '@angular/core';
2 import { Router, NavigationExtras } from '@angular/router';
3 import { Storage } from '@ionic/storage';
4
5 @Component({
6   selector: 'app-venues',
7   templateUrl: './venues.page.html',
8   styleUrls: ['./venues.page.scss'],
9 })
10 export class VenuesPage implements OnInit{
11   venuesData: Array<{ id: string, name: string, icon: string, geojson: any,
12     colorIdx: number }>;
13   valVenues: any;
14   constructor(private router: Router,private storage: Storage) {
15
16
17   ngOnInit(){
18     this.storage.get('wsdcDataStorage').then((val) => {
19       this.valVenues = val.venues;
20       this.venuesData = [];
21       let currColorIdx: number = 1;
22       for (let venue of this.valVenues) {
23         this.venuesData.push({
24           id: venue.id,
25           name: venue.name,
26           icon: venue.icon,
27           geojson: venue.geojson,
28           colorIdx: currColorIdx ,
29         });
30         if (currColorIdx > 4) {
31           currColorIdx = 1;
32         } else {
33           currColorIdx++;
34         }
35       }
36     })
37   }
38 }

```

```

37     }
38
39     itemTapped(wsdcVenue) {
40       // this.router.navigate(['venues-map', id])
41       let navigationExtras: NavigationExtras = {
42         state: {
43           venuesData: wsdcVenue
44         }
45       };
46       this.router.navigate(['venues-map'], navigationExtras);
47     }
48   }

```

Kode A.13: venues.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Venues</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <ion-grid>
12     <ion-row>
13       <ion-list no-lines>
14         <ion-button color="white" id="{{wsdcVenue.id}}" *ngFor="let wsdcVenue of
15           venuesData" (click)="itemTapped(wsdcVenue)">
16           <ion-icon name="{{wsdcVenue.icon}}></ion-icon>
17           <span>{{ wsdcVenue.name }}</span>
18         </ion-button>
19       </ion-list>
20     </ion-row>
21   </ion-grid>
22 </ion-content>

```

Kode A.14: venues.page.html

## A.8 Komponen Venues Map

```

1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute, Router } from '@angular/router';
3 import { Storage } from '@ionic/storage';
4 import { CapacitorGoogleMaps } from "@capacitor-community/google-maps";
5 import { Geolocation } from '@capacitor/geolocation';
6
7 @Component({
8   selector: 'app-venues-map',
9   templateUrl: './venues-map.page.html',
10  styleUrls: ['./venues-map.page.scss'],
11 })
12 export class VenuesMapPage implements OnInit{
13   venuesPage: any;
14   venuesMapsDetail: any;
15   userCoordinatesLat: any;
16   userCoordinatesLng: any;
17   mapid: any;
18   userDistanceTo: string[];
19   itemCounter: number;

```

```
20 items: Array<{id: string, idcolor:number }>
21 pageTitleColors: Array<string> = ["#ec1c24", "#c49a6c", "#d189bb", "#4d113f"];
22
23 constructor(private activatedRoute: ActivatedRoute, private router: Router,
24   private storage: Storage) {
25
26   //Get data from venues page.
27   this.items = [];
28   this.activatedRoute.queryParams.subscribe(params => {
29     if (this.router.getCurrentNavigation().extras.state) {
30       this.venuesPage = this.router.getCurrentNavigation().extras.state.
31       venuesData;
32     }
33
34     this.items.push({
35       id: this.venuesPage.id,
36       idcolor: this.venuesPage.colorIdx
37     });
38
39     document.getElementById("pagetitle").style.color = this.pageTitleColors[this.
40     items[0].idcolor-1];
41   });
42
43   //save user lat & lng location
44   const printCurrentPosition = async () => {
45     const coordinates = await Geolocation.getCurrentPosition();
46     this.userCoordinatesLat = coordinates.coords.latitude;
47     this.userCoordinatesLng = coordinates.coords.longitude;
48   };
49
50   printCurrentPosition();
51   Geolocation.requestPermissions();
52 }
53
54 ngOnInit(){
55   // Select data from storage
56   this.storage.get('wsdcDataStorage').then((data) => {
57     this.venuesMapsDetail = data.venues;
58     this.venuesMapsDetail = this.venuesMapsDetail.filter(d=> d.id==this.items
59     [0].id);
60   })
61 }
62
63 ionViewDidEnter() {
64   // create map and add marker
65   const initializeMap = async () => {
66     this.itemCounter = 0;
67     await CapacitorGoogleMaps.initialize({
68       key: "YOUR_IOS_MAPS_API_KEY",
69       devicePixelRatio: window.devicePixelRatio,
70     });
71     const element = document.getElementById("mapContainer");
72     const boundingRect = element.getBoundingClientRect();
73     try {
74       const result = await CapacitorGoogleMaps.createMap({
75         boundingRect: {
76           width: Math.round(boundingRect.width),
77           height: Math.round(boundingRect.height),
78           x: Math.round(boundingRect.x),
79           y: Math.round(boundingRect.y),
80         },
81         cameraPosition:{
82           target:{ //Kuta, Bali -8.722396, 115.17671
```

```

79         latitude:-8.722396,
80         longitude: 115.17671
81     },
82     zoom:11
83   },
84   preferences:{
85     controls:{
86       isCompassButtonEnabled:true ,
87       isMyLocationButtonEnabled:true ,
88       isZoomButtonsEnabled:true
89     },
90     appearance:{
91       isMyLocationDotShown:true
92     }
93   },
94 });
95
96 element.style.background = "";
97 element.setAttribute("data-maps-id", result.googleMap.mapId);
98 this.mapid = result.googleMap.mapId;
99 console.log(this.venuesMapsDetail);
100
101 //add marker to each location
102 for(let venue of this.venuesMapsDetail){
103   for(let venuesMarker of venue.geojson.features){
104     this.itemCounter +=1;
105     const koor = venuesMarker.geometry.coordinates;
106     CapacitorGoogleMaps.addMarker({
107       mapId:result.googleMap.mapId,
108       position:{
109         latitude: koor[1],
110         longitude: koor[0],
111       },
112       preferences:{
113         title: venuesMarker.properties.Name
114       },
115     });
116   }
117 }
118 } catch (e) {
119   alert("Map failed to load");
120 }
121
122 // Insert distance to each location
123 this.userDistanceTo = new Array(this.itemCounter);
124 let counter = 0;
125 for(let venue of this.venuesMapsDetail){
126   for(let venuesMarker of venue.geojson.features){
127     const koor = venuesMarker.geometry.coordinates;
128     this.userDistanceTo[counter] = this.computeDistance(this.
129 userCoordinatesLat,koor[1],this.userCoordinatesLng,koor[0]);
130     counter+=1;
131   }
132 };
133
134 (function () {
135   initializeMap();
136 })();
137 }
138
139 backToVenue() {
140   this.router.navigate(['venues']);

```

```

141 }
142
143     featTapped(venuesCoordinates){
144         const coordinates = venuesCoordinates;
145         CapacitorGoogleMaps.moveCamera({
146             mapId:this.mapid,
147             cameraPosition:{
148                 target:{
149                     latitude: coordinates[1],
150                     longitude: coordinates[0],
151                 },
152                 zoom:15
153             },
154             duration:800
155         });
156     }
157
158     computeDistance(lat1, lat2, lon1, lon2){
159
160         const R = 6371e3; // metres
161         const phi1 = lat1 * Math.PI/180; // phi, lambda in radians
162         const phi2 = lat2 * Math.PI/180;
163         const deltaPhi = (lat2-lat1) * Math.PI/180;
164         const deltaLambda = (lon2-lon1) * Math.PI/180;
165
166         const a = Math.sin(deltaPhi/2) * Math.sin(deltaPhi/2) +
167             Math.cos(phi1) * Math.cos(phi2) *
168             Math.sin(deltaLambda/2) * Math.sin(deltaLambda/2);
169         const c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
170
171         const d = R * c; // in metres
172
173
174         if(d < 1000){
175             return Math.floor(d) + " m";
176         }else if (d < 100000){
177             return Math.floor(d/1000) + " km";
178         }else{
179             return ">99 km"
180         }
181     }
182 }
```

Kode A.15: venues-map.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start" (click)="backToVenue()">
4       <ion-icon name="arrow-back-outline"></ion-icon>
5     </ion-buttons>
6     <ion-title>Venues</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <div id="container">
11   <div id="mapContainer"></div>
12 </div>
13
14 <!-- /deprecated scroll in ionic 5 -->
15 <ion-content scrollX="true">
16   <ion-label>
17     <h3 #pagetitle id="pagetitle">{{venuesPage.name}}</h3>
18   </ion-label>
```

```
19 <ion-list>
20   <ion-item *ngFor="let venue of venuesMapsDetail; let i=index">
21     <div>
22       <div class="venuesDescContainer" *ngFor="let properties of venue.geojson.
23 features; let j = index">
24         <div class="venueDesc" (click)="featTapped(properties.geometry.
25 coordinates)">
26           <h2>{{properties.properties.Name}}</h2>
27           <p>{{properties.properties.Description}}</p>
28         </div>
29         <div class="venuesDist" #distance>
30           <p>{{userDistanceTo[j]}}</p>
31         </div>
32       </div>
33     </ion-item>
34   </ion-list>
35 </ion-content>
```

Kode A.16: venues-map.page.html