

SKRIPSI

PEMBUATAN ULANG APLIKASI WSDC 2017 BALI DENGAN IONIC 5



Rajasa Cikal Maulana Solihin

NPM: 2017730084

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2021

DAFTAR ISI

DAFTAR ISI	iii
DAFTAR GAMBAR	v
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi	3
1.6 Sistematika Pembahasan	3
2 LANDASAN TEORI	5
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	14
2.3.3 Migrasi Ionic 3 ke Ionic 6	21
3 ANALISIS	31
3.1 Analisis Sistem Kini	31
3.1.1 Skenario Pengguna	32
3.1.2 Struktur Ionic 3	34
3.2 Analisis Sistem Usulan	55
3.2.1 Analisis Kebutuhan	55
3.2.2 Permasalahan Pengembangan Sistem Usulan	62
4 PERANCANGAN	63
4.1 Perancangan Kelas	63
4.2 Perancangan Struktur HTML	72
5 IMPLEMENTASI DAN PENGUJIAN	75
5.1 Implementasi	75
5.1.1 Lingkugan Implementasi	75
5.1.2 Hasil Implementasi	75
5.2 Pengujian	81
5.2.1 Pengujian Fungsional	81
5.2.2 Pengujian Eksperimental	83
6 KESIMPULAN DAN SARAN	85
6.1 Kesimpulan	85
6.2 Saran	85

DAFTAR REFERENSI	87
A KODE PROGRAM	89
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	Wireframe Aplikasi WSDC 2017 Bali	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	45
4.1	Diagram Kelas Keseluruhan	64
5.1	Tangkapan Layar Aplikasi WSDC 2017 Bali	77
5.2	Tangkapan Layar Aplikasi WSDC 2017 Bali	78
5.3	Tangkapan Layar Aplikasi WSDC 2017 Bali	79
5.4	Tangkapan Layar Aplikasi WSDC 2017 Bali	80
5.5	Tangkapan Layar Aplikasi WSDC 2017 Bali	81

¹

BAB 1

²

PENDAHULUAN

³ 1.1 Latar Belakang

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

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

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

²⁶ Pada Ionic 5 keatas, terdapat beberapa kerangka Javascript yang dapat diimplementasikan
²⁷ menggunakan *framework* Ionic, seperti Angular, React, dan Vue. Angular pada awalnya diciptakan
²⁸ oleh karyawan Google, Misko Hevert dan Adam Abrons pada tahun 2008, yang masih bernama
²⁹ AngularJS dan dikembangkan dalam JavaScript [3]. Pada saat itu sebagian besar situs web
³⁰ menggunakan aplikasi multi-halaman, yaitu ketika pengguna mengklik tautan, maka browser harus
³¹ mengambil dokumen HTML yang diminta dari server. Namun dengan Angular, digunkannya
³² *Single-page Application* (SPA), yaitu ketika halaman awal dimuat, semua yang dibutuhkan untuk
³³ membuat dan menampilkan sebuah halaman diunduh, kemudian ditampilkan kedalam layar. Dengan
³⁴ begitu, browser tidak perlu melakukan *refresh* terhadap halaman tersebut [4]. React adalah *library*

1 JavaScript *open source* untuk membangun antarmuka pengguna, dikelola oleh Facebook, dapat
2 digunakan dalam berbagai skenario termasuk aplikasi iOS dan Android [3]. Sedangkan Vue
3 merupakan *framework* progresif untuk membangun antarmuka pengguna untuk web, yang dapat
4 digunakan baik untuk projek kecil dan untuk *Single-Page Applications* (SPAs) [3].

5 *World Schools Debating Championships* (WSDC) merupakan sebuah turnamen debat Bahasa
6 Inggris tahunan untuk tim-tim tingkat sekolah menengah yang mewakili berbagai negara [5]. Pada
7 awalnya, kompetisi universitas dunia akan diselenggarakan di Sydney pada bulan Juli 1988. Anggota
8 Federasi Debat Australia menyadari bahwa tidak ada acara serupa untuk siswa sekolah menengah.
9 Namun kejuaraan universitas dunia ini menunjukkan potensi yang sangat besar untuk kompetisi
10 debat internasional yang melibatkan siswa dari seluruh dunia. Pada tahun 1991, kejuaraan diadakan
11 di Edinburgh. Dan sejak saat itu nama World Schools Debating Championships digunakan dan
12 berlangsung hingga saat ini.

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

19 Aplikasi WSDC 2017 Bali yang dibangun pada tahun 2017 oleh PT DNArtworks Komunikasi
20 Visual menggunakan Ionic versi 3. Sedangkan Ionic versi 3 pada saat skripsi ini ditulis sudah tidak
21 mendapat pembaruan lagi. Sedangkan Ionic Framework semakin berkembang dan pada saat skripsi
22 ini dibuat sudah mencapai Ionic versi 6¹. Maka dari itu, pada skripsi ini akan dibuat sebuah
23 aplikasi pembaruan dari aplikasi WSDC 2017 Bali saat ini, dengan menggunakan *framework* Ionic
24 versi 6². *Framework* yang lebih baru memungkinkan perawatan yang lebih efisien, serta dukungan
25 teknologi yang lebih terbarukan.

26 **1.2 Rumusan Masalah**

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

29 **1.3 Tujuan**

30 Tujuan yang ingin dicapai dari penulisan skripsi ini adalah melakukan migrasi aplikasi Android
31 WSDC 2017 ke *framework* Ionic versi 6.

32 **1.4 Batasan Masalah**

33 Dalam skripsi ini dibuat batasan-batasan masalah dalam pembuatan perangkat lunak. Batasan-
34 batasan masalah yang ditetapkan adalah sebagai berikut:

¹Saat 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.

²Pembaruan 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 penguji.

- 1 1. Aplikasi ini tidak akan memiliki fitur notifikasi, karena acara WSDC 2017 Bali sudah selesai
2 dan tidak diperlukan kembali fitur notifikasi.
- 3 2. Aplikasi hanya akan diuji pada *platform mobile* berbasis android.

4 1.5 Metodologi

- 5 Langkah-langkah yang dilakukan dalam skripsi ini adalah sebagai berikut:
 - 6 1. Melakukan studi mengenai *framework* Ionic versi 3 dan versi 6.
 - 7 2. Menganalisis aplikasi WSDC 2017 Bali.
 - 8 3. Mempelajari bagaimana cara melakukan migrasi Ionic versi 3 ke versi 6.
 - 9 4. Mendesain kelas aplikasi.
 - 10 5. Membangun aplikasi WSDC dengan *framework* Ionic versi 6.
 - 11 6. Melakukan pengujian dan eksperimen.
 - 12 7. Menulis dokumen skripsi.

13 1.6 Sistematika Pembahasan

- 14 Sistematika penulisan setiap bab pada skripsi ini adalah sebagai berikut:
 - 15 1. Bab Pendahuluan

Bab 1 berisi latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, dan sistematika pembahasan yang digunakan untuk menyusun skripsi ini.
 - 16 2. Bab Dasar Teori

Bab 2 berisi teori-teori yang digunakan dalam pembuatan skripsi ini. Teori-teori tersebut yaitu WSDC, Angular, Ionic Framework, Capacitor, Cordova, UI Components, dan Migrasi Ionic.
 - 17 3. Bab Analisis

Bab 3 berisi analisis yang dilakukan pada skripsi ini, meliputi analisis sistem kini, analisis kebutuhan aplikasi WSDC 2017 Bali yang akan dibangun, serta permasalahan pembangunan sistem usulan.
 - 18 4. Bab Perancangan

Bab 4 berisi perancangan aplikasi meliputi perancangan kelas beserta dengan diagram kelas, deskripsi kelas dan fungsinya, serta perancangan struktur HTML.
 - 19 5. Bab Implementasi dan Pengujian

Bab 5 berisi implementasi dan pengujian aplikasi meliputi lingkungan implementasi, hasil implementasi, pengujian fungsional, dan pengujian eksperimental.
 - 20 6. Bab Kesimpulan dan Saran

Bab 6 berisi kesimpulan dari hasil pembangunan aplikasi ini dan saran untuk pengembangan 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 5. 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

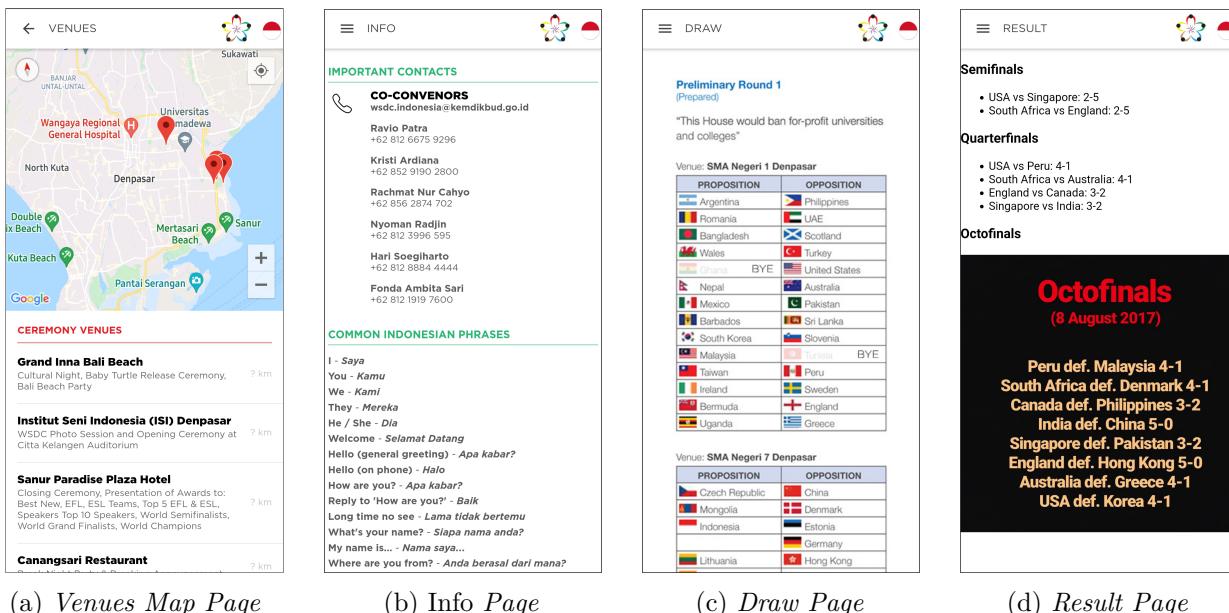
- 7 Aplikasi WSDC 2017 Bali digunakan untuk menunjang keberlangsungan acara WSDC 2017 yang dise-
8 lenggarakan di Bali, Indonesia. Aplikasi WSDC 2017 Bali dapat diunduh untuk sistem operasi *andro-*
9 *id* melalui URL <https://play.google.com/store/apps/details?id=org.wsdc2017indonesia.app>. Aplikasi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

The wireframe displays four pages of the WSDC 2017 Bali mobile application:

- (a) Home page:** Features a "Latest Announcement" section with a live streaming link for the Closing Ceremony. It also includes a "Newsletters" section with two issues of the "WSDC 2017 newsletter".
- (b) Announcements Page:** Lists various events and updates, such as the International Arrival at Ngurah Rai Airport, Team Registration at Sanur Paradise Plaza Hotel, and a dinner at Griya Agung Ballroom.
- (c) Schedule Page:** A calendar view showing events from August 1st to August 7th. Key events include the International Arrival on Tuesday, Team Registration on Wednesday, Hotel Check-In on Thursday, and a dinner on Friday.
- (d) Venues Page:** Lists ceremony venues, competition venues, delegates accommodation, and educational tour details.

Gambar 2.1: Wireframe Aplikasi WSDC 2017 Bali

- 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 [6]. 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 [4]. 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 [6]. 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*. Lalu
 11 selain itu, TypeScript juga dapat dimungkinkan untuk mendeklarasikan jenis tipe khusus yang dapat
 12 dikustomisasi. Maka 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 [7]. Selain itu di dalam komponen terdapat
 17 *module.ts* yang merupakan NgModule dari sebuah komponen, *spec.ts* yang digunakan untuk menguji
 18 komponen, dan *routing.module.ts* yang berisi definisi untuk bernavigasi antar bagian dalam aplikasi
 19 Angular. 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.

```
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.

```
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   declaration: [
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 Lalu, pada tahun 2019, Ionic mendukung penggunaan *framework* lain selain Angular, yaitu React
 29 dan 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 peforma dari aplikasi tersebut [8].

39 Ionic mendukung komunikasi dengan menggunakan Native API yang terintegrasi untuk menam-
 40 bakan fungsionalitas ke dalam aplikasi Ionic apapun dengan menggunakan Capacitor atau Cordova.

- 1 Dengan terpasangnya Ionic Native, maka aplikasi akan memiliki antar muka yang diperlukan untuk
 2 berinteraksi dengan salah satu *plug-in*, yaitu Capacitor atau Cordova.

3 **2.3.1.1 Capacitor**

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

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

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

18 1. *Official Plugins*

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

22 (a) Browser API

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

26
 27 1 `npm install @capacitor/browser`
 28 2 `npx cap sync`

Kode 2.6: Kode untuk Menginstal Browser API

30 Browser API memiliki beberapa *method*, yaitu:

- 31 • `open()`: *method* ini digunakan untuk membuka halaman *browser* dengan opsi yang
 32 sudah ditentukan.
- 33 • `close()`: *method* ini digunakan untuk menutup halaman *browser* yang sedang dibuka.
 34 *Method* ini hanya dapat digunakan pada web dan iOS.
- 35 • `addListener('browserFinished', ...)`: *method* ini merupakan sebuah *listener* untuk
 36 *browser finished event* dan dipanggil ketika *browser* ditutup oleh pengguna. *Method*
 37 ini hanya dapat digunakan pada Android dan iOS.
- 38 • `addListener('browserPageLoaded', ...)`: *method* ini merupakan sebuah *listener* untuk
 39 *page loaded event* dan aktif ketika URL diteruskan ke *method finished loading*. *Method*
 40 ini hanya dapat digunakan pada Android dan iOS.

- 1 • `removeAllListeners()`: *method* ini digunakan untuk menghapus semua *native listeners*
 2 untuk *plugin* ini.

3 (b) Geolocation API

4 Geolocation API menyediakan metode untuk mendapatkan dan melacak posisi perangkat
 5 saat ini menggunakan *Global Positioning System* (GPS), dengan latitude dan longitude,
 6 juga informasi mengenai ketinggian, arah, dan kecepatan jika tersedia. Untuk menginstal
 7 Geolocation API dapat dilakukan melalui *command line* dengan perintah seperti pada
 8 kode 2.7.

```
9
10 1 npm install @capacitor/geolocation
11 2 npx cap sync
```

Kode 2.7: Kode untuk Menginstal Geolocation API

13 Pada perangkat Android, dibutuhkan izin untuk menggunakan Geolocation API 2.8. Izin
 14 tersebut ditambahkan ke dalam baris kode pada *file* `AndroidManifest.xml`. Izin tersebut
 15 digunakan untuk meminta data lokasi (*fine* dan *coarse*), dan izin untuk menggunakan
 16 GPS. Dengan menggunakan GPS, Geolocation API akan mengembalikan koordinat dari
 17 perangkat yang berupa latitude dan longitude. Contoh dari penggunaan Geolocation
 18 API untuk mengambil koordinat perangkat tertera pada kode 2.9.

```
19
20 1 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"
21   />
22 2 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
23   />
24 3 <uses-feature android:name="android.hardware.location.gps" />
```

Kode 2.8: Permissions Geolocation API pada Android

```
26
27 1 import { Geolocation } from '@capacitor/geolocation';
28
29 2 const printCurrentPosition = async () => {
30   3   const coordinates = await Geolocation.getCurrentPosition();
31
32   4   console.log('Current position:', coordinates);
33 5 }
34 6
```

Kode 2.9: Permissions Geolocation API pada Android

35 Untuk mengambil posisi dari perangkat dengan menggunakan method `getCurrentPosition()` seperti pada kode 2.9. Selain itu terdapat beberapa method lain yang dapat
 36 diimplementasikan, diantaranya yaitu:

37 • `watchPosition()`: digunakan untuk mendaftarkan fungsi handler yang akan dipanggil
 38 secara otomatis setiap kali posisi perangkat berubah.

39 • `clearWatch()`: digunakan untuk membatalkan pendaftaran fungsi handler yang
 40 sebelumnya diinstal menggunakan `watchPosition()`.

41 • `checkPermissions()`: digunakan untuk mengecek izin penggunaan lokasi.

42 • `requestPermissions()`: digunakan untuk meminta izin penggunaan lokasi.

1 (c) Splash Screen API

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

```
6
7   1 npm install @capacitor/splash-screen
8   2 npx cap sync
```

Kode 2.10: Kode untuk Menginstal Splash Screen API

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

Kode 2.11: Contoh Kode Penggunaan Splash Screen API

27 Secara *default*, Splash Screen diatur secara otomatis untuk disembunyikan dalam waktu
 28 500ms. Selain itu, pada kondisi tertentu Splash Screen tidak sepenuhnya menutupi layar,
 29 seperti pada bagian-bagian sudut-sudut layar. Splash Screen dapat mengatur warna
 30 latar belakang, dibandingkan dengan menampilkan warna transparan saat Splash Screen
 31 tidak sepenuhnya menutupi layar. Selain itu, Splash Screen juga dapat dibuat untuk
 32 menampilkan *spinner* diatas Splash Screen.

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

37 2. *Community Plugins*

38 *Community Plugins* merupakan sekumpulan *plugin* yang diciptakan oleh komunitas untuk
 39 menambahkan fungsionalitas ke dalam aplikasi. Perbedaan dengan *Official Plugins* adalah
 40 dimana tim Capacitor tidak secara resmi melakukannya pemeliharaan terhadap *Community*
 41 *Plugins*. Salah satu *plugin* yang diciptakan oleh komunitas adalah *plugin* Google Maps
 42 yang menggunakan *native* Google Maps SDK. Untuk menginstal *plugin* Google Maps dapat
 43 dilakukan melalui *command line* dengan perintah seperti pada kode 2.12.

```

1 1 npm i --save @capacitor-community/google-maps
2 2 npx cap sync

```

Kode 2.12: Kode untuk Menginstal *Plugin* Google Maps

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

```

10
11 const initializeMap = async () => {
12   await CapacitorGoogleMaps.initialize({
13     key: "YOUR_IOS_MAPS_API_KEY",
14     devicePixelRatio: window.devicePixelRatio, // this line is very important
15   });
16
17   const element = document.getElementById("container");
18   const boundingRect = element.getBoundingClientRect();
19   try {
20     const result = await CapacitorGoogleMaps.createMap({
21       boundingRect: {
22         width: Math.round(boundingRect.width),
23         height: Math.round(boundingRect.height),
24         x: Math.round(boundingRect.x),
25         y: Math.round(boundingRect.y),
26       },
27     });
28     element.style.background = "";
29     element.setAttribute("data-maps-id", result.googleMap.mapId);
30
31     alert("Map loaded successfully");
32   } catch (e) {
33     alert("Map failed to load");
34   }
35 };
36
37 (function () {
38   initializeMap();
39 })();

```

Kode 2.13: Contoh Kode Penggunaan *Plugin* Google Maps

41 2.3.1.2 Cordova

42 Cordova merupakan *framework open source* yang dapat membuat pengembang untuk menggunakan
43 teknologi seperti HTML, JavaScript, dan CSS untuk membangun aplikasi untuk perangkat bergerak
44 yang dapat berjalan pada beberapa sistem operasi *mobile* [11]. Cordova menyediakan antarmuka
45 antara WebView dan lapisan *native* pada perangkat [8]. Selain dapat bekerja pada dua platform
46 seluler Android dan iOS, Cordova juga dapat digunakan pada platform seluler seperti Windows Pho-
47 ne, Blackberry, dan FireOS.

Untuk mengonfigurasi proyek Cordova, saat ini dapat menggunakan *Command Line Tool* (CLI). CLI membuat proyek dasar dan mengonfigurasinya agar berfungsi dengan platform seluler apa pun yang didukung yang dapat digunakan. Cordova CLI juga dapat membuat pengembang memiliki integrasi dan pengelolaan *plug-in*. Selain itu, CLI juga dapat mengkompilasi aplikasi untuk berjalan pada simulator atau pada perangkat *native*. Serupa dengan Capacitor, Cordova membuat pengembang dapat mengakses fitur *native* dari sebuah perangkat, seperti kamera, papan ketik, dan geolokasi, menggunakan *plugin*. Framework Ionic telah terdapat berbagai macam TypeScript *wrapper* untuk *plugins* Cordova. Untuk dapat menggunakan Cordova Plugins, yaitu dengan memasang Cordova Plugins terlebih dahulu (Kode 2.14), dan memperbaruiya ke versi terakhir (Kode 2.15) yang dapat dilakukan melalui CLI. Setiap *plugins* memiliki dua komponen, yaitu kode *native* (Cordova), dan kode TypeScript (Ionic Native).

```
12
13 npm install cordova-plugin-name
14 npx cap sync
```

Kode 2.14: Kode untuk Memasang Cordova Plugins

```
16
17 npm install cordova-plugin-name@2
18 npx cap update
```

Kode 2.15: Kode untuk Memperbarui Cordova Plugins

Sama seperti Capacitor, Cordova juga memiliki *plugins* yang menyediakan antarmuka JavaScript ke komponen *native*. *Plugin* memungkinkan aplikasi untuk menggunakan kemampuan *native* dari suatu perangkat. Salah satu *plugin* yang tersedia yaitu Google Maps. *Plugin* ini menghasilkan MapView secara *native*, dan menempatkannya di bawah browser. MapView yang dihasilkan bukan merupakan HTMLElements, maka tidak terkait dengan HTML. Sebelum dapat menggunakan *plugin* Google Maps, dilakukan instalasi *plugin* untuk pertama kali pada *command line* 2.16. Selanjutnya atur Google Maps API Key di dalam file config.xml 2.17.

```
27
28 cordova plugin add cordova-plugin-googlemaps
```

Kode 2.16: Kode untuk Menginstal *Plugin* Cordova Google Maps

```
30
31 <widget ...>
32   <preference name="GOOGLE_MAPS_ANDROID_API_KEY" value="(api key)" />
33   <preference name="GOOGLE_MAPS_IOS_API_KEY" value="(api key)" />
34 </widget>
```

Kode 2.17: Kode untuk Mengatur API Key untuk *Plugin* Cordova Google Maps

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 [8]. 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 yaitu :

1 • Action Sheet

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

6 • Alert

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

11 • Badge

12 Merupakan elemen *inline block* yang biasanya muncul di dekat elemen lain, berisi angka atau
13 karakter lain, yang digunakan sebagai pemberitahuan bahwa ada item tambahan yang terkait
14 dengan suatu elemen dan menunjukkan berapa banyak item yang ada. Penggunaan Badge
15 dengan menggunakan tag `<ion-badge>` (Kode 2.18).

16 1 <ion-badge>99</ion-badge>
17 2

Kode 2.18: Potongan Kode Program dari Badge Component

20 • Button

21 Merupakan elemen yang dapat diklik, biasanya digunakan dalam formulir atau di mana
22 pun yang membutuhkan fungsionalitas tombol. Button biasanya menampilkan teks, ikon,
23 atau bisa juga keduanya. Button dapat pula menggunakan atribut untuk menampilkannya
24 dengan penampilan tertentu. Penggunaan Button dengan menggunakan tag `<ion-button>`
25 (Kode 2.19).

26 1 <ion-button>Default</ion-button>
27 2

Kode 2.19: Potongan Kode Program dari Button Component

30 • Card

31 Merupakan bagian standar dari tampilan antarmuka yang berfungsi sebagai titik masuk
32 ke dalam informasi yang lebih detail. Card dapat menjadi satu komponen, tetapi sering
33 kali terdiri dari beberapa header, judul, sub judul, dan konten. Penggunaan Card dengan
34 menggunakan tag `<ion-card>` yang dapat berisi *header*, *subtitle*, *title*, dan *content* (Kode 2.20).

```

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
12
13

```

Kode 2.20: 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 2.21).

```

20 1 <ion-content
21   [scrollEvents]="true"
22   (ionScrollStart)="logScrollStart()"
23   (ionScroll)="logScrolling($event)"
24   (ionScrollEnd)="logScrollEnd()"
25
26   <h1>Main Content</h1>
27
28   <div slot="fixed">
29     <h1>Fixed Content</h1>
30   </div>
31 </ion-content>
32
33

```

Kode 2.21: 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 2.35), dan di antarmuka pemilih yang ditampilkan dari bawah layar.

```

40 1 <ion-datetime displayFormat="MM DD YY" placeholder="Select Date"></ion-
41   datetime>
42
43 2

```

Kode 2.22: Kode Program dari Datetime Component dengan Format Bulan-Hari-Tahun

1 • Grid

2 Grid digunakan untuk membuat tata letak kustom pada tampilan. Grid terdiri dari tiga bagian,
 3 yaitu *grid*, baris, dan kolom. Masing-masing kolom dapat diubah ukurannya menggunakan
 4 CSS. Grid digunakan dengan tag `<ion-grid>` (Kode 2.23).

```
5   1 <ion-row>
6   2   <ion-col size="6">
7   3     ion-col [size="6"]
8   4   </ion-col>
9   5   <ion-col>
10  6     ion-col
11  7   </ion-col>
12  8 </ion-row>
```

Kode 2.23: Potongan Kode Program dari Grid Component

15 • Infinite Scroll

16 Komponen Infinite Scroll memanggil sebuah action yang akan dilakukan ketika pengguna
 17 menggulir dengan jarak tertentu dari bawah atau atas halaman. Penggunaan Infinite Scroll
 18 dengan menggunakan tag `<ion-infinite-scroll>` (Kode 2.24).

```
19  1 <ion-infinite-scroll threshold="100px" (ionInfinite)="loadData($event)">
20  2   <ion-infinite-scroll-content
21  3     loadingSpinner="bubbles"
22  4     loadingText="Loading more data..."
23  5   </ion-infinite-scroll-content>
24  6 </ion-infinite-scroll>
25  7
```

Kode 2.24: Potongan Kode Program dari Infinite Scroll Component

28 • Icon

29 Icon merupakan komponen yang berupa gambar kecil, yang merepresentasikan sebuah berkas,
 30 dan folder di dalam aplikasi. Penggunaan Icon adalah dengan menggunakan tag `<ion-icon>`
 31 (Kode 2.25).

```
32  1 <ion-icon name="home"></ion-icon>
33  2
```

Kode 2.25: Potongan Kode Program dari Icon Home

36 • Item

37 Item merupakan elemen yang dapat berisi teks, ikon, avatar, gambar, masukan, dan elemen
 38 asli atau kustom lainnya. Biasanya, item ditempatkan di dalam sebuah *list* bersamaan dengan
 39 item lainnya dengan tag `<ion-item>` (Kode 2.26). Dapat dilakukan *swipe*, dihapus, disusun
 40 ulang, dan diedit.

```

1 1 <ion-item>
2   <ion-label>
3     Item
4   </ion-label>
5 </ion-item>
6
7
8

```

Kode 2.26: 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 2.27). List mendukung untuk pengguna melakukan *swiping*, *dragging*, dan penghapusan *item*.

```

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

```

Kode 2.27: 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 2.28). Pada dasarnya, Menu muncul dari kiri, tetapi sisi kemunculan menu dapat diganti.

```

1 <ion-menu side="start" menuId="first" contentId="main">
2   <ion-header>
3     <ion-toolbar color="primary">
4       <ion-title>Start Menu</ion-title>
5     </ion-toolbar>
6   </ion-header>
7   <ion-content>
8     <ion-list>
9       <ion-item>Menu Item</ion-item>
10      <ion-item>Menu Item</ion-item>
11      <ion-item>Menu Item</ion-item>
12      <ion-item>Menu Item</ion-item>
13      <ion-item>Menu Item</ion-item>

```

```

1  14      </ion-list>
2  15    </ion-content>
3  16  </ion-menu>
4
5  17

```

Kode 2.28: 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 2.29).

```

12
13 1 <ion-modal [isOpen]="#true">
14 2   <ng-template>
15 3     <ion-content>Modal Content</ion-content>
16 4   </ng-template>
17 5 </ion-modal>
18 6

```

Kode 2.29: 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 ??). 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.

```

32
33 1 <ion-content>
34 2   <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
35 3     <ion-refresher-content></ion-refresher-content>
36 4   </ion-refresher>
37 5 </ion-content>
38 6

```

Kode 2.30: 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 2.31).

```

1 1 <ion-segment (ionChange)="segmentChanged($event)">
2   2   <ion-segment-button value="friends">
3     3     <ion-label>Friends</ion-label>
4   4   </ion-segment-button>
5   5   <ion-segment-button value="enemies">
6     6     <ion-label>Enemies</ion-label>
7   7   </ion-segment-button>
8 </ion-segment>
9
10

```

Kode 2.31: 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 2.32).

```

16 1 <ion-slides pager="true" [options]="slideOpts">
17 2   <ion-slide>
18 3     <h1>Slide 1</h1>
19 4   </ion-slide>
20 5   <ion-slide>
21 6     <h1>Slide 2</h1>
22 7   </ion-slide>
23 8   <ion-slide>
24 9     <h1>Slide 3</h1>
25 10  </ion-slide>
26 11 </ion-slides>
27
28
29

```

Kode 2.32: Kode Program dari Slides

- Tabs

Tabs merupakan navigasi *top-level* yang mengimplementasi sebuah *tab-based navigation*. Tabs dapat digunakan dengan tag `<ion-tabs>` (Kode 2.33) yang tidak memiliki *styling* apapun dan bekerja sebagai *router outlet* untuk menangani navigasi.

```

34 1 <ion-tabs>
35 2   <ion-tab-bar slot="bottom">
36 3     <ion-tab-button tab="schedule">
37 4       <ion-icon name="calendar"></ion-icon>
38 5       <ion-label>Schedule</ion-label>
39 6       <ion-badge>6</ion-badge>
40 7     </ion-tab-button>
41
42
43 8     <ion-tab-button tab="speakers">
44 9       <ion-icon name="person-circle"></ion-icon>
45 10      <ion-label>Speakers</ion-label>
46 11      </ion-tab-button>
47 12    </ion-tab-bar>
48 13  </ion-tabs>
49 14

```

1

Kode 2.33: Kode Program dari Tabs

2 • Toast

3 Komponen Toast merupakan sebuah notifikasi yang digunakan di aplikasi modern untuk
 4 memberikan umpan balik atau menampilkan pesan sistem. Komponen ini muncul diatas
 5 konten aplikasi, dan dapat ditutup untuk melanjutkan penggunaan aplikasi. Komponen ini
 6 menggunakan ToastController yang dimiliki oleh *library* Ionic Angular (Kode 2.34)

```
7
8 1 async presentToast() {
9     2     const toast = await this.toastController.create({
10        3         message: 'Your settings have been saved.',
11        4         duration: 2000
12    5    );
13    6    toast.present();
14 7 }
15
16 8
```

Kode 2.34: Kode Program dari Toast

17 • Toolbar

18 Toolbar dapat diposisikan di atas ataupun di bawah konten. Ketika toolbar ditempatkan
 19 di header <ion-header> akan muncul di bagian atas konten, sedangkan ketika ditempatkan
 20 di footer <ion-footer> akan muncul tetap di bagian bawah. Toolbar menggunakan *tag*
 21 <ion-toolbar>, yang di dalamnya dapat berisi button, dan dapat menggunakan border
 22 (Kode 2.35).

```
23
24 1 <ion-toolbar>
25 2     <ion-buttons slot="start">
26 3         <ion-back-button></ion-back-button>
27 4     </ion-buttons>
28 5     <ion-title>Back Button</ion-title>
29 6 </ion-toolbar>
30
31 7
```

Kode 2.35: Kode Program dari Toolbar dengan Button di Dalamnya

32 Selain komponen-komponen yang telah disebutkan, tertapat beberapa komponen lainnya yang tidak
 33 disebutkan disini. Komponen-komponen tersebut yaitu Checkbox, Chip, Floating Action Button,
 34 Grid, Input, Popover, Progress Indicator, Radio, Reorder, Routing, Searchbar, Select, dan Toggle ¹.

35 **2.3.3 Migrasi Ionic 3 ke Ionic 6**

36 Untuk melakukan migrasi dari Ionic 3 ke Ionic 6 memerlukan tiga tahap, yaitu migrasi dari Ionic 3
 37 ke Ionic 4, migrasi Ionic 4 ke Ionic 5, dan migrasi dari Ionic 5 ke Ionic 6. Tahapan migrasi tersebut
 38 adalah sebagai berikut:

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

1 1. Migrasi Ionic 3 ke Ionic 4

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

3 (a) Membuat Proyek Ionic Baru

4 Untuk membuat projek Ionic baru tanpa *template* apapun dengan menggunakan perintah
 5 **ionic start myApp blank** dan memilih Angular sebagai *frameworknya* [2.36](#).

```
6    1 ionic start myApp blank
  7    2
```

Kode 2.36: Perintah Membuat Proyek Ionic Baru

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

12 (c) Menyalin *Root-level Items*

13 Menyalin seluruh *Root-level Items* pada Ionic versi 3, seperti *pipes* dan *components*.
 14 Lalu terdapat perubahan struktur direktori dengan perubahan direktori yang semula
 15 **src/components** pada Ionic 3, menjadi **src/app/components** pada Ionic 4.

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

18 (e) Menyalin Bagian-bagian Aplikasi

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

- 21 • Shadow DOM sudah aktif secara *default*.
- 22 • Page atau Components Sass tidak lagi dibungkus dengan *tag page* atau *components*
 23 dan harus menggunakan opsi styleUrls milik Angular dari dekorator *@Component*.
- 24 • Perubahan RxJS yang digunakan. Pada ionic 3 adalah versi 5, sedangkan pada Ionic
 25 4 RxJS yang digunakan adalah versi 6.
- 26 • Lifecycle Hooks tertentu harus digantikan dengan Angular Hooks.
- 27 • Perubahan markup yang mungkin saja dibutuhkan.

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

32 – *Button*

33 Terdapat perbedaan pada *tag* untuk membuat Button, yang semula pada Ionic 3
 34 adalah **<button>** menjadi **<ion-button>** pada Ionic 4 (Kode [2.37](#)).

```
35   1 <ion-button (click)="doSomething()">
  36   2   Default Button
  37   3 </ion-button>
  38
  39
```

Kode 2.37: Penggunaan Button pada Ionic 4

41 – Floating Action Button (FAB)

42 Terdapat perbedaan pada *tag* di dalam **<ion-fab>**, yang semula pada Ionic 3
 43 adalah **<button>** menjadi **<ion-fab-button>** pada Ionic 4 (Kode [2.38](#)).

```

1   1 <ion-fab>
2     2   <ion-fab-button>
3       3     <ion-icon name="add"></ion-icon>
4     4   </ion-fab-button>
5     5   <ion-fab-list>
6       6     <ion-fab-button>
7         7       <ion-icon name="logo-facebook"></ion-icon>
8       8     </ion-fab-button>
9     9   </ion-fab-list>
10    10 </ion-fab>
11

```

Kode 2.38: Penggunaan Floating Action Button pada Ionic 4

– Item

Terdapat perbedaan pada tag `<ion-item>`, dimana pada Ionic 3, tag `<ion-label>` akan secara otomatis ditambahkan ke dalam `<ion-item>`. Sedangkan pada Ionic 4 diharuskan untuk menambahkan tag `<ion-label>` secara manual ke dalam komponen item (Kode 2.39).

```

1   1 <ion-item>
2     2   <ion-label>
3       3     Default Item
4     4   </ion-label>
5   5 </ion-item>
6

```

Kode 2.39: Penggunaan Item pada Ionic 4

– Label

Pada Ionic 4, atribut untuk mengatur posisi dari label digabungkan dengan atribut *position* (Kode 2.40).

```

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

```

Kode 2.40: Penggunaan Atribut *Position* pada Ionic 4

– Menu

Terdapat beberapa perubahan nama pada Ionic 4, yaitu:

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

- `swipeEnable`

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

Pada Ionic 3: `swipeEnabled`

Sedangkan pada Ionic 4 menjadi: `swipeGesture`

- 1 · content
- 2 Terdapat perubahan content pada Ionic 4. Perubahan tersebut adalah
- 3 sebagai berikut:
- 4 Pada Ionic 3: content
- 5 Sedangkan pada Ionic 4 menjadi: contentId
- 6 * Perubahan Nama Events Terdapat perubahan nama *events* pada Ionic 4.
- 7 Perubahan-perubahan tersebut adalah sebagai berikut :
- 8 · ionClose
- 9 Terdapat perubahan ionClose pada Ionic 4. Perubahan tersebut adalah
- 10 sebagai berikut:
- 11 Pada Ionic 3: ionClose
- 12 Sedangkan pada Ionic 4 menjadi: ionDidClose
- 13 · ionOpen
- 14 Terdapat perubahan ionOpen pada Ionic 4. Perubahan tersebut adalah
- 15 sebagai berikut:
- 16 Pada Ionic 3: ionOpen
- 17 Sedangkan pada Ionic 4 menjadi: ionDidOpen
- 18 – Nav
- 19 Terdapat perubahan Nav pada Ionic 4. Perubahan-perubahan tersebut adalah
- 20 sebagai berikut:
- 21 * Perubahan Nama Method Terdapat perubahan nama *method* pada Ionic 4.
- 22 Perubahan-perubahan tersebut adalah sebagai berikut:
- 23 · remove
- 24 Terdapat perubahan remove pada Ionic 4. Perubahan tersebut adalah
- 25 sebagai berikut:
- 26 Pada Ionic 3: remove
- 27 Sedangkan pada Ionic 4 untuk menghindari konflik dengan HTML, berubah
- 28 menjadi: removeIndex
- 29 · getActiveChildNavs
- 30 Terdapat perubahan getActiveChildNavs pada Ionic 4. Perubahan tersebut
- 31 adalah sebagai berikut:
- 32 Pada Ionic 3: getActiveChildNavs
- 33 Sedangkan pada Ionic 4 menjadi: getChildNavs
- 34 * Perubahan Nama Prop
- 35 Terdapat perubahan nama prop pada Ionic 4. Perubahan tersebut adalah
- 36 sebagai berikut:
- 37 Pada Ionic 3: swipeBackEnabled
- 38 Sedangkan pada Ionic 4 menjadi: swipeGesture

1 – Navbar

2 Pada Ionic 4, terdapat penghapusan terhadap komponen `<ion-navbar>` karena
3 untuk menjaga agar selalu menggunakan `<ion-toolbar>` dengan *back button*
4 yang eksplisit (Kode 2.41).

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

Kode 2.41: Penggunaan Navbar pada Ionic 4 dengan *Back Button*

14 – Overlays

15 Pada Ionic 4, semua overlay harus menggunakan `async/await`. Overlay tersebut
16 adalah Action Sheet, Alert, Loading, Modal, Popover, and Toast (Kode 2.42).
17 Selain itu, terjadi perubahan nama propert `enableBackdropDismiss` menjadi
18 `backdropDismiss`.

```
19  1 async presentToast() {
20  2   const toast = await this.toastController.create({
21  3     message: 'Your settings have been saved.',
22  4     duration: 2000
23  5   });
24  6   toast.present();
25  7 }
26  8
```

Kode 2.42: Penggunaan Overlay untuk Toast pada Ionic 4

29 – Scroll

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

32 – Segment Button

33 Pada Ionic 4, teks pada Segment Button membutuhkan `<ion-label>` untuk
34 membungkusnya (Kode 2.43).

```
35  1 <ion-segment-button>
36  2   <ion-label>Item One</ion-label>
37  3 </ion-segment-button>
38  4
```

Kode 2.43: Penggunaan Segment Button untuk Toast pada Ionic 4

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

2. Migrasi Ionic 4 ke Ionic 5

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

- CSS

- *Activated, Focused, Hover States*

Kelas `.activated` secara otomatis ditambahkan ke komponen yang dapat diklik, mengalami perubahan nama menjadi `.ion-activated`. Selain itu juga memperbarui komponen Action Sheet sehingga variabel akan diawali dengan `button`. Hal ini dapat memungkinkan aplikasi tetap memiliki kontrol atas `opacity` jika diinginkan, tetapi saat memperbarui status, hanya perlu mengatur variabel utama, yaitu `-background-activated`, `-background-focused`, `-background-hover`. Hal tersebut penting saat mengubah tema global, karena memperbarui warna `toolbar` akan secara otomatis memperbarui *hover states* untuk semua `buttons` di `toolbar` (Kode 2.44).

```

1 /* Setting the button background on hover to solid red */
2 ion-button {
3   --background-hover: red;
4   --background-hover-opacity: 1;
5 }
6
7 /* Setting the action sheet button background on focus to an opaque
8   green */
9 ion-action-sheet {
10   --button-background-focus: green;
11   --button-background-focus-opacity: 0.5;
12 }
13 /*
14 * Setting the fab button background on hover to match the text color
15   with
16 * the default --background-hover-opacity on md
17 */
18 .md ion-fab-button {
19   --color: #222;
20   --background-hover: #222;
21 }
```

Kode 2.44: Contoh Kode *Hover States* pada Ionic 5

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

1 – CSS Utilities

2 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 2.45).

```
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>
5
```

Kode 2.45: Contoh Kode Kelas CSS *Utility* pada Ionic 5

15 – Display Classes

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

17 – Distributed Scss

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

19 • Komponen

20 Terdapat perubahan beberapa komponen pada Ionic 5, yaitu :

21 – Back Button dan Button

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

23 – Controllers

24 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.

25 – Header dan Footer

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

27 – List Header

28 Konten berupa teks apa pun di dalam `<ion-list-header>` harus dibungkus dengan `<ion-label>` sesuai dengan gaya desain yang baru (Kode 2.46). Jika label tidak ada, maka perataan tombol di header bisa saja terlihat tidak aktif.

```
1 <ion-list-header>
2   <ion-label>New This Week</ion-label>
3   <ion-button>See All</ion-button>
4 </ion-list-header>
5
```

Kode 2.46: Kode Program untuk List Header

1 – Menu

2 Fungsi swipeEnable() telah dihapus di Angular, sebagai gantinya menggunakan
 3 swipeGesture(). Lalu nilai *left* dan *right* telah dihapus, gunakan *start* dan *end*
 4 sebagai gantinya. Selain itu ada penghapusan atribut utama, sebagai gantinya yaitu
 5 dengan menggunakan content-id (untuk vanila JS atau Vue) dan contentId (untuk
 6 Angular atau React) (Kode 2.47).

```
7           1 <ion-menu content-id="main"></ion-menu>
8           2 <ion-content id="main">...</ion-content>
9
10          3
```

Kode 2.47: Kode Program untuk Menu

12 – Select Option

13 Properti selected telah dihapus. Sebagai gantinya harus mengatur properti nilai
 14 pada ion-select induk agar sesuai dengan opsi terpilih yang diinginkan (Kode 2.48).

```
15          1 <ion-select value="two">
16           2   <ion-select-option value="one">One</ion-select-option>
17           3   <ion-select-option value="two">Two</ion-select-option>
18           4 </ion-select>
19
20          5
```

Kode 2.48: Kode Program untuk Select Option

22 – Toast

23 Properti close button seperti showCloseButton dan closeButtonText telah dihapus.
 24 Sebagai gantinya, gunakan buttons array untuk fungsi batal (Kode 2.49).

```
25          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 2.49: 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 ³.

- 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 2.50). 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
2
```

Kode 2.50: 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 2.51).

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

Kode 2.51: Kode untuk Memperbarui Versi Ionic 6 dengan versi Angular Terbaru

- Icon

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

³ ‘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 2.52) dan ion-popover (Kode 2.53) menggunakan Shadow DOM.

```
6
7 1 ion-modal::part(content) {
8   ...
9 }
10
11 5 ion-modal::part(backdrop) {
12   ...
13 }
14
```

Kode 2.52: Kode ion-modal menggunakan Shadow DOM pada CSS

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

Kode 2.53: Kode ion-popover menggunakan Shadow DOM pada CSS

28 • Radio

29 Menghapus semua penggunaan antarmuka RadioChangeEventDetail.

1

BAB 3

2

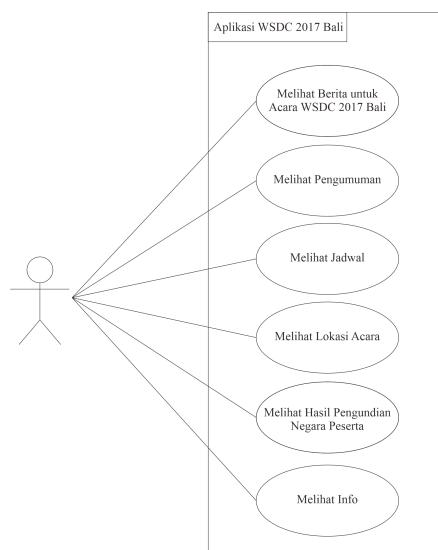
ANALISIS

- 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 Bali
15 menggunakan teknologi web seperti HTML, dan CSS. Lalu untuk membangun aplikasi WSDC 2017
16 Bali agar dapat berjalan secara *native*, digunakanlah Cordova. 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 *sidebar* untuk pengguna agar dapat bermavigasi ke dalam menu-menu yang terdapat pada
 3 aplikasi WSDC 2017 Bali. Untuk mengakses *sidebar*, pengguna dapat menekan tombol navigasi
 4 berada di sebelah kiri atas aplikasi WSDC 2017 Bali. Selain itu dapat pula dengan cara mengusap
 5 layar dari kiri ke kanan. Untuk menutup *sidebar*, pengguna dapat menekan area di luar *sidebar*,
 6 atau dengan cara menekan tombol silang di sebelah kiri atas *sidebar*. Terdapat fitur-fitur yang
 7 ada pada aplikasi WSDC 2017 Bali yang dapat diakses melalui *sidebar*. Fitur-fitur tersebut adalah
 8 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 *sidebar*.

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 sidebar (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 sidebar (Tabel 3.3).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Announcement</i> pada <i>sidebar</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 sidebar (Tabel 3.4).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Schedule</i> pada <i>sidebar</i>	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 sidebar (Tabel 3.5).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Venues</i> pada <i>sidebar</i>	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 sidebar (Tabel 3.6).

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Draw</i> pada <i>sidebar</i>	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 sidebar (Tabel 3.7).

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

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 sidebar (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 side bar
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.

1 Setelah index.html dijalankan, titik masuk ke dalam aplikasi WSDC 2017 Bali adalah file
 2 app.module.ts yang berada di src/app/app.module.ts. Di dalam file ini terdapat NgModule untuk
 3 mendeklarasi komponen apa saja yang akan digunakan, mengimpor module, bootstrap apa yang
 4 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 }
21 export class AppModule {}
```

Kode 3.1: NgModule pada app.module.ts

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

```

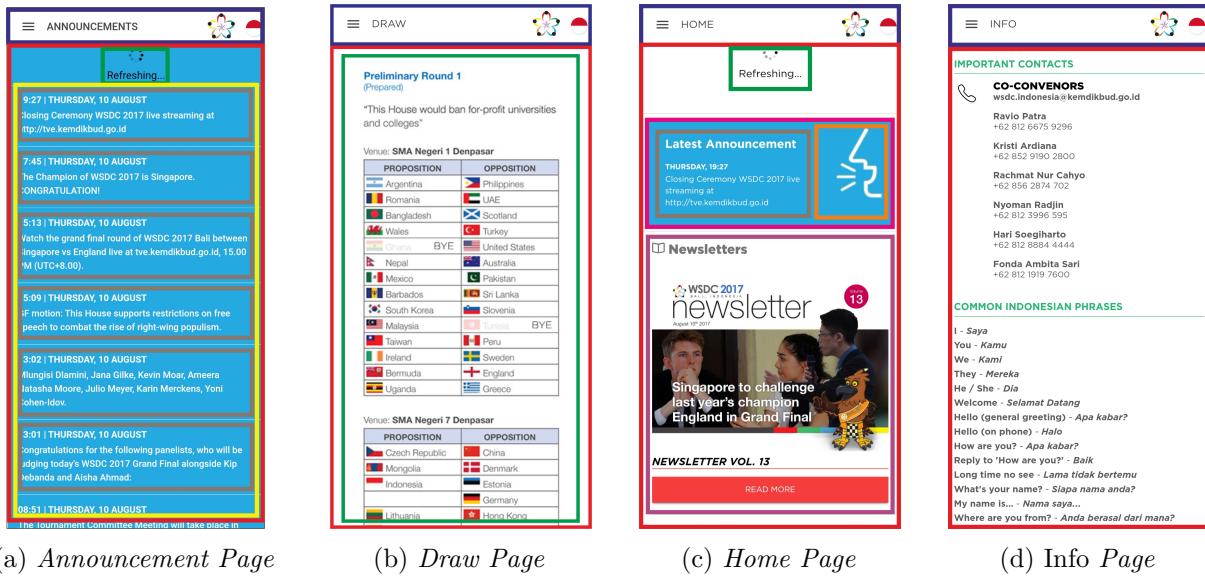
1 <ion-menu [content]="content">
2   <ion-header>
3     <ion-toolbar>
4       <ion-title>
5         <button menuClose id="menu-close-btn">
6           <ion-icon menu-close ios="ios-close-circle-outline" md="md-close-circle">
7         </ion-icon>
8       </button>
9       <span class="text">Menu</span>
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
 36 komponen lain. Lalu, NavController merupakan *base class* untuk mengatur komponen navigasi.
 37 Ini berguna agar aplikasi dapat berpindah antar halaman. Sedangkan Storage berfungsi untuk
 38 menyimpan 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. Lalu *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

Lalu, terdapat sebuah kelas AnnouncementsPage pada komponen ini 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 *sidebar*, <ion-icon> untuk menampilkan icon
 18 dari tombol pada *tag button*, dan <ion-title> untuk menampilkan judul dari halaman,
 19 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. Lalu, @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

Lalu, 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 *sidebar*, <ion-icon> untuk
 18 menampilkan icon dari tombol pada *tag button*, dan *tag* <ion-title> sebagai judul dari
 19 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 ekxternal 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 chache terhadap
 23 halaman ini, maka ionViewDidLoad() tidak akan berjalan lagi. *Lifecycle event* ini
 24 berguna untuk melakukan pengaturan awal, yaitu untuk 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 Lalu, *method* ini akan melakukan penyegaran terhadap data yang sudah ada di dalam
 32 penyimpanan dengan mengambil data dari server dengan batas waktu maksimal untuk
 33 terhubung dengan server. Jika sudah melewati batas waktu, dan aplikasi belum terhubung
 34 dengan server, maka *method* ini akan memanggil *method* showToast() yang akan
 35 menampilkan Toast yang berisi teks ‘Failed to refresh information’. Jika penyegaran
 36 berhasil, maka data yang didapatkan dari server akan dimasukan ke dalam penyimpanan
 37 internal.

- 38 • launch(url: string)

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

- 1 • formatDatetime(sqlDatetime: string)

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

- 6 • doRefresh(refresher)

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

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

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

- 22 • onAnnouncementClick()

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

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

- 29 • *Header*

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

```
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>Home</ion-title>
7 7   </ion-navbar>
8 8 </ion-header>
```

Kode 3.11: *Header* pada home.html

- *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. Lalu kolom kedua ditandai dengan kotak berwarna oranye berisi gambar.

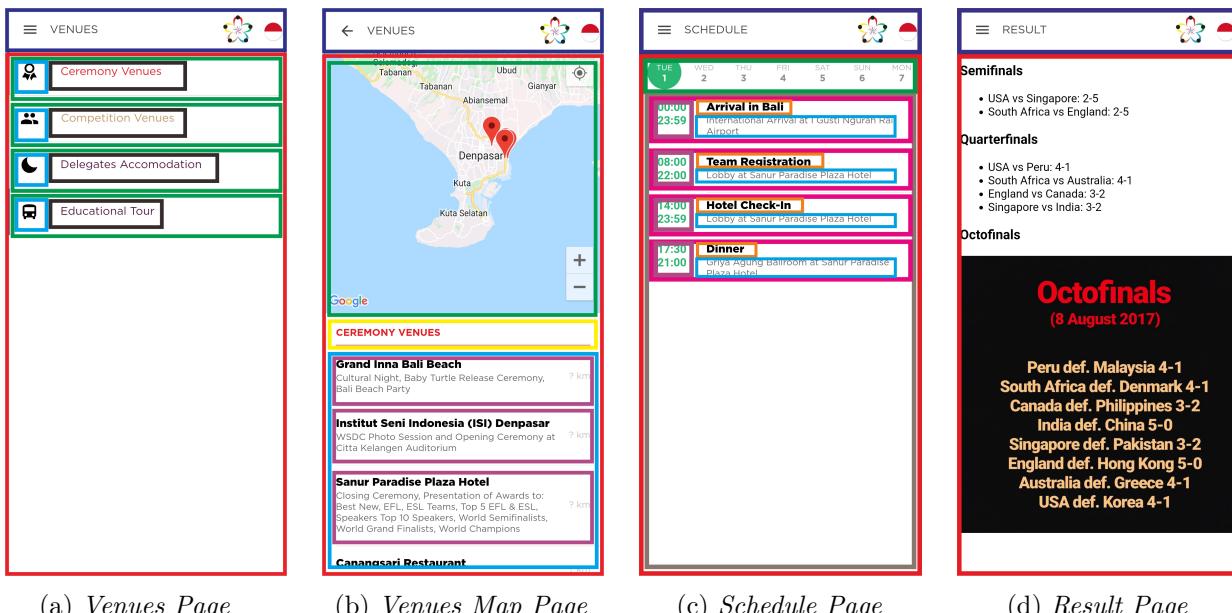
Selanjutnya terdapat sebuah tag `<ion-list>` untuk menyimpan berita-berita terkait acara WSDC 2017 Bali, yang ditandai dengan warna ungu. Di dalam *list* tersebut terdapat tag `<ion-list-header>` sebagai judul dari *list*, dan tag `<ion-item>` untuk menyimpan berita-berita terkait acara WSDC 2017 Bali. 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      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-card (click)="onAnnouncementClick()">
9   9     <ion-grid>
10 10       <ion-row>
11 11         <ion-col col-9>
12 12           <ion-card-header text-wrap>
13 13             Latest Announcement
14 14           </ion-card-header>
15 15           <ion-card-content>
16 16             <h3>{{formatDatetime(wsdcData?.announcements[0].localtime)
17 17             }}</h3>
18 18             <p>{{wsdcData?.announcements[0].message}}</p>
19 19           </ion-card-content>
20 20         </ion-col>
21 21         <ion-col col-3>
22 22           
23 23         </ion-col>
24 24       </ion-row>
25 25     </ion-grid>
26 26   </ion-card>
27 27   <ion-list>
28 28     <ion-list-header>
29 29       <ion-icon ios="ios-book-outline" md="md-book"></ion-icon>
30 30         Newsletters
31 31     </ion-list-header>
32 32     <ion-item *ngFor="let wsdcNews of wsdcData?.newsletters">
33 33       
35 35       <h2 text-wrap>{{wsdcNews.title}}</h2>
36 36       <button ion-button full block color="danger" (click)="launch(
37 37         wsdcNews.url)">Read More</button>
38 38     </ion-item>
39 39   </ion-list>
40 40 </ion-content>
41 41

```

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 memasukkannya 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 sidebar dan tag <ion-icon> untuk menampilkan icon dari tombol pada tag button. Lalu terdapat tag <ion-title> sebagai judul dari halaman, yaitu “Info”.

```

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>Info</ion-title>
7        7   </ion-navbar>
8      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 ekxternal HTML *template* yang akan digunakan. *Template* HTML yang digunakan adalah file info.html. Lalu, `@ViewChild` digunakan untuk memanggil elemen dari DOM untuk memanggil komponen API ke dalam TypeScript, yaitu pada komponen *result* adalah `resultIFrame` yang berada di file result.html.

```

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

```

Kode 3.16: `@Component` pada result.ts

```

1 @ViewChild('resultIFrame') resultIFrame: ElementRef;

```

Kode 3.17: `@ViewChild` pada result.ts

1 Lalu terdapat kelas ResultPage dengan beberapa *method* yang digunakan, yaitu:

- 2 • ionViewDidLoad()

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

- 10 • presentLoading()

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

- 15 • onResultIframeLoad()

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

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

- 22 • *Header*

23 Halaman *result* memiliki *header* dengan tag `<ion-header>` (Kode 3.18) seperti pada
 24 gambar 3.3d. 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 *sidebar*, dan `<ion-icon>` untuk menampilkan
 27 icon dari tombol pada tag *button*. Lalu terdapat tag `<ion-title>` sebagai judul dari
 28 halaman, yaitu “Result”.

```
29
30 1 <ion-header>
31 2   <ion-navbar>
32 3     <button ion-button menuToggle>
33 4       <ion-icon name="menu"></ion-icon>
34 5     </button>
35 6     <ion-title>Result</ion-title>
36 7   </ion-navbar>
37 8 </ion-header>
```

Kode 3.18: *Header* pada result.html

- 39 • *Content*

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

```

1
2     1 <ion-content>
3         2   <iframe #resultIFrame (load)="onResultIframeLoad()" class="iframe-
4             fullscreen"></iframe>
5     3 </ion-content>

```

Kode 3.19: Content pada result.html

6. Komponen *Schedule*

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

```

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

```

Kode 3.20: @Component pada schedule.ts

```

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

```

Kode 3.21: @ViewChild pada schedule.ts

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

- onSegmentChanged(segmentButton)

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

- 1 • onSlideChanged()

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

- 7 • getDayName(sqlDate: string)

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

- 9 • getDate(sqlDate: string)

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

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

- 14 • *Header*

15 Halaman *schedule* memiliki *header* dengan *tag* *<ion-header>* (Kode 3.22) seperti pada
 16 gambar 3.3c. *Tag* tersebut merupakan komponen *parent* yang menampung komponen
 17 navbar yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat
 18 sebuah *tag* *<button>* untuk memunculkan *sidebar* dan *ion-icon* untuk menampilkan
 19 icon dari tombol pada *tag button*. Lalu terdapat *tag* *<ion-title>* sebagai judul dari
 20 halaman, yaitu “Schedule”.

```
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>Schedule</ion-title>
27 7   </ion-navbar>
28 8 </ion-header>
```

Kode 3.22: *Header* pada *schedule.html*

- 31 • *Content*

32 *Content* pada halaman result memiliki *tag* *<ion-content>* (Kode 3.23) yang pada gam-
 33 bar 3.3c dengan kotak berwarna merah. Di dalam *tag* *<ion-content>* terdapat dua buah
 34 *tag* *<div>* yang masing masing berisi *tag* *<ion-segment>* dan *tag* *<ion-slides>*. *Tag*
 35 *<ion-segment>* digunakan untuk tampilan hari, seperti pada gambar 3.3c yang ditandai
 36 dengan kotak berwarna hijau. Lalu, *tag* *<ion-slides>* digunakan untuk tampilan jadwal
 37 di dalam satu hari, seperti yang ditandai dengan kotak berwarna coklat.

38 Setiap jadwal yang berada di *tag* *<ion-slides>* dibungkus dengan *tag* *<ion-list>*
 39 seperti pada kotak berwarna merah muda di gambar 3.3c. Dalam satu *tag* *<ion-list>*
 40 terdapat *tag* *<ion-note>* yang berisi waktu mulai dan waktu selesai dari satu jadwal
 41 seperti yang ditandai dengan kotak berwarna ungu, *tag* *<h3>* berisi nama acara seperti
 42 yang ditandai dengan kotak berwarna oranye, dan *tag* *<p>* yang berisi tempat acara
 43 tersebut diadakan ditandai dengan kotak berwarna biru muda.

```

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

```

Kode 3.23: *Content* pada schedule.html

33 7. Komponen *Venues*

34 Komponen ini digunakan untuk menampilkan halaman *Venues* pada aplikasi. Komponen ini
 35 memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam *file* venues.ts
 36 terdapat sebuah *decorator* @Component untuk komponen (Kode 3.24). Di dalam decorator
 37 ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta templateUrl
 38 untuk mendefinisikan ekxternal HTML *template* yang akan digunakan. *Template* HTML yang
 39 digunakan adalah *file* venues.html

```

40
41 1 @Component({
42 2   selector: 'page-venues',
43 3   templateUrl: 'venues.html'
44 4 })

```

Kode 3.24: @Component pada venues.ts

46 Lalu, terdapat kelas VenuesPage yang memiliki satu *constructor*. *Constructor* kelas ini
 47 berfungsi untuk mengambil data *venues* yang berada di dalam penyimpanan internal. Data
 48 tersebut kemudian disimpan ke dalam variabel lokal valVenues. Selain itu, terdapat sebuah

1 method itemTapped() yang berfungsi untuk berpindah halaman ke halaman venues-map, yang
 2 akan menampilkan peta lokasi berlangsungnya acara, sesuai dengan *venues* yang dipilih.
 3 Selain itu, terdapat file *venues.html* yang digunakan untuk menampilkan halaman *venues*.
 4 Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan
 5 ke dalam halaman *venues*. Diantaranya adalah sebagai berikut:

6 • *Header*

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

```
12 1 <ion-header>
13 2   <ion-navbar>
14 3     <button ion-button menuToggle>
15 4       <ion-icon name="menu"></ion-icon>
16 5     </button>
17 6     <ion-title>Venues</ion-title>
18 7   </ion-navbar>
19 8 </ion-header>
```

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

22 • *Content*

23 *Content* pada halaman *venues* memiliki tag `<ion-content>` (Kode 3.26) yang pada gam-
 24 bar 3.3a dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat sebuah
 25 tag `<ion-grid>` dan sebuah tag `<ion-row>`. Di dalam tag `<ion-row>` terdapat sebuah
 26 tag `<ion-list>` yang berisi tag `<ion-button>` yang ditandai dengan kotak berwarna
 27 hijau pada gambar 3.3a. Masing-masing tag `<ion-button>` berisi tag `<ion-icon>` yang
 28 ditandai dengan kotak berwarna biru muda, dan tag `` berisi nama *venues* yang
 29 ditandai dengan kotak berwarna hitam.

```
30 1 <ion-content>
31 2   <ion-grid>
32 3     <ion-row>
33 4       <ion-list style="width: 100%;" no-lines>
34 5         <button ion-item id="{{wsdcVenue.id}}" *ngFor="let wsdcVenue
35 6           of venuesData" (click)="itemTapped($event, wsdcVenue)">
36 7           <ion-icon ios="ios-{{wsdcVenue.icon}}-outline" md="md-{{
37 8             wsdcVenue.icon}}" item-start></ion-icon>
38 9             <span>{{wsdcVenue.name}}</span>
39 10            </button>
40 11        </ion-list>
41 12      </ion-row>
42 13    </ion-grid>
43 14 </ion-content>
```

Kode 3.26: *Content* pada *venues.html*

Lalu, 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

Lalu, 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.

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

Terdapat beberapa *method* yang digunakan, diantaranya yaitu:

- ngAfterViewInit()

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

1 • `loadMap()`

2 Method ini dipanggil di dalam *constructor*, dan berfungsi untuk menampilkan peta dengan
3 bantuan *plugin* Google Maps. Pada *method* ini, peta pertama kali akan dibuat dengan
4 pengaturan kamera yang mengarah ke lokasi latitude dan longitude dari kota Kuta, Bali.
5 *Plugin* Google Maps sendiri akan memanfaatkan fitur-fitur *native* dari suatu perangkat.
6 Fitur-fitur tersebut adalah untuk melakukan *gesture* seperti *scroll*, *tilt*, *rotate*, dan *zoom*.
7 Lalu fitur untuk menagkses kontrol pada Google Maps, seperti mengakses kompas, tombol
8 lokasi pengguna saat ini, dan melihat peta di dalam ruangan. Saat Google Maps sudah
9 tersedia untuk digunakan, *method* ini akan memanggil *method* `loadMarkers()` untuk
10 membuat penanda pada peta.

11 • `loadMarkers()`

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

16 • `featTapped(event, index)`

17 Method ini merupakan sebuah *template statement* dipanggil oleh *event* di dalam *tag*
18 `<ion-item>` pada `venues_map.html` yaitu *event* (*click*). Saat pengguna melakukan klik
19 di dalam *tag* `<ion-item>`, maka peta akan melakukan *zoom* sesuai dengan lokasi yang
20 ada pada *tag* `<ion-item>`.

21 • `computeDistance(p1, p2)`

22 Method ini digunakan untuk menghitung jarak antara pengguna ke lokasi *venues*. *Method*
23 ini memanfaatkan fitur dari *plugin* Google Maps, yaitu `computeDistanceBetween` dengan
24 parameter lokasi *venues* dan lokasi perangkat pengguna yang didapatkan dari paramter
25 *method* ini.

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

29 • *Header*

30 Halaman *venues* memiliki *header* dengan *tag* `<ion-header>` (Kode 3.28) seperti pada
31 gambar 3.3b. *Tag* tersebut merupakan komponen *parent* yang menampung komponen
32 navbar seperti yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut,
33 terdapat sebuah *tag* `<button>` untuk memunculkan *sidebar* dan `<ion-icon>` untuk
34 menampilkan icon dari tombol pada *tag* *button*. Lalu terdapat *tag* `<ion-title>` sebagai
35 judul dari halaman, yaitu “Venues”.

```

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

```

Kode 3.28: *Header* pada venues_map.html

- *Content*

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

```

1 <ion-content>
2   <div #map id="map"></div>
3   <h3 #pagetitle>
4     {{selectedItem.name}}
5   </h3>
6   <ion-scroll scrollY="true">
7     <ion-list>
8       <ion-item text-wrap *ngFor="let feature of items; let i=index" (
9         click)="featTapped($event, i)">
10         <h2>{{feature.name}}</h2>
11         <p>{{feature.description}}</p>
12         <ion-note item-end>
13           {{feature.distance}}
14         </ion-note>
15       </ion-item>
16     </ion-list>
17   </ion-scroll>
18 </ion-content>

```

Kode 3.29: *Content* pada venues_map.html

1 3.2 Analisis Sistem Usulan

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

8 3.2.1 Analisis Kebutuhan

9 Aplikasi WSDC 2017 Bali yang akan dibangun akan mengadopsi desain dan tata letak yang sama
10 persis dengan aplikasi WSDC 2017 Bali saat ini. Namun dengan perubahan penggunaan Ionic
11 Framework yang digunakan, yaitu versi 5, serta Angular versi 12. Pada Ionic Framework terbaru
12 saat skripsi ini dibuat, aplikasi WSDC 2017 Bali yang akan dibangun akan memanfaatkan fasilitas
13 yang disediakan oleh Ionic Framework, yaitu UI Component, dan CSS Utilities.

14 Struktur yang dibuat akan menggunakan struktur Ionic Framework versi 5, namun menggunakan
15 komponen-komponen yang sama dengan Ionic Framework versi 3. Tapi, karena terdapat beberapa
16 perubahan, maka perubahan di dalam komponen seperti UI Component, dan CSS akan mengikuti
17 Ionic Framework versi 5.

18 UI Component yang akan digunakan akan mengikuti perkembangan pada Ionic Framework
19 versi 5. Pada setiap komponen, akan terdapat sebuah *header* dengan tag `<ion-header>`. Tag
20 tersebut akan membungkus tag `<ion-toolbar>` sebagai *toolbar* dari aplikasi. Tag ini akan meng-
21 gantikan tag `<ion-navbar>` yang telah dihapus sejak Ionic versi 4. Didalam `<ion-toolbar>`
22 terdapat tag `<ion-buttons>` sebagai pengganti tag `<button>` yang dihapus sejak Ionic versi 4,
23 dan tag `<ion-title>`. Dibandingkan dengan aplikasi sistem kini, terdapat perubahan pada tag
24 `<ion-toolbar>` yang semula bernama `<ion-navbar>` pada Ionic Framework versi 3. Selain itu
25 ada tag `<ion-buttons>` yang semula bernama `<button>`. Di dalam tag `<ion-buttons>` akan ada
26 sebuah tag `<ion-menu-button>`.

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

29 1. *Announcement*

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

- 32 • *Content*

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

- 37 • *Refresher*

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

1 • *List*

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

6 • *Item*

7 *Item* dengan *tag* `<ion-item>` sejak Ionic 4 mengalami perubahan dibandingkan pada
 8 Ionic 3, yaitu wajib menambahkan *label* dengan *tag* `<ion-label>`. Pada aplikasi WSDC
 9 2017 Bali saat ini yang menggunakan Ionic 3, tidak terdapat *tag* `<ion-label>` pada
 10 `<ion-item>` (Kode 3.30). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi
 11 yang akan dibangun yang menggunakan Ionic 6, akan menggunakan *tag* `<ion-label>` di
 12 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

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

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

28 2. *Draw*

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

34 3. *Home*

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

36 • *Content*

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

41 • *Refresher*

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

1 • *Card*

2 Komponen ini akan digunakan sebagai tampilan antar muka, yang dapat menjadi
 3 titik masuk ke dalam informasi yang lebih detail. UI Component *Card* dengan tag
 4 <ion-card>, <ion-card-title> dan <ion-card-content> pada Ionic Framework versi
 5 6 tidak mengalami perubahan dari Ionic Framework versi 3.

6 • *Grid*

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

11 • *List Header*

12 *List Header* dengan tag <ion-list-header> sejak Ionic 4 diwajibkan untuk selalu me-
 13 nambahkan tag <ion-label>. Pada aplikasi WSDC 2017 Bali saat ini yang menggunakan
 14 Ionic 3, tidak terdapat tag <ion-label> pada <ion-list-header> (Kode 3.32). Se-
 15 dangkan sesuai dengan ketentuan yang berlaku, pada aplikasi yang akan dibangun yang
 16 menggunakan Ionic 6, akan menggunakan tag <ion-label> di dalam <ion-item-header>
 17 (Kode 3.33).

```
18
19   1 <ion-list-header>
20   2   <ion-icon ios="ios-book-outline" md="md-book"></ion-icon>
21   3   Newsletters
22   4 </ion-list-header>
```

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

```
24
25   1 <ion-list-header>
26   2   <ion-icon name="book-outline"></ion-icon>
27   3   <ion-label>Newsletters</ion-label>
28   4 </ion-list-header>
```

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

30 • *Icon*

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

34 • *Button*

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

```
41
42   1 <button ion-button full block color="danger" (click)="launch(wsdcNews.url
43   2   )">Read More</button>
```

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

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

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>
9
10

```

Kode 3.36: Tag <ion-item> 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 <ion-item> 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     2     <div class="day">{{getDayName(schedule.date)}}</div>
45     3     <div class="date">{{getDate(schedule.date)}}</div>
46   4 </ion-segment-button>

```

Kode 3.38: Tag <ion-segment-button> 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       3 <div class="day">{{getDayName(schedule.date)}}</div>
5       4 <div class="date">{{getDate(schedule.date)}}</div>
6     </ion-label>
7   6 </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>` dan `<ion-slide>` 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 ini akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area yang dapat digulir dan menampilkan isi konten dari halaman *venues*. 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. 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). Lalu 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 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>
7

```

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

```

9 1 <ion-button ion-item id="{{wsdcVenue.id}}" *ngFor="let wsdcVenue of
10   venuesData" (click)="itemTapped($event, wsdcVenue)">
11   <ion-icon ios="ios-{{wsdcVenue.icon}}-outline" md="md-{{wsdcVenue.icon
12     }}" item-start></ion-icon>
13   <span>{{wsdcVenue.name}}</span>
14 </ion-button>
15

```

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 5 tidak mengalami perubahan dari Ionic Framework versi 3.

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

3.2.2 Permasalahan Pengembangan Sistem Usulan

Saat sedang melakukan proses migrasi aplikasi WSDC 2017 Bali dari Ionic Framework versi 3 ke Ionic Framework versi 5, terdapat beberapa kendala yang dialami. Kendala-kendala tersebut adalah sebagai berikut :

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

¹

BAB 4

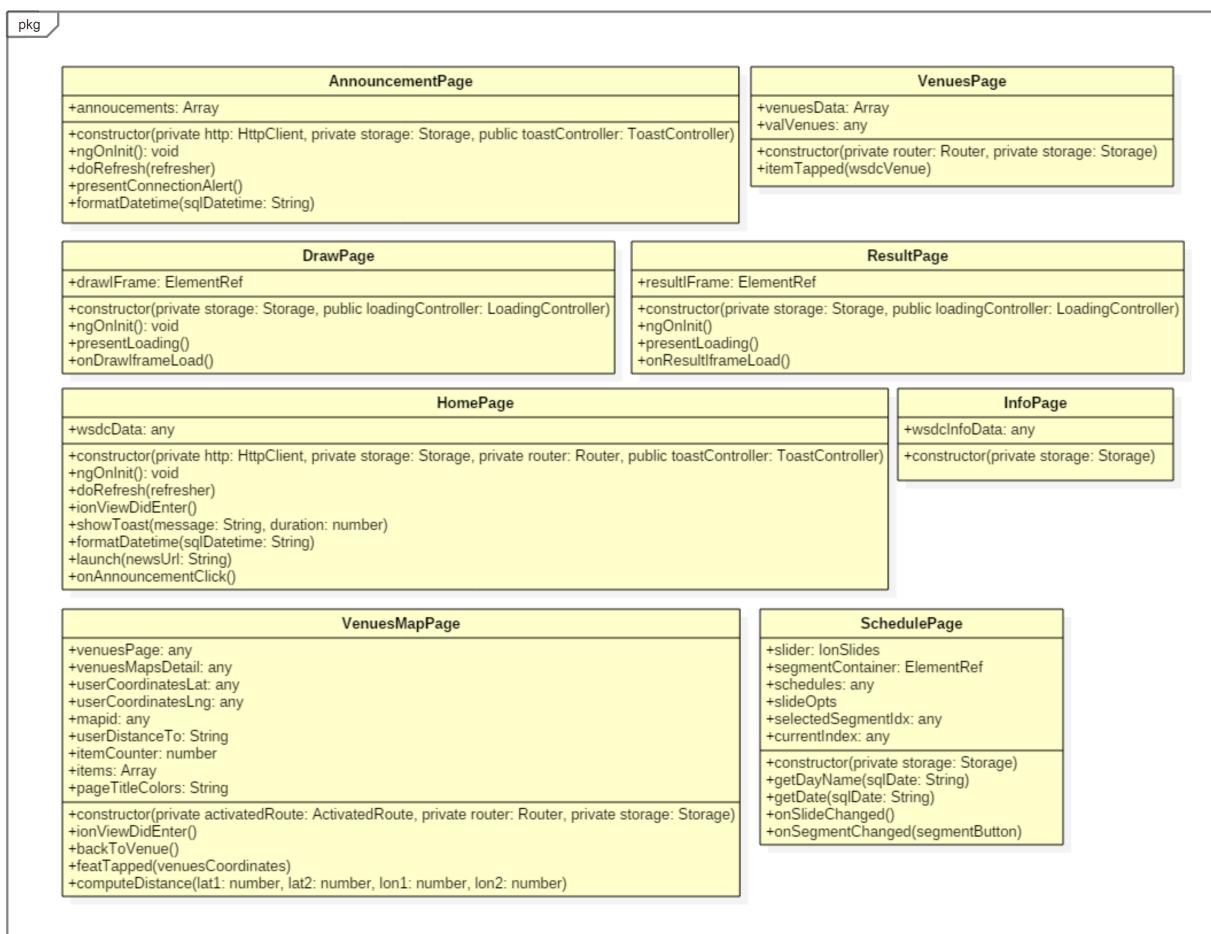
²

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. Lalu, 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

Perancangan kelas dari masing-masing komponen adalah sebagai berikut:

1. Komponen *Announcements*

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

- **constructor(private http: HttpClient, private storage: Storage, 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.
- **toastController**: Parameter ini digunakan untuk menyimpan API ToastController.

Kembalian: tidak ada.

1 • **ngOnInit(): void**

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

4 **Parameter:** tidak ada.

5 **Kembalian:** tidak ada.

6 • **doRefresh(refresher)**

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

14 **Parameter:** *Method* ini memiliki sebuah parameter yaitu *refresher*, yang berisi *event*
15 *refresher*.

16 **Kembalian:** tidak ada.

17 • **presentConnectionAlert()**

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

21 **Parameter:** tidak ada.

22 **Kembalian:** tidak ada.

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

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

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

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

28 2. Komponen *Draw*

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

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

35 • **constructor(private storage: Storage, public loadingController: LoadingCon-**
36 **troller)**

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

38 **Parameter:**

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

40 – **loadingController**: Parameter ini digunakan untuk menyimpan API LoadingCon-
41 troller.

42 **Kembalian:** tidak ada.

1 • **ngOnInit()**

2 *Method* ini berfungsi untuk mengambil data *draws* yang terdapat di dalam *storage*. Data
3 tersebut lalu disimpan ke dalam *child drawIFrame*. Kemudian *method* ini akan memanggil
4 *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 • **onDrawIframeLoad()**

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

16 **Parameter:** tidak ada.

17 **Kembalian:** tidak ada.

18 3. Komponen *Home*

19 Di dalam komponen *home* terdapat sebuah kelas *HomePage*. Kelas ini menjadi sebagai
20 kelas yang pertama kali diakses oleh aplikasi. Maka dari itu, kelas ini berfungsi untuk
21 menginisialisasi sebuah *storage* yang diisi oleh data yang digunakan oleh seluruh kelas pada
22 aplikasi. Kelas ini memiliki sebuah atribut, yaitu *wsdcData* yang bertipe *any*. Atribut ini
23 akan menyimpan data json berisi data yang akan digunakan untuk aplikasi.

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

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

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

27 **Parameter:**

28 – **http:** Parameter ini digunakan untuk menyimpan API *HttpClient*.

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

30 – **router:** Parameter ini digunakan untuk menyimpan API *Router*.

31 – **toastController:** Parameter ini digunakan untuk menyimpan API *ToastController*.

32 **Kembalian:** tidak ada.

33 • **ionViewDidEnter()**

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

36 **Parameter:** tidak ada.

37 **Kembalian:** tidak ada.

38 • **ngOnInit(): void()**

39 *Method* ini dipanggil ketika setelah semua properti telah diinisialisasi. *Method* ini
40 berfungsi untuk mengambil data json dari *storage*. Jika data di dalam *storage* tersebut
41 belum pernah dibuat sebelumnya, maka *method* ini akan membuat sebuah data baru

1 di dalam *storage* yang berisi data json yang dibutuhkan untuk aplikasi. Secara *default*,
2 untuk mengantisipasi jika pengguna tidak memiliki koneksi internet, maka pertama kali
3 *method* ini mengambil data adalah dari asset yang berada di *folder assets*. Setelah itu,
4 *method* ini akan mengambil data terbaru dari server. Jika server tidak merespon dan
5 data tidak berhasil didapatkan, maka akan memanggil *method* `showToast()`.

6 **Parameter:** tidak ada.

7 **Kembalian:** tidak ada.

8 • **doRefresh(refresher)**

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

16 **Parameter:** tidak ada.

17 **Kembalian:** tidak ada.

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

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

21 **Parameter:**

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

24 **Kembalian:** tidak ada.

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

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

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

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

30 • **launch(newsUrl: string)**

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

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

34 **Kembalian:** tidak ada.

35 • **onAnnouncementClick()**

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

37 **Parameter:** tidak ada.

38 **Kembalian:** tidak ada.

1 4. Komponen Info

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

9 5. Komponen *Result*

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

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

- 16 • **constructor(private storage: Storage, public loadingController: LoadingCon-**
17 **troller)**

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

19 **Parameter:**

- 20 – **storage:** Parameter ini digunakan untuk menyimpan API Storage.
- 21 – **loadingController:** Parameter ini digunakan untuk menyimpan API LoadingCon-
22 troller.

23 **Kembalian:** tidak ada.

- 24 • **ngOnInit()**

25 *Method* ini dipanggil ketika setelah semua properti telah diinisialisasi. *Method* ini
26 berfungsi untuk mengambil data *result* yang terdapat di dalam *storage*. Data tersebut
27 lalu disimpan ke dalam *child* resultIFrame. Setelah itu *method* ini akan memanggil
28 *method* presentLoading().

29 **Parameter:** tidak ada.

30 **Kembalian:** tidak ada.

- 31 • **async presentLoading()**

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

34 **Parameter:** tidak ada.

35 **Kembalian:** tidak ada.

- 36 • **onResultIframeLoad()**

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

40 **Parameter:** tidak ada.

41 **Kembalian:** tidak ada.

1 6. Komponen *Schedule*

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

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

- 6 • **slider**: Atribut ini merupakan sebuah @ViewChild yang digunakan untuk memanggil
7 elemen dari komponen *schedule*, yaitu scheduleSlider pada file schedule.page.html.
- 8 • **segmentContainer**: Atribut ini merupakan sebuah @ViewChild yang digunakan untuk
9 memanggil elemen dari komponen *schedule*, yaitu segmentContainer pada file schedu-
10 le.page.html.
- 11 • **schedules**: Atribut ini akan menyimpan data *schedules* yang diambil dari *storage*.
- 12 • **slideOpts**: Atribut ini berisi pengaturan dasar untuk *slides*. Berisi initialSlide untuk
13 mengatur *slides* ke berapa saat pertama kali halaman *schedule* dibuka, dan speed untuk
14 mengatur kecepatan transisi antar *slides*.
- 15 • **selectedSegmentIdx**: Atribut ini digunakan untuk menyimpan index dari *segment*
16 yang sedang aktif.
- 17 • **currentIndex**: Atribut ini digunakan untuk menyimpan index dari *slides* yang sedang
18 aktif.

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

- 20 • **constructor(private storage: Storage)**

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

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

23 **Kembalian**: tidak ada.

- 24 • **getDayName(sqlDate: string)**

25 *Method* ini berfungsi untuk mengambil hari dari parameter.

26 **Parameter**: sqlDate: Sebuah *string* yang berisi tahun, bulan, dan tanggal.

27 **Kembalian**: *string* nama hari.

- 28 • **getDate(sqlDate: string)**

29 *Method* ini berfungsi untuk mengambil tanggal dari parameter.

30 **Parameter**: sqlDate: Sebuah *string* yang berisi tahun, bulan, dan tanggal.

31 **Kembalian**: *string* tanggal.

- 32 • **onSlideChanged()**

33 *Method* ini dipanggil saat *slides* dipindahkan dengan cara digeser ke kanan atau ke kiri.

34 *Method* ini akan mengubah atribut currentIndex menjadi index *slides* saat ini, kemudian

35 mengubah atribut selectedSegmentIdx menjadi index *slides* saat ini. Hal ini bertujuan

36 agar indeks dari *segment* yang aktif dapat diganti sesuai dengan indeks *slides* yang aktif.

37 Dengan begitu tampilan *segment* dan *slides* yang aktif akan sesuai.

38 **Parameter**: tidak ada.

39 **Kembalian**: tidak ada.

1 • **onSegmentChanged(segmentButton)**

2 Method ini berfungsi untuk mengubah *slides* yang aktif sesuai dengan indeks dari *segment*
3 yang sedang aktif.

4 **Parameter:** segmentButton: Merupakan sebuah *event* dari *segment* yang akan diambil
5 *value* yang berisi indeks dari *segment* yang aktif.

6 **Kembalian:** tidak ada.

7. Komponen *Venues*

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

11 Kelas ini memiliki beberapa atribut, yaitu:

- 12 • **venuesData:** Atribut ini bertipe array. Atribut ini digunakan untuk menyimpan data
13 dari *venues* yang berisi id, name, icon, geojson, dan colorIdx dari masing masing kategori
14 *venues*. Data ini nantinya akan dikirimkan ke kelas *Venues Maps*.

- 15 • **valVenues:** Digunakan untuk menyimpan data *venues* yang didapatkan dari *storage*.

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

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

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

19 **Parameter:**

20 – **router:** Parameter ini berfungsi untuk menyimpan API Router.

21 – **storage:** Parameter ini berfungsi untuk menyimpan API Storage.

22 **Kembalian:** tidak ada.

- 23 • **itemTapped(wsdcVenue)**

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

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

29 **Kembalian:** tidak ada.

30. Komponen *Venues Maps*

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

- 35 • **venuesPage :** Atribut ini digunakan untuk menyimpan data yang dikirimkan dari
36 komponen *Venues*.

- 37 • **venuesMapsDetail :** Atribut ini digunakan untuk menyimpan data *venues* dari *storage*
38 sesuai dengan kategori yang sedang dipilih.

- 39 • **userCoordinatesLat :** Atirbut ini digunakan untuk menyimpan koordinat latitude dari
40 perangkat pengguna.

- 41 • **userCoordinatesLng :** Atirbut ini digunakan untuk menyimpan koordinat longitude
42 dari perangkat pengguna.

- 1 • **mapid** : Atribut ini digunakan untuk menyimpan id dari peta.
- 2 • **userDistanceTo** : Atribut ini digunakan untuk menyimpan jarak dari posisi perangkat
3 pengguna ke posisi *venues*.
- 4 • **itemCounter** : Atribut ini digunakan sebagai penghitung berapa banyak *venues* dalam
5 sebuah kategori.
- 6 • **items** : Atribut ini digunakan untuk menyimpan id dan id warna dari sebuah katego-
7 ri *venues*.
- 8 • **pageTitleColors** : Atribut ini merupakan sebuah *array* yang berisi kode warna un-
9 tuk masing-masing judul kategori *venues*.

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

- 11 • **constructor(private activatedRoute: ActivatedRoute, private router: Router,
12 private storage: Storage)**

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

16 **Parameter:**

- 17 – **activatedRoute** : Parameter ini digunakan untuk menyimpan API *ActivatedRoute*.
- 18 – **router** : Parameter ini digunakan untuk menyimpan API *Router*.
- 19 – **storage** : Parameter ini digunakan untuk menyimpan API *Storage*.

20 **Kembalian:** tidak ada.

- 21 • **ionViewDidEnter()**

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

29 **Parameter:** tidak ada.

30 **Kembalian:** tidak ada.

- 31 • **backToVenue()**

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

33 **Parameter:** tidak ada.

34 **Kembalian:** tidak ada.

- 35 • **featTapped(venuesCoordinates)**

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

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

40 **Kembalian:** tidak ada.

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

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

4 **Parameter:**

- 5 – **lat1**: koordinat latitude dari pengguna.
6 – **lat2**: koordinat latitude dari *venues*.
7 – **lon1**: koordinat longitude dari pengguna.
8 – **lon2**: koordinat longitude dari *venues*.

9 **Kembalian:** sebuah *string* yang berisi jarak dari posisi perangkat pengguna ke posisi
10 *venues*.

11

4.2 Perancangan Struktur HTML

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

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

18 1. *Header*

19 *Header* untuk setiap komponen pada umumnya memiliki struktur yang serupa namun dibedakan
20 dengan judul dari setiap halaman. *Header* digunakan untuk menampilkan judul dari sebuah
21 halaman, serta menyediakan sebuah *menu button* sebagai salah satu cara untuk membuat
22 *sidebar* untuk melakukan navigasi antar halaman. *Header* dibungkus oleh tag `<ion-header>`
23 yang didalamnya terdapat tag `<ion-toolbar>` yang disediakan Ionic Framework. Lalu untuk
24 *menu button* dibuat oleh tag `<ion-menu-button>` pada tag `<ion-buttons>`. Selanjutnya
25 untuk judul dari sebuah halaman dibungkus oleh tag `<ion-title>`. Judul akan berbeda beda
26 tergantung dengan halamannya. Salah satu contoh dari penggunaan *header* adalah pada
27 gambar [3.2c](#) yang ditandai dengan kotak berwarna biru.

28 2. *Content*

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

34 • Halaman *Announcements*

35 *Content* pada halaman *announcements* berisi sebuah *refresher* dengan tag `<ion-refresher>`
36 untuk melakukan penyegaran ulang terhadap halaman *announcements* yaitu mengambil
37 data terbaru dari server. Penggunaan tag `<ion-refresher>` seperti pada gambar [3.2a](#)
38 yang ditandai dengan kotak berwarna hijau. Selain itu terdapat sebuah list dengan tag
39 `<ion-list>` yang ditandai dengan kotak berwarna kuning. List menampilkan pengumuman
40 yang berisi waktu dan tanggal, serta pesan dari pengumuman tersebut. Setiap
41 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 menyematkan 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>`. Lalu 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>`. Lalu 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

1 keseluruhan pertandingan WSDC 2017 Bali. Data tersebut diambil dari server, kemudian
2 dimasukan ke dalam tag `<iframe>`.

3 • Halaman *Schedule*

4 Content pada halaman *schedule* berisi *segment* dengan tag `<ion-segment>` dan sebuah
5 *slides* dengan tag `<ion-slides>`. *Segment* digunakan untuk menampilkan tanggal dan
6 hari, serta berfungsi untuk memindahkan *slides* ke hari yang dipilih seperti yang ditandai
7 dengan kotak berwarna hijau pada gambar 3.3c. Untuk melakukan hal tersebut, di
8 dalam *segment* terdapat sebuah *button* dengan tag `<ion-segment-button>` yang berisi
9 tag `<ion-label>` untuk menampung tanggal dan hari. Sedangkan *slides* digunakan untuk
10 menampilkan jadwal acara WSDC 2017 Bali pada hari yang sesuai dengan *segment* yang
11 terpilih seperti yang ditandai dengan kotak berwarna coklat. Untuk menampilkan jadwal
12 menggunakan *list* dengan tag `<ion-list>` yang ditandai dengan warna merah muda.
13 Di dalam *list* terdapat sebuah *item* dengan tag `<ion-item>`. Untuk setiap *item* berisi
14 waktu mulai dan selesai sebuah acara dengan tag `<ion-note>` yang ditandai dengan
15 warna ungu, serta nama acara dengan tag `<h3>` yang ditandai dengan kotak berwarna
16 jingga dan lokasi acara dengan tag `<p>` yang ditandai dengan kotak berwarna biru muda.
17 Nama dan lokasi acara dibungkus dengan tag `<ion-label>`.

18 • Halaman *Venues*

19 Content pada halaman *venues* berisi *grid* dengan tag `<ion-grid>` dengan satu baris
20 menggunakan tag `<ion-row>`. Di dalamnya terdapat sebuah *list* dengan tag `<ion-list>`.
21 *List* tersebut berisi tombol dengan tag `<ion-button>` yang ditandai dengan kotak ber-
22 warna hijau pada gambar 3.3a. Masing-masing tombol digunakan untuk melakukan
23 navigasi ke halaman *venues map*. Setiap tombol berisi ikon dengan tag `<ion-icon>` yang
24 ditandai dengan kotak berwarna biru muda pada gambar 3.3a dan sebuah tag ``
25 yang berisi nama dari *venues*, ditandai dengan kotak berwarna hitam.

26 • Halaman *Venues Map*

27 Content pada halaman *venues map* berisi tag `<div>` yang digunakan untuk menampung
28 peta dari sebuah kategori *venues* seperti yang ditandai dengan kotak berwarna hijau
29 pada gambar 3.3b. Lalu selain itu terdapat sebuah label dengan tag `<ion-label>` yang
30 berisi nama kategori *venues* yang ditandai dengan kotak berwarna kuning. Kemudian
31 untuk menampilkan nama dan deskripsi sebuah *venues*, serta jarak antara pengguna
32 dengan *venues*, menggunakan *list* dengan tag `<ion-list>` yang ditandai dengan kotak berwarna
33 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

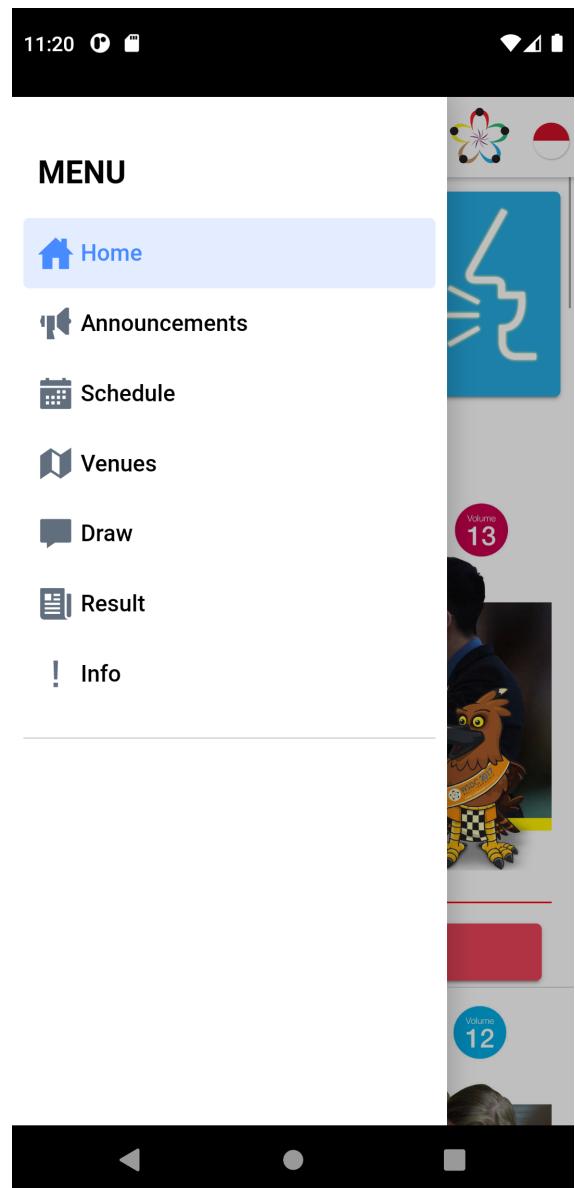
5.1.2 Hasil Implementasi

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 *sidebar*. Tangkapan layar *sidebar* dapat dilihat pada
20 Gambar 5.1b. Halaman-halaman yang ada pada aplikasi WSDC 2017 Bali tersebut yaitu:

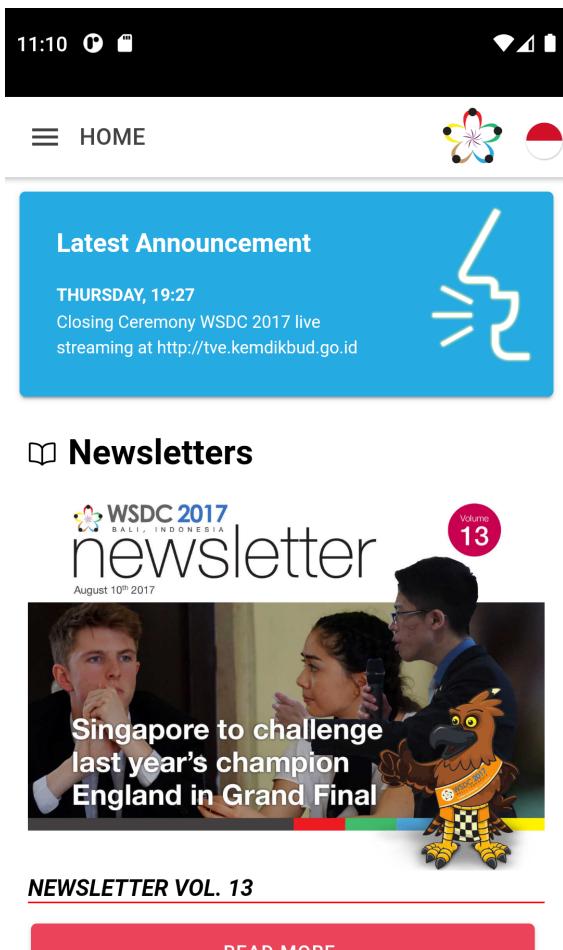
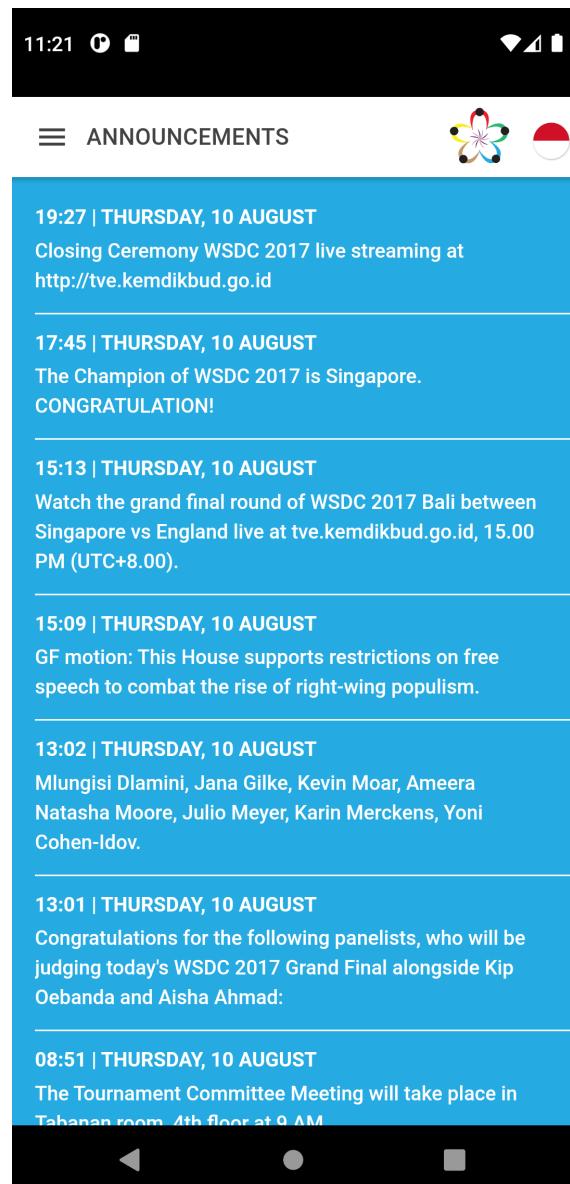
21 1. Halaman *Home*

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 5.2a.

- 1 2. Halaman *Announcements* Halaman *announcements* berisi pengumuman-pengumuman terkait dengan acara WSDC 2017 Bali yang disajikan terurut menurun dengan waktu terbaru yang pertama. Tangkapan layar halaman *announcements* dapat dilihat pada Gambar 5.2b.
- 2 3. Halaman *Schedule* Halaman *schedule* berisi jadwal acara WSDC 2017 Bali yang ditampilkan berkelompok berdasarkan tanggal dan hari. Jadwal yang ditampilkan berupa waktu mulai dan waktu selesai, lokasi acara, serta nama acara. Pengguna dapat berpindah ke hari manapun untuk melihat jadwal yang ada pada hari tersebut dengan menggulir menyamping pada bagian tanggal dan hari, serta bagian jadwal. Tangkapan layar halaman *schedule* dapat dilihat pada Gambar 5.4b.
- 3 4. Halaman *Venues* Halaman *venues* berisi kategori *venues* WSDC 2017 Bali. Setiap kategori yang ditampilkan merupakan sebuah tombol yang dapat diklik untuk mengarahkan pengguna ke halaman *venues map*. Tangkapan layar halaman *venues* dapat dilihat pada Gambar 5.5a.
- 4 5. Halaman *Venues Map* Halaman *venues map* berisi lokasi *venues* yang digunakan oleh WSDC 2017 Bali. Lokasi tersebut ditampilkan dengan peta, dan detail dari lokasi ditampilkan dengan *list* yang berisi nama *venues*, lokasi *venues*, serta jarak dari pengguna ke lokasi *venues*. Tangkapan layar dari halaman *venues map* dapat dilihat pada Gambar 5.5b.
- 5 6. Halaman *Draw* Halaman *Draw* menampilkan hasil dari pembagian grup oposisi dan proposisi dari negara-negara peserta WSDC 2017 Bali. Tangkapan layar halaman *draw* dapat dilihat pada Gambar 5.3a.
- 6 7. Halaman *Result* Halaman *result* menampilkan hasil dari pertandingan WSDC 2017 Bali pada babak seperdelapan final, seperempat final, dan semifinal. Tangkapan layar dari halaman *result* dapat dilihat pada Gambar 5.4a.
- 7 8. Halaman Info Halaman info menampilkan info-info seperti kontak-kontak penting yang dapat dihubungi, kosa kata dalam Bahasa Indonesia sehari-hari, serta *credits* kepada pembuat aplikasi WSDC 2017 Bali. Tangkapan layar dari halaman info dapat dilihat pada Gambar 5.3b.

(a) *Splash Screen*(b) *Sidebar*

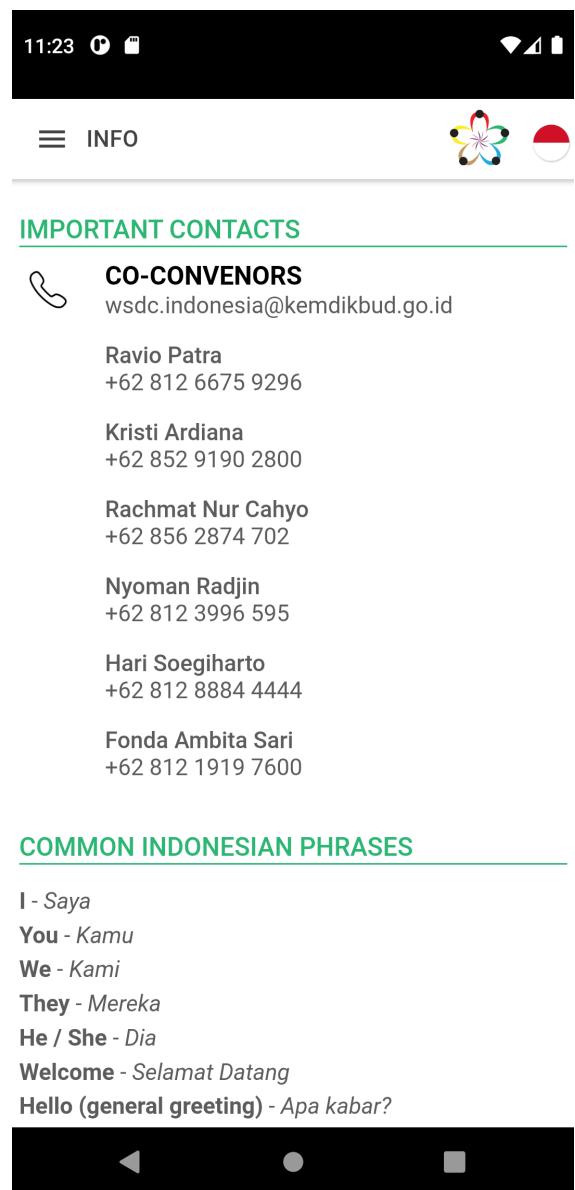
Gambar 5.1: Tangkapan Layar Aplikasi WSDC 2017 Bali

(a) Halaman *Home*(b) Halaman *Announcements*

Gambar 5.2: Tangkapan Layar Aplikasi WSDC 2017 Bali



(a) Halaman Draw



(b) Halaman Info

Gambar 5.3: Tangkapan Layar Aplikasi WSDC 2017 Bali



≡ RESULT



Semifinals

- USA vs Singapore: 2-5
- South Africa vs England: 2-5

Quarterfinals

- USA vs Peru: 4-1
- South Africa vs Australia: 4-1
- England vs Canada: 3-2
- Singapore vs India: 3-2

Octofinals



(a) Halaman *Result*



≡ SCHEDULE

TUE 1	WED 2	THU 3	FRI 4	SAT 5	SUN 6
----------	----------	----------	----------	----------	----------

00:00 **23:59** **Arrival in Bali**
International Arrival at I Gusti Ngurah Rai Airport

08:00 **22:00** **Team Registration**
Lobby at Sanur Paradise Plaza Hotel

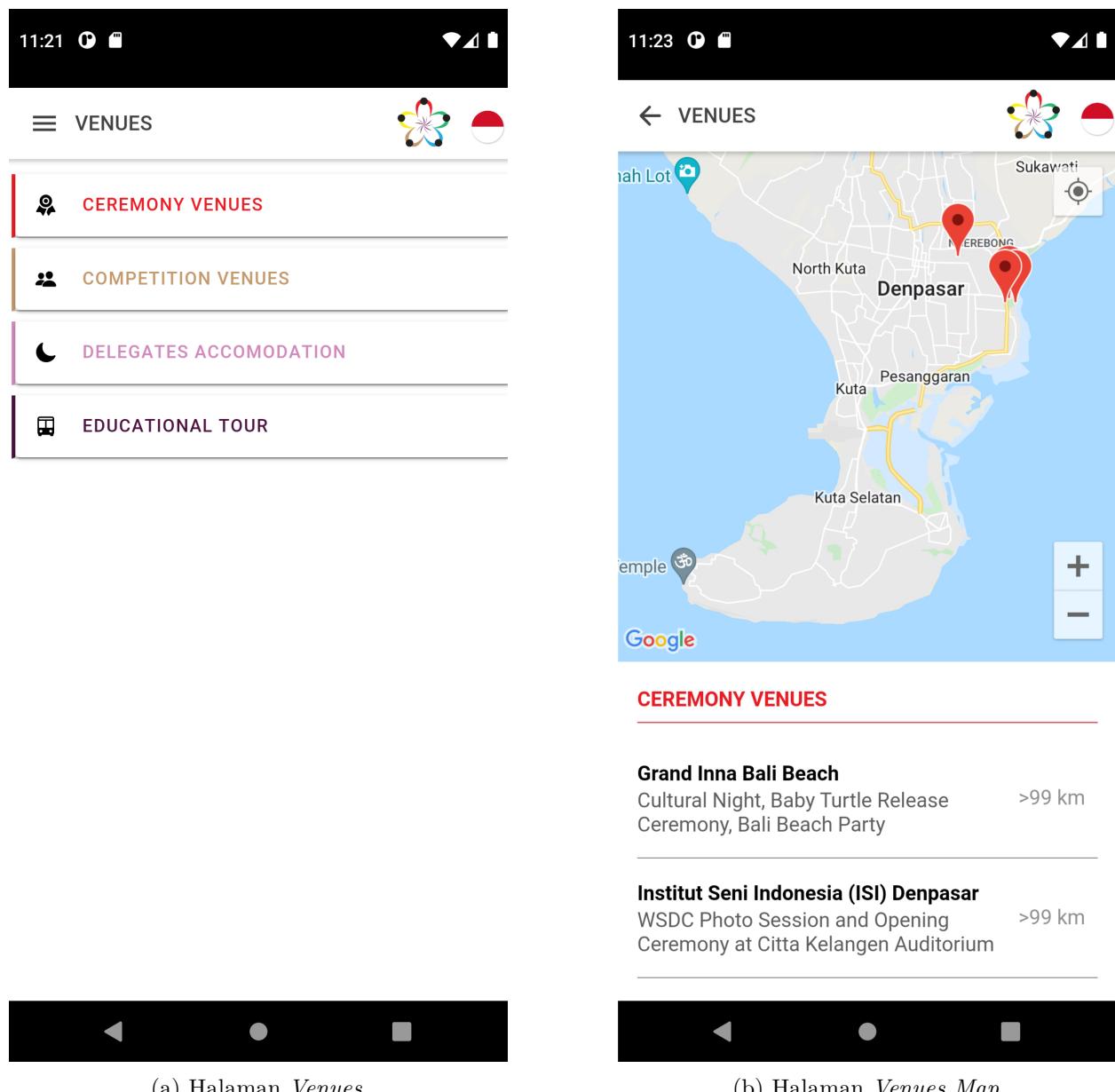
14:00 **23:59** **Hotel Check-In**
Lobby at Sanur Paradise Plaza Hotel

17:30 **21:00** **Dinner**
Griya Agung Ballroom at Sanur Paradise Plaza Hotel



(b) Halaman *Schedule*

Gambar 5.4: Tangkapan Layar Aplikasi WSDC 2017 Bali



Gambar 5.5: Tangkapan Layar Aplikasi WSDC 2017 Bali

1 5.2 Pengujian

2 5.2.1 Pengujian Fungsional

- 3 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 5.1 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	Sidebar terbuka menampilkan menu	Sesuai
3	Pengguna melakukan swipe dari kiri layar ke kanan layar	Sidebar terbuka menampilkan menu	Sesuai
4	Pengguna memilih menu Announcements pada Sidebar	Aplikasi menampilkan halaman Announcements	Sesuai
5	Pengguna memilih menu Home pada Sidebar	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 Sidebar	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 Sidebar	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
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 Sidebar	Aplikasi menampilkan halaman Draw	Sesuai
18	Pengguna memilih menu Result pada Sidebar	Aplikasi menampilkan halaman Result	Sesuai
19	Pengguna memilih menu Info pada Sidebar	Aplikasi menampilkan halaman Info	Sesuai
20	Pengguna menekan nomor telepon pada halama info	Aplikasi mengarahkan pengguna ke aplikasi panggilan	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. Kemudian, pengguna diminta untuk mengisi beberapa pertanyaan terkait pengalaman
⁵ menggunakan aplikasi WSDC 2017 Bali melalui Google Form. Berikut ini merupakan pertanyaan
⁶ dan rangkuman jawaban dari hasil pengujian eksperimental terhadap sembilan responden sebagai
⁷ berikut:

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

⁹ Seorang responden menjawab versi Android 5.1, seorang menjawab versi Android 8.0, seorang
¹⁰ menjawab versi Android 8.1, seorang menjawab versi Android 10, dan empat orang menjawab
¹¹ versi Android 11.

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

¹⁴ Semua responden menjawab aplikasi WSDC 2017 Bali terbaru menampilkan logo WSDC.

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

¹⁷ Semua responden menjawab tidak ada halaman yang tidak memiliki isi.

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

²⁰ Semua responden menjawab semua tombol berfungsi dengan normal.

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

²³ Sebanyak delapan responden menjawab tidak ada crash, forced close, atau kendala lain saat
²⁴ menggunakan aplikasi WSDC 2017 Bali terbaru, dan ada satu responden yang menjawab iya,
²⁵ namun tidak menjelaskan kendala apa yang terjadi.

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

²⁸ Dua orang responden berpendapat bahwa tampilan *sidemenu* tampak lebih segar dan menarik.
²⁹ Lalu sebanyak satu responden berpendapat bahwa tampilan *icon* terlihat lebih beragam
³⁰ dan menarik. Kemudian sebanyak empat responden berpendapat bahwa aplikasi WSDC
³¹ 2017 Bali terbaru dapat dibuka dengan lebih cepat dibandingkan dengan aplikasi terdahulu.
³² Lalu sebanyak satu responden berpendapat bahwa perubahan aplikasi menjadi lebih baik
³³ dari sebelumnya, satu responden menjawab perubahan yang terjadi hanya sedikit, dan satu
³⁴ responden menjawab tidak ada perubahan positif yang dirasakan.

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

³⁷ Sebanyak empat orang responden menjawab bahwa tidak ada perubahan negatif pada aplikasi
³⁸ WSDC 2017 Bali terbaru. Seorang responden menjawab *font* tulisan lebih kaku, dan halaman
³⁹ *draw* kualitasnya terlihat lebih rendah dibandingkan aplikasi terdahulu. Lalu seorang responden
⁴⁰ menjawab tampilan pada aplikasi yang terbaru dari menu yang ada di masing-masing *Venues*
⁴¹ sedikit aneh, karena nama tempat dan alamatnya saling berdekatan tanpa ada jarak spasi.
⁴² Dan seorang responden menjawab pemakaian aplikasi terbaru lebih boros.

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

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

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

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

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

11 Terdapat beberapa kritik dan saran dari responden sebagai berikut:

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

¹

BAB 6

²

KESIMPULAN DAN SARAN

³ 6.1 Kesimpulan

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

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

¹⁰ 6.2 Saran

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

- ¹³ 1. Dapat memilih font yang lebih menarik dan nyaman untuk dibaca.
- ¹⁴ 2. Dapat memperbaiki halaman *Result* agar memiliki tampilan lebih menarik.
- ¹⁵ 3. Dapat mengecilkan ukuran aplikasi.
- ¹⁶ 4. Pada halaman *venues* dimana tulisan headline dari list map dijauhkan sedikit dari tulisan
¹⁷ body karena terlalu dekat.

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] 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.
- [4] Emmit A. Scott, J. (2015) *SPA Design and Architecture: Understanding single-page web applications*, 1st edition. Manning Publications, New York, USA.
- [5] World Schools Debate Championship (2021) WSDC. <https://wsdcdebate.org/history>. [Online; diakses 8-Juli-2021].
- [6] Moiseev, A. dan Fain, Y. (2018) *Angular Development with TypeScript*, 2nd edition. Manning Publications, New York, USA.
- [7] Kunz, G. (2018) *Mastering Angular Components: Build component-based user interfaces using Angular*, 2nd edition. Pact Publishing Ltd., Birmingham, UK.
- [8] Griffith, C. (2017) *Mobile App Development with Ionic : Cross-Platform Apps with Ionic, Angular and Cordova*, 1st edition. O'Reilly Media, Inc., California, USA.
- [9] Grønli, T.-M., Biørn-Hansen, A., dan Majchrzak, T. A. (2019) Median trajectories using well-visited regions and shortest pathssoftware 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.
- [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: announcement.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: announcement.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 {
12   @ViewChild('drawIFrame') drawIFrame: ElementRef;
13   constructor(private storage: Storage, public loadingController: LoadingController
14   ) { }
15
16   ionViewDidLoad() {
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 { Browser } 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(){ //runs when the page has fully entered and is now the active
21     page.
22     SplashScreen.hide()
23   }
24
25   ngOnInit(): void { //called after Angular has initialized all data-bound
26     properties of a directive
27     this.storage.get('wsdcDataStorage').then((data) => {
28
29       if(data == null){
30         //Default from asset
31         this.http.get('../assets/json/wsdc_data.json').subscribe((data: any) => {
32           this.wsdcData = data;
33           this.storage.set('wsdcDataStorage',data);
34         },
35         error => {
36           this.showToast('Failed to refresh information from local storage');
37         });
38       }else{
39         this.wsdcData = data;
40       }
41
42       // Refresh data
43       setTimeout(() => {
44         this.http.get('http://wsdc.dnartworks.com/wsdc_data.json')
45         .subscribe((data: any) => {
46           this.storage.set('wsdcDataStorage', data);
47           this.wsdcData = data;
48         },
49         error => {
50           this.showToast('Failed to refresh information');
51         });
52       }, 1000);
53     })
54
55   doRefresh(refresher) {
56     this.http.get('https://wsdc.dnartworks.com/wsdc_data.json').subscribe((data:
57     any) => {
58       this.storage.clear();
```

```

58     this.storage.set('wsdcDataStorage', data);
59     this.wsdcData = data;
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(wsdData?.announcements[0].localtime) }}</h3>
24             <p>{{ wsdData?.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 wsdNews of wsdData?.newsletters;">
40         <div class="newsletters" >
41             
42             <h2 text-wrap>{{wsdNews.title}}</h2> <br>
43             <ion-button class="ion-padding-end" full block color="danger" (click)="launch(wsdNews.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   wsdInfoData: any;
13
14   constructor(private storage: Storage) {
15
16     this.storage.get('wsdcDataStorage').then((data) => {
17       this.wsdInfoData = data.info;
18       this.wsdInfoData = this.wsdInfoData.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{
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     this.storage.get('wsdcDataStorage').then((val) => {
```

```

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

```

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()">
27         <ion-slide *ngFor="let schedule of schedules;">

```

```

25      <ion-list>
26          <ion-item *ngFor="let agenda of schedule.agenda;" >
27              <ion-note item-start>
28                  {{agenda.start}} <br>
29                  {{agenda.end}}
30          </ion-note>
31          <ion-label class="ion-text-wrap">
32              <h3>{{agenda.title}}</h3>
33              <p>{{agenda.subtitle}}</p>
34          </ion-label>
35      </ion-item>
36  </ion-list>
37 </ion-slide>
38 </ion-slides>
39 </div>
40 </div>
41 </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{
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         this.storage.get('wsdcDataStorage').then((val) => {
16             this.valVenues = val.venues;
17             this.venuesData = [];
18             let currColorIdx: number = 1;
19             for (let venue of this.valVenues) {
20                 this.venuesData.push({
21                     id: venue.id,
22                     name: venue.name,
23                     icon: venue.icon,
24                     geojson: venue.geojson,
25                     colorIdx: currColorIdx,
26                 });
27                 if (currColorIdx > 4) {
28                     currColorIdx = 1;
29                 } else {
30                     currColorIdx++;
31                 }
32             }
33         })
34     }
35     itemTapped(wsdcVenue) {
36         // this.router.navigate(['venues-map', id])
37         let navigationExtras: NavigationExtras = {
38             state: {

```

```

39     venuesData: wsdcVenue
40   }
41 }
42 this.router.navigate(['venues-map'], navigationExtras);
43 }
44 }
```

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{
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 //Get data from venues page.
26 this.items = [];
27 this.activatedRoute.queryParams.subscribe(params => {
28     if (this.router.getCurrentNavigation().extras.state) {
29         this.venuesPage = this.router.getCurrentNavigation().extras.state.
30         venuesData;
31     }
32
33     this.items.push({
34         id: this.venuesPage.id,
35         idcolor: this.venuesPage.colorIdx
36     });
37
38     document.getElementById("pagetitle").style.color = this.pageTitleColors[this
39     .items[0].idcolor-1];
40 });
41
42 // Select data from storage
43 this.storage.get('wsdcDataStorage').then((data) => {
44     this.venuesMapsDetail = data.venues;
45     this.venuesMapsDetail = this.venuesMapsDetail.filter(d=> d.id==this.items
46 [0].id);
47 })
48
49 //save user lat & lng location
50 const printCurrentPosition = async () => {
51     const coordinates = await Geolocation.getCurrentPosition();
52     this.userCoordinatesLat = coordinates.coords.latitude;
53     this.userCoordinatesLng = coordinates.coords.longitude;
54 };
55
56 printCurrentPosition();
57 Geolocation.requestPermissions();
58 }
59
60 ionViewDidEnter() {
61     // create map and add marker
62     const initializeMap = async () => {
63         this.itemCounter = 0;
64         await CapacitorGoogleMaps.initialize({
65             key: "YOUR_IOS_MAPS_API_KEY",
66             devicePixelRatio: window.devicePixelRatio,
67         });
68         const element = document.getElementById("mapContainer");
69         const boundingRect = element.getBoundingClientRect();
70         try {
71             const result = await CapacitorGoogleMaps.createMap({
72                 boundingRect: {
73                     width: Math.round(boundingRect.width),
74                     height: Math.round(boundingRect.height),
75                     x: Math.round(boundingRect.x),
76                     y: Math.round(boundingRect.y),
77                 },
78                 cameraPosition: {
79                     target:{ //Kuta, Bali -8.722396, 115.17671
80                         latitude:-8.722396,
81                         longitude: 115.17671
82                     },
83                     zoom:11
84                 },
85                 preferences: {
86                     controls: {
87                         isCompassButtonEnabled:true,
```

```

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

```

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

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
20   <ion-list>
21     <ion-item *ngFor="let venue of venuesMapsDetail; let i=index">
22       <div>
23         <div class="venuesDescContainer" *ngFor="let properties of venue.geojson.
features; let j = index">
24           <div class="venueDesc" (click)="featTapped(properties.geometry.
coordinates)">
```

```
25      <h2>{{properties.properties.Name}}</h2>
26      <p>{{properties.properties.Description}}</p>
27    </div>
28    <div class="venuesDist" #distance>
29      <p>{{userDistanceTo [j]}}</p>
30    </div>
31    </div>
32  </div>
33  </ion-item>
34 </ion-list>
35 </ion-content>
```

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