

## **SKRIPSI**

**PEMBUATAN ULANG APLIKASI WSDC (*WORLD SCHOOL DEBATING CHAMPIONSHIP*) 2017 BALI DENGAN IONIC 6**



**Rajasa Cikal Maulana Solihin**

**NPM: 2017730084**

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



**UNDERGRADUATE THESIS**

**RE-CREATION OF WSDC (WORLD SCHOOL DEBATING  
CHAMPIONSHIP) 2017 BALI APP WITH IONIC 6**



**Rajasa Cikal Maulana Solihin**

**NPM: 2017730084**

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



## ABSTRAK

*World Schools Debating Championships* (WSDC) merupakan sebuah turnamen debat Bahasa Inggris tahunan untuk tim-tim tingkat sekolah menengah yang mewakili berbagai negara. Pada tahun 2017, WSDC diselenggarakan di Bali, Indonesia. Untuk menunjang acara tersebut, diciptakan sebuah aplikasi WSDC 2017 Bali yang memiliki beberapa fitur seperti melihat jadwal acara, melihat pengumuman acara, melihat lokasi dan peta lokasi acara, dan fitur notifikasi. Aplikasi tersebut dibangun menggunakan Ionic Framework versi 3 dengan Cordova dan diimplementasikan menggunakan *framework* JavaScript Angular serta dapat digunakan pada perangkat mobile berbasis Android dan iOS. Karena pada saat skripsi ini dibuat, Ionic Framework versi 3 sudah tidak lagi dikembangkan, dan aplikasi WSDC 2017 Bali sudah diturunkan dari App Store pada perangkat iOS karena tidak mendapat pembaruan sejak lama, maka dari itu dibuatlah aplikasi pembaruan aplikasi WSDC 2017 Bali menggunakan Ionic Framework versi 6.

Aplikasi WSDC 2017 Bali dengan Ionic Framework 6 diimplementasikan dengan menggunakan *framework* JavaScript Angular. Ionic Framework memiliki UI Component, yaitu berupa tag khusus yang digunakan untuk menambah fungsionalitas aplikasi, contohnya tag <ion-button> digunakan untuk menambah tombol. Ionic Framework dapat menggunakan fitur *native* dari perangkat dengan menggunakan Native Api, yaitu Capacitor. Capacitor memiliki *plugin* yang digunakan untuk mengakses fitur *native* pada perangkat mobile, seperti *Geolocation* dan *In App Browser*. *Plugin* yang digunakan pada aplikasi ini yaitu: Geolocation API, Google Maps API, Splash Screen, dan Browser API. Dengan digunakannya Capacitor maka aplikasi WSDC 2017 Bali yang ditulis menggunakan bahasa pemrograman web seperti HTML, CSS, dan JavaScript, dapat dijalankan pada perangkat *mobile* dengan sistem operasi Android dan iOS. Namun pada skripsi ini hanya akan diuji pada perangkat Android.

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

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



## ABSTRACT

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

The WSDC 2017 Bali application with Ionic Framework 6 is implemented using the Angular JavaScript framework. The Ionic Framework has a UI Component, which is a special tag that is used to add application functionality, for example the `<ion-button>` tag is used to add buttons. Ionic Framework can use the native features of the device by using the Native API, i.e. Capacitor. Capacitor has plugins that are used to access native features on mobile devices, such as Geolocation and In App Browser. The plugins used in this application are: Geolocation API, Google Maps API, Splash Screen, and Browser API. By using Capacitor, the WSDC 2017 Bali application, which is written using web programming languages such as HTML, CSS, and JavaScript, can be run on mobile devices with Android and iOS operating systems. However, this thesis will only be tested on Android devices.

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

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



# DAFTAR ISI

<b>DAFTAR ISI</b>	<b>ix</b>
<b>DAFTAR GAMBAR</b>	<b>xi</b>
<b>1 PENDAHULUAN</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	3
1.3 Tujuan . . . . .	3
1.4 Batasan Masalah . . . . .	3
1.5 Metodologi . . . . .	3
1.6 Sistematika Pembahasan . . . . .	3
<b>2 LANDASAN TEORI</b>	<b>5</b>
2.1 WSDC 2017 Bali . . . . .	5
2.2 Angular . . . . .	6
2.3 Ionic Framework . . . . .	13
2.3.1 Native API . . . . .	13
2.3.2 UI Component . . . . .	20
2.3.3 Migrasi Ionic 3 ke Ionic 6 . . . . .	27
<b>3 ANALISIS</b>	<b>37</b>
3.1 Analisis Sistem Kini dan Sistem Usulan . . . . .	37
3.2 Tantangan Pengembangan Sistem Usulan . . . . .	76
<b>4 PERANCANGAN</b>	<b>77</b>
4.1 Perancangan Kelas . . . . .	77
4.2 Perancangan Struktur HTML . . . . .	85
<b>5 IMPLEMENTASI DAN PENGUJIAN</b>	<b>89</b>
5.1 Implementasi . . . . .	89
5.1.1 Lingkungan Implementasi . . . . .	89
5.1.2 Hasil Implementasi . . . . .	89
5.2 Pengujian . . . . .	96
5.2.1 Pengujian Fungsional . . . . .	96
5.2.2 Pengujian Eksperimental . . . . .	97
<b>6 KESIMPULAN DAN SARAN</b>	<b>99</b>
6.1 Kesimpulan . . . . .	99
6.2 Saran . . . . .	99
<b>DAFTAR REFERENSI</b>	<b>101</b>
<b>A KODE PROGRAM</b>	<b>103</b>
A.1 Komponen Announcements . . . . .	103

A.2 Komponen Draw . . . . .	104
A.3 Komponen Home . . . . .	106
A.4 Komponen Info . . . . .	108
A.5 Komponen Result . . . . .	109
A.6 Komponen Schedule . . . . .	110
A.7 Komponen Venues . . . . .	112
A.8 Komponen Venues Map . . . . .	113

## DAFTAR GAMBAR

2.1	Aplikasi WSDC 2017 Bali Saat Ini pada Perangkat Android . . . . .	5
2.2	Aplikasi WSDC 2017 Bali Saat Ini pada Perangkat Android . . . . .	6
2.3	Desain <i>server-side</i> tradisional [1] . . . . .	7
2.4	Desain SPA [1] . . . . .	8
2.5	Proses Pengemasan Aplikasi Cordova [2] . . . . .	19
3.1	<i>Use Case Diagram</i> Aplikasi WSDC 2017 Bali . . . . .	37
3.2	Komponen Announcements pada Aplikasi WSDC 2017 Bali . . . . .	42
3.3	Komponen Draw pada Aplikasi WSDC 2017 Bali . . . . .	46
3.4	Komponen Home pada Aplikasi WSDC 2017 Bali . . . . .	48
3.5	Komponen Info pada Aplikasi WSDC 2017 Bali . . . . .	54
3.6	Komponen <i>Result</i> pada Aplikasi WSDC 2017 Bali . . . . .	56
3.7	Komponen <i>Schedule</i> pada Aplikasi WSDC 2017 Bali . . . . .	58
3.8	Komponen <i>Venues</i> pada Aplikasi WSDC 2017 Bali . . . . .	63
3.9	Komponen <i>Venues Map</i> pada Aplikasi WSDC 2017 Bali . . . . .	67
3.10	Penggunaan Community Google Maps dan Geolocation pada Aplikasi WSDC 2017 Bali . . . . .	72
3.11	Penggunaan Community Google Maps pada Aplikasi WSDC 2017 Bali . . . . .	73
4.1	Diagram Kelas Keseluruhan . . . . .	77
5.1	Tangkapan Layar Halaman Splash Screen Aplikasi WSDC 2017 Bali . . . . .	91
5.2	Tangkapan Layar Sidemenu Aplikasi WSDC 2017 Bali . . . . .	91
5.3	Tangkapan Layar Halaman Home Aplikasi WSDC 2017 Bali . . . . .	92
5.4	Tangkapan Layar Halaman <i>Announcements</i> Aplikasi WSDC 2017 Bali . . . . .	92
5.5	Tangkapan Layar Halaman <i>Draw</i> Aplikasi WSDC 2017 Bali . . . . .	93
5.6	Tangkapan Layar Halaman <i>Info</i> Aplikasi WSDC 2017 Bali . . . . .	93
5.7	Tangkapan Layar Halaman <i>Result</i> Aplikasi WSDC 2017 Bali . . . . .	94
5.8	Tangkapan Layar Halaman <i>Schedule</i> Aplikasi WSDC 2017 Bali . . . . .	94
5.9	Tangkapan Layar Halaman <i>Venues</i> Aplikasi WSDC 2017 Bali . . . . .	95
5.10	Tangkapan Layar Halaman <i>Venues Map</i> Aplikasi WSDC 2017 Bali . . . . .	95



## BAB 1

### PENDAHULUAN

#### 3    1.1 Latar Belakang

4    *World Schools Debating Championships* (WSDC) merupakan sebuah turnamen debat Bahasa Inggris tahunan  
5    untuk tim-tim tingkat sekolah menengah yang mewakili berbagai negara [3]. Pada awalnya, kompetisi  
6    universitas dunia diselenggarakan di Sydney pada bulan Juli 1988. Anggota Federasi Debat Australia  
7    menyadari bahwa tidak ada acara serupa untuk siswa sekolah menengah. Namun kejuaraan universitas dunia  
8    ini menunjukkan potensi yang sangat besar untuk kompetisi debat internasional yang melibatkan siswa dari  
9    seluruh dunia. Pada tahun 1991, kejuaraan diadakan di Edinburgh. Dan sejak saat itu nama *World Schools*  
10   *Debating Championships* digunakan dan berlangsung hingga saat ini.

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

17      Ionic Framework merupakan sebuah *framework open source* lintas platform yang digunakan untuk  
18   mengembangkan aplikasi *hybrid* yang bekerja pada berbagai macam platform seluler seperti Android, iOS,  
19   dan Windows [4]. Aplikasi *hybrid* merupakan sebuah WebApp yang berjalan di dalam sebuah *WebView*  
20   namun dapat menggunakan fitur-fitur yang disediakan oleh perangkat *mobile* [5]. Dalam kata lain, aplikasi  
21   *hybrid* merupakan sebuah aplikasi *native* yang menampilkan *web app* di dalam *web view*. Perbedaan *web app*  
22   dengan aplikasi *native* adalah dalam hal penggunaan perangkat keras pada perangkat mobile, dimana aplikasi  
23   *native* dapat dengan bebas mendapatkan akses penuh penggunaan perangkat keras sedangkan *web app* tidak.

24      Salah satu aplikasi *hybrid* yaitu Apache Cordova yang dikembangkan oleh Adobe digunakan untuk  
25   memecahkan masalah dimana pada saat mengembangkan suatu aplikasi seluler, setiap vendor perangkat  
26   lunak memiliki alat-alat yang unik yang hanya dimiliki oleh vendor tersebut, yaitu *Software Development Kit*  
27   (*SDK*) [5]. Karena *SDK* yang digunakan berbeda-beda tergantung kepada jenis sistem operasi perangkat  
28   seluler, maka tidak dapat membuat aplikasi untuk platform yang berbeda, namun dengan baris kode yang sama.  
29   Apache Cordova sebagai aplikasi *hybrid* dapat membuat aplikasi berbasis web seperti *HTML*, *Cascading*  
30   *Style Sheets* (*CSS*), dan *Javascript*, dikemas sebagai aplikasi *native* yang dapat mengakses fitur-fitur perangkat  
31   keras dari suatu perangkat.

1     Ionic Framework mendukung komunikasi dengan menggunakan Native API, yaitu pengembangan  
2 aplikasi langsung terintegrasi ke dalam platform [6]. Cordova merupakan Native API yang digunakan  
3 untuk menambahkan fungsionalitas ke dalam aplikasi Ionic apapun. Selain Cordova, terdapat Native API  
4 lain yang didukung oleh Ionic Framework, yaitu Capacitor. Capacitor merupakan penerus dari Cordova  
5 yang menyediakan akses ke perangkat *native* dan fitur platform, serta untuk menyediakan satu set API  
6 untuk mengembangkan aplikasi seluler secara *hybrid*, *Progressive Web Apps* berbasis web, dan aplikasi  
7 komputer berbasis Electron [7]. Ionic juga memiliki berbagai macam *front-end library* dan *User Interface*(UI)  
8 *Components* berupa *tag* khusus yang digunakan untuk menambah fungsionalita aplikasi.

9     Pada Ionic 5 ke atas, terdapat beberapa *framework Javascript* yang dapat diimplementasikan menggunakan  
10 *framework* Ionic, yaitu:

- Angular

12     Angular pada awalnya diciptakan oleh karyawan Google, Misko Hevert dan Adam Abrons pada tahun  
13 2008, yang masih bernama AngularJS dan dikembangkan dengan JavaScript [8]. Pada saat AngularJS  
14 pertama kali diciptakan, sebagian besar situs web menggunakan aplikasi multi-halaman, yaitu ketika  
15 pengguna mengklik tautan, maka browser harus mengambil dokumen HTML yang diminta dari server.  
16 Angular tidak mengimplementasi hal tersebut, melainkan menggunakan *Single-page Application*  
17 (SPA), yaitu ketika halaman awal dimuat, semua yang dibutuhkan untuk membuat dan menampilkan  
18 sebuah halaman diunduh, kemudian ditampilkan kedalam layar. Dengan begitu, *browser* tidak perlu  
19 mengunduh ulang yang dibutuhkan saat menampilkan halaman [1].

- React

21     React adalah *library* JavaScript *open source* untuk membangun antarmuka pengguna, dikelola oleh  
22 Facebook, dapat digunakan dalam berbagai skenario termasuk aplikasi iOS dan Android [8].

- Vue

24     Vue merupakan *framework* progresif untuk membangun antarmuka pengguna untuk web, yang dapat  
25 digunakan baik untuk projek kecil dan untuk *Single-Page Applications* (SPAs) [8].

26     Aplikasi WSDC 2017 Bali yang dibangun pada tahun 2017 oleh PT DNArtworks Komunikasi Visual  
27 untuk perangkat iOS untuk sistem operasi iOS telah diturunkan dari App Store dikarenakan tidak mengalami  
28 pembaruan dalam jangka waktu tertentu. Maka dari itu diperlukan pembaruan pada aplikasi WSDC 2017  
29 Bali. Pembaruan tersebut dilakukan dengan memperbarui versi Ionic Framework yang sebelumnya versi  
30 3, menjadi versi 6. Pembaruan Ionic Framework diperlukan karena Ionic versi 3 sudah tidak mendapat  
31 dukungan lagi dari tim pengembang Ionic Framework. Untuk melakukan pembaruan tersebut, maka pada  
32 skripsi ini dibuat sebuah aplikasi WSDC 2017 Bali baru, yang merupakan sebuah pembaruan dari aplikasi  
33 WSDC 2017 Bali. Pembaruan dilakukan dengan cara membuat aplikasi WSDC 2017 Bali baru menggunakan  
34 *framework* Ionic versi 6 dan Capacitor dengan fitur-fitur dan halaman yang sama seperti aplikasi sebelumnya.  
35 Dilakukan juga penyesuaian pembaruan yang terjadi pada saat melakukan pembaruan dari Ionic 3 ke Ionic 6  
36 serta pembaruan pada saat mengganti Cordova dengan Capacitor.

## 1 1.2 Rumusan Masalah

- 2 Rumusan masalah yang dibahas pada skripsi ini yaitu:
  - 3 1. Bagaimana melakukan migrasi aplikasi Android WSDC 2017 Bali ke *framework* Ionic versi 6?
  - 4 2. Bagaimana menjalankan aplikasi WSDC 2017 Bali pada perangkat Android setelah dilakukan migrasi?

## 5 1.3 Tujuan

- 6 Tujuan yang ingin dicapai dari penulisan skripsi ini yaitu:
  - 7 1. Melakukan migrasi aplikasi Android WSDC 2017 Bali ke *framework* Ionic versi 6.
  - 8 2. Menjalankan aplikasi WSDC 2017 Bali pada perangkat Android setelah dilakukan migrasi.

## 9 1.4 Batasan Masalah

- 10 Dalam skripsi ini dibuat batasan-batasan masalah dalam pembuatan perangkat lunak. Batasan-batasan  
11 masalah yang ditetapkan adalah sebagai berikut:
  - 12 1. Aplikasi ini tidak akan memiliki fitur notifikasi, dikarenakan sudah tidak terdapat *service* dari Ionic  
13 dan tidak dikembangkan lagi dari sisi servernya.
  - 14 2. Walaupun Ionic Framework mendukung pengembangan aplikasi untuk sistem operasi iOS, aplikasi  
15 hanya akan diuji pada *platform mobile* berbasis android.

## 16 1.5 Metodologi

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

## 24 1.6 Sistematika Pembahasan

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

1     **4. Bab Perancangan**

2       Bab 4 berisi perancangan aplikasi meliputi perancangan kelas beserta dengan diagram kelas, deskripsi  
3       kelas dan fungsinya, serta perancangan struktur HTML.

4     **5. Bab Implementasi dan Pengujian**

5       Bab 5 berisi implementasi dan pengujian aplikasi meliputi lingkungan implementasi, hasil implemen-  
6       tasi, pengujian fungsional, dan pengujian eksperimental.

7     **6. Bab Kesimpulan dan Saran**

8       Bab 6 berisi kesimpulan dari hasil pembangunan aplikasi ini dan saran untuk pengembangan selanjut-  
9       nya.

1

## BAB 2

2

## LANDASAN TEORI

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

### 6 2.1 WSDC 2017 Bali

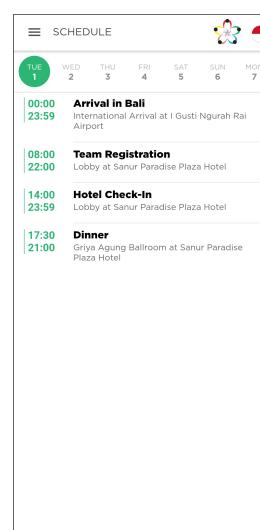
- 7 Aplikasi WSDC 2017 Bali digunakan untuk menunjang keberlangsungan acara WSDC 2017 yang diselenggarakan di Bali, Indonesia. Aplikasi WSDC 2017 Bali dapat diunduh untuk sistem operasi *android* melalui URL  
8 <https://play.google.com/store/apps/details?id=org.wsdc2017indonesia.app>. Aplikasi ini dibangun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



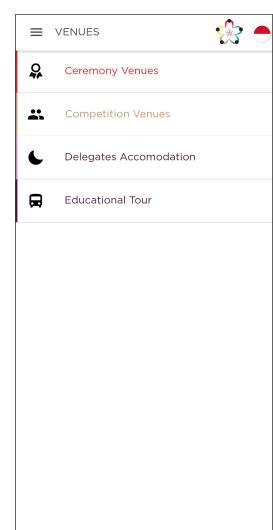
(a) Home page



(b) Announcements Page



(c) Schedule Page

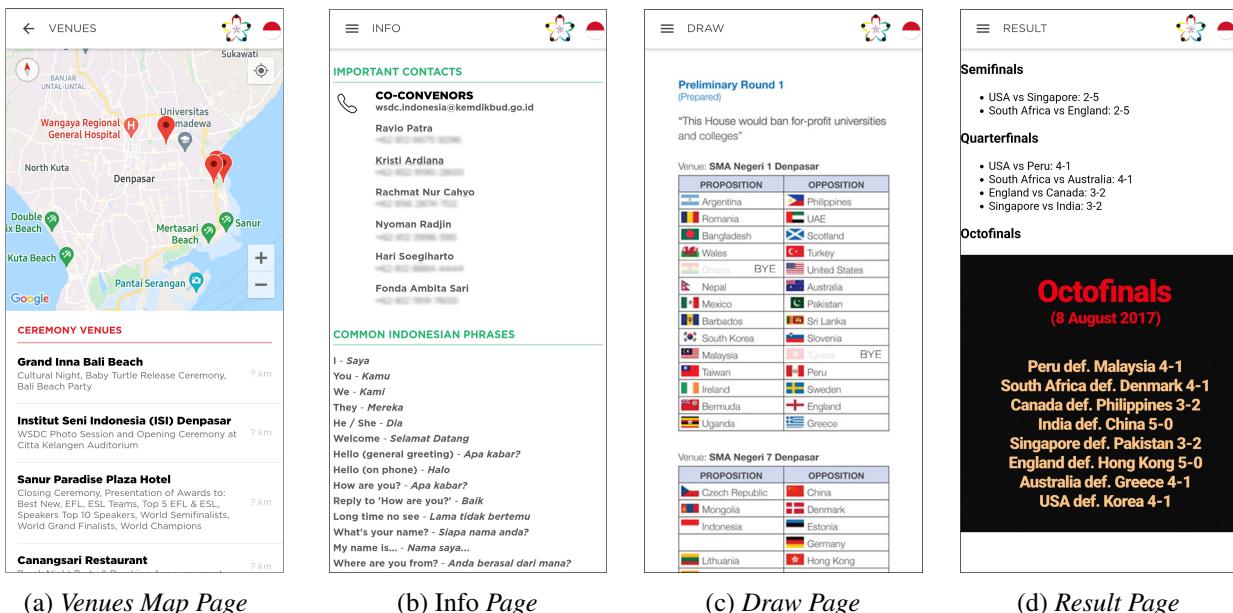


(d) Venues Page

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

Fitur-fitur yang terdapat di aplikasi WSDC 2017 Bali saat ini yaitu :

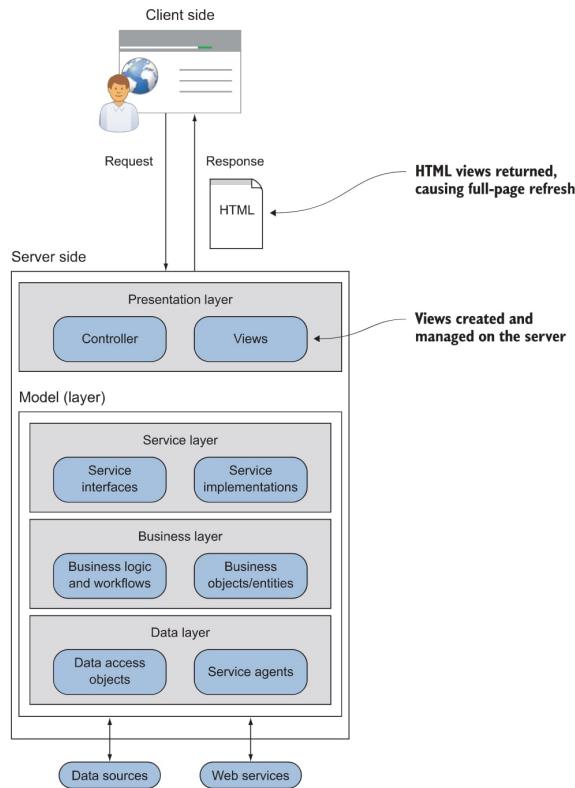
1. *Home Page*: Pengguna dapat melihat pengumuman terbaru, dan *headline* dari berita-berita terkait acara WSDC 2017 Bali dengan tombol yang dapat diklik untuk melihat berita tersebut secara lebih detail (Gambar 2.1a).
2. *Announcements*: Pengguna dapat melihat pemberitahuan tentang berjalannya acara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



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

## 2.2 Angular

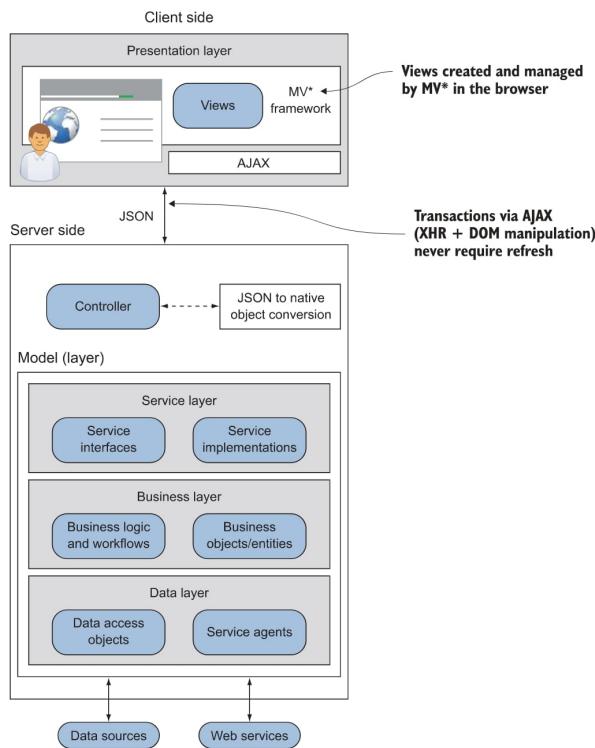
Angular merupakan *framework* Javascript terbuka yang dikembangkan oleh Google, dan merupakan penerus dari versi sebelumnya yaitu AngularJS [9]. Angular versi pertama dirilis pada tahun 2016 dengan nama Angular 2. Aplikasi Angular dapat dibangun dengan menggunakan JavaScript, atau TypeScript.



Gambar 2.3: Desain *server-side* tradisional [1]

1       Angular dapat digunakan untuk membuat aplikasi Single-page-application (SPA), yaitu ketika halaman  
 2       awal dimuat, semua yang dibutuhkan untuk membuat dan menampilkan sebuah halaman diunduh, kemudian  
 3       ditampilkan kedalam layar [1]. Website dengan desain SPA berbeda dengan desain tradisional. Seperti  
 4       pada Gambar 2.3, pada desain tradisional setiap permintaan untuk tampilan baru, yaitu halaman HTML,  
 5       memerlukan *request* ke server dan mengembalikannya kembali ke *client*. Ketika data baru diperlukan  
 6       oleh *client*, permintaan dikirim ke sisi server. Di sisi server, permintaan diambil oleh *controller* di dalam  
 7       *presentation layer*.

8       *Controller* kemudian berinteraksi dengan *model layer* melalui *service layer* untuk menentukan data yang  
 9       diperlukan. Permintaan terhadap data tersebut diteruskan ke *data layer* yang kemudian mengambil data dari  
 10      *web service*. Setelah data diambil, setiap perubahan yang diperlukan pada data kemudian dibuat oleh *business*  
 11      *logic* di *business layer*. Lalu setelah perubahan yang diperlukan pada data telah selesai di *business logic*,  
 12      data dikembalikan ke *presentation layer*. Di dalam *presentation layer* menentukan bagaimana data yang  
 13      baru diperoleh direpresentasikan ke dalam tampilan yang dipilih. Setelah data dan tampilan digabungkan,  
 14      tampilan dikembalikan ke browser. Browser kemudian menerima halaman HTML setelah melakukan *refresh*  
 15      pada tampilan antarmuka *browser*. Pada akhirnya pengguna melihat tampilan baru yang berisi data yang  
 16      diminta setelah *browser* melakukan *refresh*.



Gambar 2.4: Desain SPA [1]

Pada desain SPA, pengelolaan tampilan antarmuka dan server terpisah seperti pada Gambar 2.4. Hal tersebut membuat desain SPA hampir sama dengan desain tradisional, namun terdapat perbedaan berupa tidak ada lagi *refresh* pada *browser* untuk mendapatkan tampilan halaman dengan data yang baru, dan *presentation layer* berada di *client* yang membuat tugas menggabungkan HTML dan data dipindahkan dari server ke browser. Pada desain SPA, setelah halaman awal dimuat, semua data yang diperlukan untuk membuat dan menampilkan tampilan diunduh dan siap digunakan. Jika diperlukan tampilan baru, tampilan tersebut dibuat secara lokal di browser dan ditampilkan secara dinamis melalui JavaScript sehingga tidak diperlukan *refresh* pada browser.

Angular dapat dibangun dengan menggunakan JavaScript atau TypeScript. Namun, penggunaan TypeScript saat ini menjadi lebih produktif dibandingkan dengan JavaScript [9]. TypeScript mengikuti perkembangan terakhir dari ECMAScript, yaitu bahasa skrip yang distandarisasi oleh Ecma International dalam spesifikasi ECMA-262 dan ISO/IEC 16262 [10]. TypeScript juga menambahkan *types*, *interface*, *decorators*, *class member variables*, *generic*, *enum*, dan *keyword* seperti *public*, *protected*, dan *private* serta dapat digunakan untuk mendeklarasikan jenis tipe khusus yang dapat dikustomisasi. Maka dari itu, Framework Angular sendiri ditulis menggunakan TypeScript.

Angular terdiri dari komponen-komponen yang merupakan sebuah penyusun utama untuk aplikasi Angular. Setiap komponen yang ada pada aplikasi, secara *default* memiliki *critical files* yang terdiri dari *file* HTML untuk mendeklarasi halaman yang akan dimuat, *file* TypeScript, serta *file* css [11]. Di dalam komponen terdapat *module.ts* yang merupakan NgModule dari sebuah komponen, *spec.ts* yang digunakan untuk menguji komponen, dan *routing.module.ts* yang berisi definisi untuk bernavigasi antar bagian dalam aplikasi Angular. Penjelasan untuk masing-masing *file* adalah sebagai berikut:

- *File routing.module.ts*

1     File ini digunakan untuk melakukan navigasi antar komponen. Untuk melakukannya, harus melakukan  
 2     *import* RouterModule dan Routes sehingga dapat memiliki fungsionalitas *routing* (Kode 2.1). Pada  
 3     kode tersebut, dilakukan *import* RouterModule dan Routes dari @angular/router. Selanjutnya lakukan  
 4     *import* untuk komponen lain yang akan dituju untuk melakukan navigasi ke komponen tersebut.  
 5     Sebagai contoh, pada kode 2.1 melakukan *import* untuk komponen “Heroes“.

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

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

10    Setelah itu, lakukan konfigurasi *routes*. Routes kemudian memberi tahu Router tampilan mana yang  
 11    akan ditampilkan saat pengguna melakukan navigasi. Seperti pada kode 2.2, dengan menggunakan  
 12    key path dan component. Path digunakan untuk melakukan navigasi di alamat url sesuai dengan path  
 13    tersebut. Sedangkan component digunakan untuk menampilkan komponen yang dituju.

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

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

- *File CSS*

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

- *File HTML*

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

- *File specs.ts*

27    File ini digunakan untuk melakukan pengujian terhadap aplikasi Angular.

- *File TypeScript*

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

```
1 @Component({
2   ...
3 })
4 export class AppComponent {
5   ...
6 }
```

Kode 2.3: Anotasi Komponen dengan *Decorator*

42    Pada Angular terdapat decorator @Component yang digunakan untuk memanggil HTML *template*  
 43    dan CSS untuk komponen. Pada setiap komponen terdapat HTML *template* yang berada di dalam  
 44    komponen tersebut, atau di dalam file yang berada di luar file komponen yang direferensikan dengan

1 menggunakan properti **templateUrl** di dalam komponen pada *file* TypeScript. File HTML *template*  
 2 yang terpisah dengan komponen, memberikan efek kode yang lebih bersih di dalam komponen nya.  
 3 Selain HTML *template*, file lain seperti file CSS dapat diletakan terpisah dari file komponen dengan  
 4 menggunakan styleUrls untuk mereferensikan file CSS ke dalam komponen (Kode 2.4). Lalu terdapat  
 5 juga properti selector untuk mendefinisikan nama dari tag yang bisa digunakan atau dipanggil oleh  
 6 komponen lain.

```
7
8 @Component({
9   selector: 'app-search',
10  templateUrl: './search.component.html',
11  styleUrls:[ './search.component.css' ]
12 })
13 export class SearchComponent{
14   ...
15 }
```

Kode 2.4: Contoh Mereferensikan HTML dan CSS di Dalam Komponen

17 • *File* module.ts

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

```
24
25 @NgModule({
26   declarations: [
27     AppComponent
28   ],
29   imports: [
30     BrowserModule
31   ],
32   bootstrap: [AppComponent]
33 })
34 export class AppModule{
35   ...
36 }
```

Kode 2.5: Module dengan Komponen

38 Di dalam aplikasi Angular, terdapat beberapa komponen, seperti Parent Component, dan Child Com-  
 39 ponent. Parent Component dapat mengirimkan data ke properti Child Component-nya. Namun, Child  
 40 Component tidak tahu siapa yang mengirimkannya. Sedangkan Child Component juga dapat mengirimkan  
 41 data ke Parent Component, meskipun dia tidak tahu siapa Parent Component-nya. Arsitektur seperti ini dapat  
 42 membuat komponen menjadi mandiri dan dapat digunakan kembali.

Angular menyediakan beberapa *libraries*, yaitu:

### 1. Angular Router

Angular Router digunakan untuk menangani navigasi dari satu tampilan ke tampilan lain, dan kemudian menampilkannya di layar [12]. Langkah-langkah untuk membuat *route* pada Angular adalah sebagai berikut:

#### (a) Import RouterModule dan Routes ke routing module.

Pada tahap ini, Angular CLI membuatnya secara otomatis saat pembuatan proyek Angular. Angular CLI membuat array Routes, dan melakukan import serta export array di `@NgModule()`. Seperti pada contoh kode 2.6 pada baris ke-2 dilakukan import untuk Routes dan RouterModule dari Angular, serta pada baris ke-4 Angular CLI membuat array Routes.

```

1 import { NgModule } from '@angular/core';
2 import { Routes, RouterModule } from '@angular/router';
3
4 const routes: Routes = [];
5
6 @NgModule({
7   imports: [RouterModule.forRoot(routes)],
8   exports: [RouterModule]
9 })
10 export class AppRoutingModule { }
```

Kode 2.6: Contoh *Routing* pada Angular

#### (b) Menambahkan *routes* di array Routes.

Tahap selanjutnya adalah memasukan *route* pada array routes yang telah dibuat pada tahap sebelumnya. Seperti pada kode 2.7, terdapat properti path dan component pada setiap elemen di array. Properti path digunakan untuk path URL untuk route yang dituju. Properti component digunakan untuk menunjukkan komponen apa yang digunakan. Sebagai contoh pada kode 2.7 baris ke-2, properti path berisi komponen 'first-component' yang komponennya berada di FirstComponent pada properti component. Ketika pengguna memnulis path 'first-component' (seperti pada properti path) di dalam URL, maka komponen FirstComponent pada properti component yang akan dipanggil yang kemudian menampilkan komponen first-component.

```

1 const routes: Routes = [
2   { path: 'first-component', component: FirstComponent },
3   { path: 'second-component', component: SecondComponent },
4 ];
```

Kode 2.7: Contoh *Routing* pada Angular

#### (c) Menambahkan Routes ke Aplikasi.

Selanjutnya, untuk mengakses halaman sesuai dengan *routes*-nya, yaitu dengan memanggil properti path pada tahap sebelumnya di dalam routerLink seperti pada kode 2.8 baris ke-4. RouterLink digunakan untuk membuka komponen ketika pengguna melakukan klik. Setelah pengguna melakukan klik, maka routerLink akan memanggil routes dengan path first-component dan mengembalikan komponen FirstComponent.

```

1 <h1>Angular Router App</h1>
2 <nav>
```

```

1      3   <ul>
2      4     <li><a routerLink="/first-component" routerLinkActive="active"
3       5       ariaCurrentWhenActive="page">First Component</a></li>
4      6     <li><a routerLink="/second-component" routerLinkActive="active"
5       7       ariaCurrentWhenActive="page">Second Component</a></li>
6      8   </ul>
7 </nav>
8 <router-outlet></router-outlet>

```

Kode 2.8: Contoh *Routing* pada Angular

## 2. Angular HttpClient.

Angular HttpClient digunakan untuk melakukan komunikasi dengan server dengan menggunakan protokol HTTP, baik itu melakukan *download* atau *upload* data. Untuk dapat menggunakan HttpClient, pertama harus melakukan import pada AppModule di file app.module.ts. Seperti pada kode 2.9, pada baris ke-3 dilakukan import HttpClientModule. Kemudian lakukan import pada @NgModule seperti pada baris ke-8.

```

1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3 import { HttpClientModule } from '@angular/common/http';
4
5 @NgModule({
6   imports: [
7     BrowserModule,
8     HttpClientModule,
9   ],
10  declarations: [
11    AppComponent,
12  ],
13  bootstrap: [ AppComponent ]
14 })
15 export class AppModule {}

```

Kode 2.9: Contoh *Routing* pada Angular

HttpClient memiliki sebuah *method* `get()` yang digunakan untuk mengambil data dari server. *Method* ini mengirim HTTP Request, dan mengembalikan sebuah data JSON yang ada pada *response body*. Kode 2.10 adalah contoh untuk mengambil data dari server dengan menggunakan HttpClient. Pada kode tersebut, terdapat *method* `subscribe()` seperti pada baris ke-4 yang membuat HttpClient menulis dan mengirim HTTP Request ke server yang kemudian mendapatkan data untuk digunakan di dalam aplikasi.

```
1 const testData: Data = {name: 'Test Data'};  
2  
3 httpClient.get<Data>(testUrl)  
4 .subscribe(data =>  
5   expect(data).toEqual(testData)  
6 );  
7  
8 const req = httpTestingController.expectOne('/data');  
9  
10 expect(req.request.method).toEqual('GET');  
11  
12 req.flush(testData);  
13  
14 httpTestingController.verify();  
15  
16
```

Kode 2.10: Contoh *Routing* pada Angular

## 17 2.3 Ionic Framework

18 Ionic Framework merupakan sebuah *framework open source* lintas platform yang memungkinkan untuk  
19 mengembangkan aplikasi hibrida yang bekerja pada berbagai macam platform seluler seperti *android*, *iOS*,  
20 dan *Windows* [4]. Ionic memiliki berbagai macam *front-end library* dan komponen *User Interface*(UI) yang  
21 digunakan untuk perancangan aplikasi menggunakan teknologi web seperti *HTML*, *CSS*, dan *Javascript*,  
22 dengan integrasi untuk berbagai *framework* seperti *Angular*, *React*, dan *Vue*. Saat pertama kali dibuat, Ionic  
23 menggunakan *AngularJS*. Namun, pada saat *Angular* versi 2 yang menggunakan *TypeScript* dirilis, Ionic  
24 versi 2 dan selanjutnya menggunakan *Angular*. Pada tahun 2019, Ionic mendukung penggunaan *framework*  
25 lain selain *Angular*, yaitu *React* dan *Vue*. Di dalam Ionic, *Angular* digunakan untuk membangun aplikasi  
26 dan perutean, sehingga aplikasi dapat sejalan dengan ekosistem *Angular* lainnya. Ionic menyediakan *toolkit*  
27 *Angular* untuk membangun aplikasi dan terintegrasi dengan *Angular CLI* resmi yang menyediakan fitur  
28 khusus untuk aplikasi Ionic *Angular*. Pada saat skripsi ini dibuat, Ionic versi terbaru adalah Ionic versi 6,  
29 dengan dukungan penggunaan *Angular* versi 12 sampai dengan yang lebih baru.

### 30 2.3.1 Native API

31 Native API memungkinkan pengembangan aplikasi langsung terintegrasi ke dalam platform. Pengembang  
32 dapat membuat aplikasi pada perangkat *mobile* untuk dapat diimplementasikan ke berbagai *platform*, seperti  
33 *iOS* dan *Android*, setelah pengembangan selesai di dalam *framework native* tanpa perlu perubahan, dan tidak  
34 mempengaruhi performa dari aplikasi tersebut [6].

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

5 **2.3.1.1 Capacitor**

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

11 Capacitor digunakan untuk membuat aplikasi secara *hybrid* yang memungkinkan pembuatan aplikasi  
12 seluler untuk beberapa platform dengan menggunakan baris kode yang sama [13]. Karena aplikasi  
13 yang dibangun dengan Ionic Framework merupakan sebuah aplikasi web (WebApp), maka dari itu untuk  
14 menjalankan WebApp tersebut secara langsung dari platform mobile dibutuhkan sebuah penghubung antara  
15 WebApp dengan perangkat, yaitu Capacitor. Capacitor menjadi sebuah *native bridge* yang menjembatani  
16 antara WebView dengan siklus hidup terakhir pada program yaitu Runtime.

17 Aplikasi WebApp menjalankan URL, kemudian WebView akan ditampilkan sama seperti saat membuka  
18 *browser*, tapi dengan Capacitor maka WebApp tersebut memiliki akses ke berbagai fitur *native* perangkat  
19 seperti kamera, notifikasi, GPS, dan file sistem. Untuk mengakses fitur-fitur tersebut dibutuhkan plugin.  
20 Capacitor memiliki *plugins* untuk mengakses fitur-fitur tersebut. *Plugins* yang dimiliki oleh Capacitor terbagi  
21 menjadi dua, yaitu:

22 1. *Official Plugins*

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

26 (a) Browser API

27 Browser API menyediakan kemampuan untuk membuka browser dalam aplikasi. Untuk menginstal  
28 Browser API dapat dilakukan melalui *command line* dengan perintah seperti pada kode 2.11.

29  
30 1 npm install @capacitor/browser  
31 2 npx cap sync

Kode 2.11: Kode untuk Instalasi Browser API

33 Browser API memiliki beberapa *method*, yaitu:

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

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

4       (b) Geolocation API

5       Geolocation API menyediakan *method* untuk mendapatkan dan melacak posisi perangkat saat ini  
 6       menggunakan *Global Positioning System* (GPS), dengan latitude dan longitude, juga informasi  
 7       mengenai ketinggian, arah, dan kecepatan jika tersedia. Untuk menginstal Geolocation API dapat  
 8       dilakukan melalui *command line* dengan perintah seperti pada kode 2.12.

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

Kode 2.12: Kode untuk Menginstal Geolocation API

11       Pada perangkat Android, dibutuhkan izin untuk menggunakan Geolocation API 2.13. Izin tersebut  
 12       ditambahkan ke dalam baris kode pada *file* *AndroidManifest.xml*. Izin tersebut digunakan untuk  
 13       meminta data lokasi (*fine* dan *coarse*), dan izin untuk menggunakan GPS. Dengan menggunakan  
 14       GPS, Geolocation API akan mengembalikan koordinat dari perangkat yang berupa latitude dan  
 15       longitude. Contoh dari penggunaan Geolocation API untuk mengambil koordinat perangkat  
 16       tertera pada kode 2.14.

```
17      1 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
18      2 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
19      3 <uses-feature android:name="android.hardware.location.gps" />
```

Kode 2.13: *Permissions* Geolocation API pada Android

```
20      1 import { Geolocation } from '@capacitor/geolocation';
21
22      2
23      3 const printCurrentPosition = async () => {
24        const coordinates = await Geolocation.getCurrentPosition();
25
26        6 console.log('Current position:', coordinates);
27      7};
```

Kode 2.14: *Permissions* Geolocation API pada Android

28       Untuk mengambil posisi dari perangkat dengan menggunakan *method* *getCurrentPosition()*  
 29       seperti pada kode 2.14. Selain itu terdapat beberapa *method* lain yang dapat diimplementasikan,  
 30       diantaranya yaitu:

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

7 (c) Splash Screen API

8 Splash Screen API digunakan sebagai penyedia *method* untuk menampilkan atau menyembunyikan  
9 gambar *splash*. Untuk menginstal Splash Screen API dapat dilakukan melalui *command line*  
10 dengan perintah seperti pada kode 2.15. Contoh penggunaan Splash Screen dapat dilihat pada  
11 kode 2.16

```
16 1 npm install @capacitor/splash-screen
17 2 npx cap sync
```

Kode 2.15: Kode untuk Menginstal Splash Screen API

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

Kode 2.16: Contoh Kode Penggunaan Splash Screen API

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

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

1    2. *Community Plugins*

2    *Community Plugins* merupakan sekumpulan *plugin* yang diciptakan oleh komunitas untuk menambahkan fungsionalitas ke dalam aplikasi. Perbedaan dengan *Official Plugins* yaitu tim Capacitor tidak secara resmi melakukan pemeliharaan terhadap *Community Plugins*. Salah satu *plugin* yang diciptakan oleh komunitas adalah *plugin Google Maps* yang menggunakan *native Google Maps SDK*. Untuk menginstal *plugin Google Maps* dapat dilakukan melalui *command line* dengan perintah seperti pada kode 2.17.

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

Kode 2.17: Kode untuk Menginstal *Plugin Google Maps*

12 Maps SDK merender *native element* (*MapView*) di belakang aplikasi web (WebApp) membutuhkan  
 13 *boundaries* yang dirender. Maka dari itu Maps SDK menggunakan *native element* dibandingkan  
 14 *HTMLElement*. Sebelum dapat menggunakan *plugin* ini, harus dilakukan impor terlebih dahulu seperti  
 15 pada kode 2.18. Setelah mengimpor *plugin*, sebuah instance Maps sederhana dapat diinisialisasi seperti  
 16 pada kode 2.19. Pada contoh kode tersebut, Capacitor Community Google Maps memiliki sebuah  
 17 *function* *createMap* yang digunakan untuk membuat peta Google Maps. Di dalam *function* tersebut  
 18 terdapat sebuah *option* *boundingRect* yang digunakan sebagai lokasi untuk menyimpan peta Google  
 19 Maps. Karena google maps ditempatkan di dalam sebuah kontainer pada HTML, maka dari itu pada  
 20 *option* dibutuhkan lokasi *width*, *height*, *x*, dan *y*. Lokasi tersebut diambil dari kontainer yang dibuat  
 21 pada HTML yang sebelumnya sudah didapatkan pada baris ke-10. Kemudian nilai tersebut dimasukan  
 22 ke dalam *option* *boundingRect* pada baris ke-11 sebagai lokasi dari Google Maps.  
 23

```
24 1 import { CapacitorGoogleMaps } from "@capacitor-community/google-maps";
```

Kode 2.18: Kode untuk Import *Plugin Google Maps*

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

```

```

1   22 } catch (e) {
2   23 alert("Map failed to load");
3   24 }
4   25 };
5   26
6   27 (function () {
7   28 initializeMap();
8   29 })();

```

Kode 2.19: Contoh Kode Penggunaan *Plugin Google Maps*

10 Capacitor dapat menambahkan platform *native* ke dalam proyek Ionic, baik untuk sistem operasi Android  
11 seperti pada kode 2.20 maupun iOS seperti pada kode 2.21. Setelah itu, Capacitor akan melakukan instalasi  
12 *package* platform Capacitor, dan menyalin *template native* platform ke dalam proyek. Kemudian, Capacitor  
13 juga dapat melakukan pembuatan aplikasi *native* dengan menyalin aset web ke dalam *native* platform dengan  
14 perintah seperti pada kode 2.22 untuk perangkat berbasis iOS dan perintah pada kode 2.23 untuk perangkat  
15 berbasis Android. Setelah perintah tersebut dieksekusi, maka Capacitor akan membuka IDE dari proyek  
16 *native*, seperti Xcode untuk perangkat berbasis iOS dan Android Studio untuk perangkat berbasis Android.  
17 Dengan begitu, maka aplikasi dapat berjalan secara *native* di dalam perangkat terkait.

```

18 ionic capacitor add android
20

```

Kode 2.20: Kode untuk Menambahkan Platform Android dengan Capacitor

```

21 ionic capacitor add ios
23

```

Kode 2.21: Kode untuk Menambahkan Platform iOS dengan Capacitor

```

24 ionic capacitor build ios
26

```

Kode 2.22: Kode untuk Membuat Aplikasi Capacitor Untuk Perangkat iOS

```

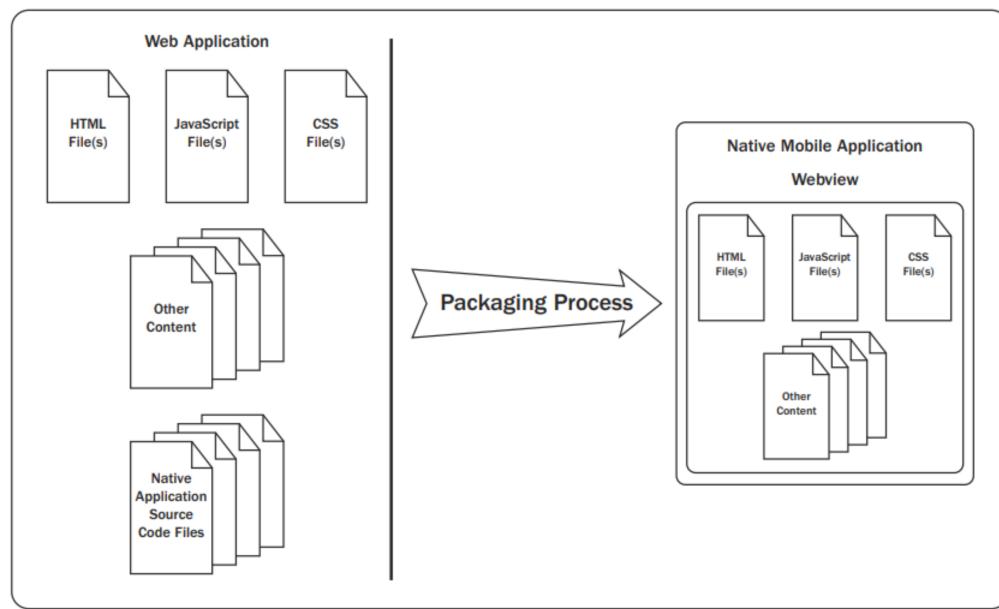
27 ionic capacitor build android
29

```

Kode 2.23: Kode untuk Membuat Aplikasi Capacitor Untuk Perangkat Android

### 30 2.3.1.2 Cordova

31 Cordova merupakan *framework open source* yang dapat membuat pengembang untuk menggunakan teknologi seperti HTML, JavaScript, dan CSS untuk membangun aplikasi untuk perangkat bergerak yang  
32 dapat berjalan pada beberapa sistem operasi *mobile* [14]. Cordova menyediakan antarmuka antara WebView dan lapisan *native* pada perangkat [6]. Selain dapat bekerja pada dua platform seluler Android dan  
33 iOS, Cordova juga dapat digunakan pada platform seluler seperti Windows Phone, Blackberry, dan FireOS.  
34 Cordova digunakan untuk membangun aplikasi seluler lintas platform dan mengimplementasikannya  
35 sebagai kombinasi aplikasi web dan aplikasi *native*, yaitu aplikasi Hybrid [2]. Aplikasi dibuat dengan  
36 menggunakan teknologi web seperti HTML dan CSS yang dinamakan WebApp. Cordova mengemas aplikasi  
37 web ke dalam *native container*, seperti yang diilustrasikan pada Gambar 2.5.



Gambar 2.5: Proses Pengemasan Aplikasi Cordova [2]

1 Di dalam aplikasi Cordova, antarmuka pengguna aplikasi terdiri dari satu layar yang hanya berisi  
 2 satu tampilan web. Saat aplikasi dijalankan, aplikasi memuat halaman awal WebApp ke dalam tampilan  
 3 web. WebApp yang berjalan di dalam *native container* sama seperti aplikasi web lainnya yang berjalan di  
 4 dalam browser web pada perangkat *mobile*. WebApp dapat membuka halaman HTML, menjalankan logika  
 5 JavaScript, dan mengimplementasi CSS, namun WebApp tidak dapat menjalankan fitur-fitur perangkat keras  
 6 dan perangkat lunak dari perangkat *mobile* seperti akselerometer, kamera, data kontak, dan fitur lainnya.  
 7 Cordova menyediakan berbagai JavaScript API yang dapat digunakan agar aplikasi web yang berjalan  
 8 di dalam *native container* dapat mengakses kemampuan perangkat di luar kemampuan *browser*. API ini  
 9 diimplementasikan ke dalam JavaScript *libraries* yang menjalankan fitur-fitur perangkat ke aplikasi web  
 10 dengan implementasi yang disesuaikan dengan sistem operasi yang digunakan.

11 Cordova membuat aplikasi dapat mengakses fitur-fitur perangkat keras dan perangkat lunak suatu perang-  
 12 kat dengan menggunakan *plugin*. Framework Ionic telah terdapat berbagai macam TypeScript *wrapper* untuk  
 13 *plugins* Cordova. Untuk dapat menggunakan Cordova Plugins, yaitu dengan memasang Cordova Plugins  
 14 terlebih dahulu yang dapat dipasang dengan menjalankan Kode 2.24, dan memperbaruiinya ke versi terakhir  
 15 pada Kode 2.25 yang dapat dilakukan melalui CLI. Setiap *plugins* memiliki dua komponen, yaitu kode *native*  
 16 (Cordova), dan kode TypeScript (Ionic Native).

```
17
18 npm install cordova-plugin-name
19 npx cap sync
```

Kode 2.24: Kode untuk Memasang Cordova Plugins

```
21
22 npm install cordova-plugin-name@2
23 npx cap update
```

Kode 2.25: Kode untuk Memperbarui Cordova Plugins

1 Sama seperti Capacitor, Cordova memiliki *plugins* yang menyediakan antarmuka JavaScript ke komponen  
 2 *native*. Dengan menggunakan *plugin* Cordova, aplikasi dapat menjalankan fitur-fitur perangkat *mobile*  
 3 seperti kamera, geolokasi, dan file sistem. Salah satu *plugin* yang tersedia yaitu Google Maps. *Plugin* ini  
 4 menghasilkan MapView secara *native*, dan menempatkannya di bawah browser. MapView yang dihasilkan  
 5 bukan merupakan HTMLElements, maka tidak terkait dengan HTML. Sebelum dapat menggunakan *plugin*  
 6 Google Maps, dilakukan instalasi *plugin* untuk pertama kali pada *command line* 2.26. Selanjutnya atur  
 7 Google Maps API Key di dalam *file config.xml* 2.27.

```
8 cordova plugin add cordova-plugin-googlemaps
10
```

Kode 2.26: Kode untuk Menginstal *Plugin* Cordova Google Maps

```
11 <widget ...>
12   <preference name="GOOGLE_MAPS_ANDROID_API_KEY" value="(api key)" />
13   <preference name="GOOGLE_MAPS_IOS_API_KEY" value="(api key)" />
14 </widget>
16
```

Kode 2.27: Kode untuk Mengatur API Key untuk *Plugin* Cordova Google Maps

17 Cordova juga dapat menambahkan platform *native* ke dalam proyek Ionic dengan mengetikan perintah  
 18 seperti pada kode 2.28 untuk menambahkan platform Android, dan kode 2.29 untuk menambahkan platform  
 19 iOS. Untuk menjalankan proyek Ionic dengan Cordova dengan mengetikan perintah seperti pada kode 2.30  
 20 untuk perangkat Android, dan kode 2.31 untuk perangkat iOS. Dengan begitu, Cordova akan membuka IDE  
 21 sesuai dengan perintah yang dijalankan.

```
22 ionic cordova platform add android
24
```

Kode 2.28: Kode untuk Menambahkan Platform Android dengan Cordova

```
25 ionic cordova platform add ios
27
```

Kode 2.29: Kode untuk Menambahkan Platform iOS dengan Cordova

```
28 ionic cordova platform run android
30
```

Kode 2.30: Kode untuk Membuat Aplikasi Cordova Untuk Perangkat Android

```
31 ionic cordova platform run ios
33
```

Kode 2.31: Kode untuk Membuat Aplikasi Cordova Untuk Perangkat iOS

### 34 2.3.2 UI Component

35 Framework Ionic dapat menggunakan kemampuan Angular dalam memperluas kosakata HTML, yaitu  
 36 menyertakan *tag* khusus untuk menciptakan seluruh rangkaian komponen [6]. Semua komponen memiliki  
 37 awalan ion, sehingga dapat dikenali dalam markup. Sama seperti *tag* HTML standar, komponen Ionic  
 38 juga dapat menerima berbagai macam atribut sebagai pengaturan dari *tag* tersebut, seperti mengatur id atau  
 39 mendefinisikan kelas CSS tambahan. Terdapat beberapa komponen yang ada pada *framework* Ionic versi  
 40 6 [1]. Komponen-komponen tersebut yaitu:

1     • Action Sheet

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

6     • Alert

7       Alert merupakan dialog yang menampilkan informasi kepada pengguna, atau mengumpulkan informasi  
 8       dari pengguna menggunakan input. Alert muncul di atas konten aplikasi, dan harus ditutup secara  
 9       manual oleh pengguna sebelum pengguna dapat melanjutkan interaksi dengan aplikasi. Secara opsional,  
 10      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 karakter  
 13      lain, yang digunakan sebagai pemberitahuan bahwa ada item tambahan yang terkait dengan suatu  
 14      elemen dan menunjukkan berapa banyak item yang ada. Penggunaan Badge dengan menggunakan tag  
 15      <ion-badge> (Kode 2.32).

16    1    <ion-badge>99</ion-badge>

Kode 2.32: Potongan Kode Program dari Badge Component

19    • Button

20      Merupakan elemen yang dapat diklik, biasanya digunakan dalam formulir atau di mana pun yang  
 21      membutuhkan fungsionalitas tombol. Button biasanya menampilkan teks, ikon, atau bisa juga keduanya.  
 22      Button dapat pula menggunakan atribut untuk menampilkannya dengan penampilan tertentu.  
 23      Penggunaan Button dengan menggunakan tag <ion-button> (Kode 2.33).

24    1    <ion-button>Default</ion-button>

Kode 2.33: Potongan Kode Program dari Button Component

27    • Card

28      Merupakan bagian standar dari tampilan antarmuka, yang tampak seperti kartu yang berfungsi sebagai  
 29      titik masuk ke dalam informasi yang lebih detail. Card dapat menjadi satu komponen, tetapi sering kali  
 30      terdiri dari beberapa header, judul, sub judul, dan konten. Penggunaan Card dengan menggunakan tag  
 31      <ion-card> yang dapat berisi *header*, *subtitle*, *title*, dan *content* (Kode 2.34).

32    1    <ion-card>  
 33    2      <ion-card-header>  
 34    3        <ion-card-subtitle>Card Subtitle</ion-card-subtitle>  
 35    4        <ion-card-title>Card Title</ion-card-title>  
 36    5      </ion-card-header>  
 37    6  
 38    7      <ion-card-content>  
 39    8        Card Content  
 40    9      </ion-card-content>  
 41    10     </ion-card>

Kode 2.34: Potongan Kode Program dari Card Component

1 • Content

2 Komponen content merupakan penyedia area konten yang bisa digunakan untuk mengontrol area  
 3 yang dapat digulir. Dalam satu tampilan, setidaknya terdapat satu buah content. Content juga dapat  
 4 dimodifikasi padding, margin, dan lainnya menggunakan *global style* yang berada di CSS Utilites atau  
 5 mengubahnya secara individual dengan menggunakan CSS. Penggunaan Content dengan menggunakan  
 6 *tag <ion-content>* (Kode 2.35).

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

Kode 2.35: Potongan Kode Program dari Content Component

20 • Date and Time Pickers

21 Datetime merupakan penampil antarmuka untuk pengguna memilih tanggal dan waktu. Terdapat kolom  
 22 yang dapat digulir yang dapat digunakan untuk memilih tahun, bulan, hari, jam, dan menit secara  
 23 individual. Komponen ini menampilkan nilai di dua tempat, yaitu di komponen <ion-datetime>  
 24 (Kode 2.49), dan di antarmuka pemilih yang ditampilkan dari bawah layar.

```
25 1 <ion-datetime displayFormat = "MM DD YY" placeholder = "Select Date" ></ion-datetime>
```

Kode 2.36: Kode Program dari Datetime Component dengan Format Bulan-Hari-Tahun

28 • Grid

29 Grid digunakan untuk membuat tata letak kustom pada tampilan. Grid terdiri dari tiga bagian, yaitu  
 30 *grid*, baris, dan kolom. Masing-masing kolom dapat diubah ukurannya menggunakan CSS. Grid  
 31 digunakan dengan *tag <ion-grid>* (Kode 2.37).

```
32 1 <ion-row>
 2   <ion-col size = "6">
 3     ion-col [size = "6"]
 4   </ion-col>
 5   <ion-col>
 6     ion-col
 7   </ion-col>
 8 </ion-row>
```

Kode 2.37: Potongan Kode Program dari Grid Component

1     • Infinite Scroll

2       Komponen Infinite Scroll memanggil sebuah action yang akan dilakukan ketika pengguna menggulir  
3       dengan jarak tertentu dari bawah atau atas halaman. Penggunaan Infinite Scroll dengan menggunakan  
4       tag `<ion-infinite-scroll>` (Kode 2.38).

```
5     1 <ion-infinite-scroll threshold="100px" (ionInfinite)="loadData($event)">
6       2   <ion-infinite-scroll-content
7       3     loadingSpinner="bubbles"
8       4     loadingText="Loading more data...">
9       5   </ion-infinite-scroll-content>
10      6 </ion-infinite-scroll>
```

Kode 2.38: Potongan Kode Program dari Infinite Scroll Component

13    • Icon

14       Icon merupakan komponen yang berupa gambar kecil, yang merepresentasikan sebuah berkas, dan fo-  
15       lder di dalam aplikasi. Penggunaan Icon adalah dengan menggunakan tag `<ion-icon>` (Kode 2.39).

```
16     1 <ion-icon name="home"></ion-icon>
```

Kode 2.39: Potongan Kode Program dari Icon Home

19    • Item

20       Item merupakan elemen yang dapat berisi teks, ikon, avatar, gambar, masukan, dan elemen asli atau  
21       kustom lainnya. Biasanya, item ditempatkan di dalam sebuah *list* bersamaan dengan item lainnya  
22       dengan tag `<ion-item>` (Kode 2.40). Dapat dilakukan *swipe*, dihapus, disusun ulang, dan diedit.

```
23     1 <ion-item>
24       2   <ion-label>
25       3     Item
26       4   </ion-label>
27       5 </ion-item>
```

Kode 2.40: Potongan Kode Program dari Item Component

30    • List

31       Komponen List terdiri dari beberapa baris *item* yang dapat berisi teks, tombol, *toggles*, ikon, thumbnails,  
32       dan komponen-komponen lainnya menggunakan tag `<ion-list>` (Kode 2.41). List mendukung  
33       untuk pengguna melakukan *swiping*, *dragging*, dan penghapusan *item*.

```
34     1 <ion-list>
35       2   <ion-item>
36       3     <ion-label>Pokemon Yellow</ion-label>
37       4   </ion-item>
38       5   <ion-item>
39       6     <ion-label>Mega Man X</ion-label>
40       7   </ion-item>
41       8   <ion-item>
42       9     <ion-label>The Legend of Zelda</ion-label>
43       10  </ion-item>
44       11  <ion-item>
45       12    <ion-label>Pac-Man</ion-label>
46       13  </ion-item>
```

```

1 14 <ion-item>
2 15   <ion-label>Super Mario World</ion-label>
3 16 </ion-item>
4 17 </ion-list>

```

Kode 2.41: 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 2.42). Pada dasarnya, Menu muncul dari kiri, tetapi sisi kemunculan menu dapat diganti.

```

10 1 <ion-menu side="start" menuId="first" contentId="main">
11 2   <ion-header>
12 3     <ion-toolbar color="primary">
13 4       <ion-title>Start Menu</ion-title>
14 5     </ion-toolbar>
15 6   </ion-header>
16 7   <ion-content>
17 8     <ion-list>
18 9       <ion-item>Menu Item</ion-item>
19 10      <ion-item>Menu Item</ion-item>
20 11      <ion-item>Menu Item</ion-item>
21 12      <ion-item>Menu Item</ion-item>
22 13      <ion-item>Menu Item</ion-item>
23 14     </ion-list>
24 15   </ion-content>
25 16 </ion-menu>

```

Kode 2.42: 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 2.43).

```

34 35 <ion-modal [isOpen]="true">
36 36   <ng-template>
37 37     <ion-content>Modal Content</ion-content>
38 38   </ng-template>
39 39 </ion-modal>

```

Kode 2.43: Kode Program dari Modal

1     • Navigation

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

8     • Refresher

9     Refresher merupakan komponen yang menyediakan fungsionalitas *pull-to-refresh* pada komponen  
 10   konten dengan tag `<ion-refresher>` (Kode 2.44). Refresher digunakan pada saat pengguna  
 11   menarik layar dari atas ke bawah, maka Refresher akan menyegarkan data atau mendapatkan daftar  
 12   data untuk mengambil lebih banyak data.

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

Kode 2.44: Kode Program dari Refresher

20    • Segment

21    Segment berfungsi untuk menampilkan pilihan tombol bagi pengguna untuk beralih di antara tampilan  
 22   berbeda di dalam satu halaman yang sama. Segment menampilkan sekelompok tombol-tombol yang da-  
 23   pat diklik, dalam baris horizontal. Penggunaan Segment dengan menggunakan tag `<ion-segment>`  
 24   (Kode 2.45).

```
26 <ion-segment (ionChange)="segmentChanged($event)">
27   <ion-segment-button value="friends">
28     <ion-label>Friends</ion-label>
29   </ion-segment-button>
30   <ion-segment-button value="enemies">
31     <ion-label>Enemies</ion-label>
32   </ion-segment-button>
33 </ion-segment>
```

Kode 2.45: Kode Program dari Segment

35    • Slides

36    Komponen Slides merupakan sebuah *multi-section container* yang setiap bagianya dapat dilakukan  
 37   *swipe* atau *drag*. Untuk menggunakan komponen ini dengan menggunakan tag `<ion-slides>` seperi-  
 38   ti pada kode 2.46 sebagai pembungkus slide. Masing-masing slide menggunakan tag `<ion-slide>`  
 39   seperti pada baris ke-2. Tag tersebut dapat digunakan untuk menggeser slide dari kiri ke kanan, atau  
 40   sebaliknya.

```

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

```

Kode 2.46: Kode Program dari Slides

- Tabs

Tabs merupakan navigasi *top-level* yang mengimplementasi sebuah *tab-based navigation*. Tabs dapat digunakan dengan tag `<ion-tabs>` (Kode 2.47) yang tidak memiliki *styling* apapun dan bekerja sebagai *router outlet* untuk menangani navigasi.

```

1 <ion-tabs>
2   <ion-tab-bar slot="bottom">
3     <ion-tab-button tab="schedule">
4       <ion-icon name="calendar"></ion-icon>
5       <ion-label>Schedule</ion-label>
6       <ion-badge>6</ion-badge>
7     </ion-tab-button>
8
9     <ion-tab-button tab="speakers">
10    <ion-icon name="person-circle"></ion-icon>
11    <ion-label>Speakers</ion-label>
12  </ion-tab-button>
13 </ion-tab-bar>
14 </ion-tabs>

```

Kode 2.47: Kode Program dari Tabs

- Toast

Komponen Toast merupakan sebuah notifikasi yang digunakan di aplikasi modern untuk memberikan umpan balik atau menampilkan pesan sistem. Komponen ini muncul diatas konten aplikasi, dan dapat ditutup untuk melanjutkan penggunaan aplikasi. Komponen ini menggunakan `ToastController` yang dimiliki oleh *library* Ionic Angular (Kode 2.48)

```

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

```

Kode 2.48: Kode Program dari Toast

1     • Toolbar

2     Toolbar dapat diposisikan di atas ataupun di bawah konten. Ketika toolbar ditempatkan di header <ion-header> akan muncul di bagian atas konten, sedangkan ketika ditempatkan di footer <ion-footer> akan muncul tetap di bagian bawah. Toolbar menggunakan tag <ion-toolbar>, yang di dalamnya dapat berisi button, dan dapat menggunakan border (Kode 2.49).

```
1 <ion-toolbar>
2   <ion-buttons slot="start">
3     <ion-back-button></ion-back-button>
4   </ion-buttons>
5   <ion-title>Back Button</ion-title>
6 </ion-toolbar>
```

Kode 2.49: Kode Program dari Toolbar dengan Button di Dalamnya

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

17 **2.3.3 Migrasi Ionic 3 ke Ionic 6**

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

21 1. Migrasi Ionic 3 ke Ionic 4

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

23 (a) Membuat Proyek Ionic Baru

24     Untuk membuat projek Ionic baru tanpa *template* apapun dengan menggunakan perintah **ionic start myApp blank** dan memilih Angular sebagai *frameworknya* 2.50.

```
26     1 ionic start myApp blank
```

Kode 2.50: Perintah Membuat Proyek Ionic Baru

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

31 (c) Menyalin *Root-level Items*

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

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

36 (e) Menyalin Bagian-bagian Aplikasi

37     Menyalin keseluruhan bagian yang ada pada aplikasi, baik itu halaman maupun fitur yang ada, 38 dengan ketentuan sebagai berikut :

- 39     • Shadow DOM sudah aktif secara *default*.

- 40     • Page atau Components Sass tidak lagi dibungkus dengan tag *page* atau *components* dan harus menggunakan opsi *styleUrls* milik Angular dari dekorator *@Component*.

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

- 1 • Perubahan RxJS yang digunakan. Pada ionic 3 adalah versi 5, sedangkan pada Ionic 4 RxJS  
2 yang digunakan adalah versi 6.
- 3 • Lifecycle Hooks tertentu harus digantikan dengan Angular Hooks.
- 4 • Perubahan markup yang mungkin saja dibutuhkan.

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

8     – *Button*

9 Terdapat perbedaan pada *tag* untuk membuat Button, yang semula pada Ionic 3 adalah  
10 <button> menjadi <ion-button> pada Ionic 4 (Kode 2.51).

```
11 1 <ion-button (click)="doSomething()">
12 2   Default Button
13 3 </ion-button>
```

Kode 2.51: Penggunaan Button pada Ionic 4

16     – Floating Action Button (FAB)

17 Terdapat perbedaan pada *tag* di dalam <ion-fab>, yang semula pada Ionic 3 adalah  
18 <button> menjadi <ion-fab-button> pada Ionic 4 (Kode 2.52).

```
19 1 <ion-fab>
20 2   <ion-fab-button>
21 3     <ion-icon name="add"></ion-icon>
22 4   </ion-fab-button>
23 5   <ion-fab-list>
24 6     <ion-fab-button>
25 7       <ion-icon name="logo-facebook"></ion-icon>
26 8     </ion-fab-button>
27 9   </ion-fab-list>
28 10 </ion-fab>
```

Kode 2.52: Penggunaan Floating Action Button pada Ionic 4

31     – Item

32 Terdapat perbedaan pada *tag* <ion-item>, dimana pada Ionic 3, *tag* <ion-label>  
33 akan secara otomatis ditambahkan ke dalam <ion-item>. Sedangkan pada Ionic 4  
34 diharuskan untuk menambahkan *tag* <ion-label> secara manual ke dalam komponen  
35 item (Kode 2.53).

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

Kode 2.53: Penggunaan Item pada Ionic 4

43     – Label

44 Pada Ionic 4, atribut untuk mengatur posisi dari label digabungkan dengan atribut  
45 *position* (Kode 2.54).

```

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

```

Kode 2.54: Penggunaan Atribut *Position* pada Ionic 4

7           – Menu

8           Terdapat beberapa perubahan nama pada Ionic 4, yaitu:

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

Properti	Perubahan	
	Ionic 3	Ionic 4
swipeEnabled	swipeEnabled	swipeGesture
content	content	contentId

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

Events	Perubahan	
	Ionic 3	Ionic 4
ionClose	ionClose	ionDidClose
ionOpen	ionOpen	ionDidOpen

13           – Nav

14           Terdapat perubahan Nav pada Ionic 4. Perubahan-perubahan tersebut adalah sebagai  
15          berikut:

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

Nama Method	Perubahan	
	Ionic 3	Ionic 4
remove	remove	getChildNavs
getActiveChildNavs	getActiveChildNavs	getChildNavs

18           \* Perubahan Nama Prop

19           Terdapat perubahan nama prop pada Ionic 4. Perubahan tersebut adalah sebagai  
20          berikut:

Nama Prop	Perubahan	
	Ionic 3	Ionic 4
swipeBackEnabled	swipeBackEnabled	swipeGesture

1            - Navbar

2            Pada Ionic 4, terdapat penghapusan terhadap komponen <ion-navbar> karena un-  
 3            tuk menjaga agar selalu menggunakan <ion-toolbar> dengan *back button* yang  
 4            eksplisit (Kode 2.55).

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

Kode 2.55: Penggunaan Navbar pada Ionic 4 dengan *Back Button*

13           - Overlays

14           Pada Ionic 4, semua overlay harus menggunakan *async/await*. Overlay tersebut adalah  
 15           Action Sheet, Alert, Loading, Modal, Popover, and Toast (Kode 2.56). Terjadi perubahan  
 16           nama properti enableBackdropDismiss menjadi backdropDismiss.

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

Kode 2.56: Penggunaan Overlay untuk Toast pada Ionic 4

26           - Scroll

27           Tag <ion-scroll> sudah dihapus pada Ionic 4 agar penggunaan tag <ion-content>  
 28           menjadi lebih maksimal.

29           - Segment Button

30           Pada Ionic 4, teks pada Segment Button membutuhkan <ion-label> untuk mem-  
 31           bungkusnya (Kode 2.57).

```
1 <ion-segment-button>
2   <ion-label>Item One</ion-label>
3 </ion-segment-button>
```

Kode 2.57: Penggunaan Segment Button untuk Toast pada Ionic 4

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

42           2. Migrasi Ionic 4 ke Ionic 5

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

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

1     • CSS

2       – *Activated, Focused, Hover States*

3       Kelas `.activated` secara otomatis ditambahkan ke komponen yang dapat diklik, mengalami  
 4       perubahan nama menjadi `.ion-activated`. Terdapat pembaruan komponen Action Sheet  
 5       sehingga variabel akan diawali dengan `button`. Hal ini dapat memungkinkan aplikasi tetap  
 6       memiliki kontrol atas `opacity` jika diinginkan, tetapi saat memperbarui status, hanya perlu  
 7       mengatur variabel utama, yaitu `-background-activated`, `-background-focused`, `-background-hover`. Hal tersebut penting saat mengubah tema global, karena memperbarui warna `toolbar`  
 8       akan secara otomatis memperbarui `hover states` untuk semua `buttons` di `toolbar` (Kode 2.58).

```
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 green
   */
8  ion-action-sheet {
9      --button-background-focus: green;
10     --button-background-focus-opacity: 0.5;
11 }
12
13 /*
14 * Setting the fab button background on hover to match the text color with
15 * the default --background-hover-opacity on md
16 */
17 .md ion-fab-button {
18     --color: #222;
19     --background-hover: #222;
20 }
```

Kode 2.58: Contoh Kode *Hover States* pada Ionic 5

33       – *CSS Utilities*

34       Karena pada versi sebelumnya, yaitu Ionic versi 4, terdapat masalah dengan menggunakan  
 35       atribut CSS dengan *framework* yang menggunakan JSX dan TypeScript, Ionic *Framework*  
 36       menambahkan dukungan untuk beberapa *framework*, dan pada Ionic 5 menambahkan kelas  
 37       CSS. Ionic versi 5 menghapus atribut CSS dan mendukung konsistensi. Ionic versi 5 juga  
 38       mengubah ke kelas dengan diawali `ion` untuk menghindari konflik dengan atribut asli dan  
 39       CSS dari pengguna (Kode 2.59).

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

Kode 2.59: Contoh Kode Kelas CSS *Utility* pada Ionic 5

1           – *Display Classes*

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

4           – *Distributed Scss*

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

7           • Komponen

8     Terdapat perubahan beberapa komponen pada Ionic 5, yaitu :

9           – Back Button dan Button

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

12           – Controllers

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

17           – Header dan Footer

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

20           – List Header

21    Konten berupa teks apa pun di dalam <ion-list-header> harus dibungkus dengan  
22    <ion-label> sesuai dengan gaya desain yang baru (Kode 2.60). Jika label tidak ada,  
23    maka perataan tombol di header bisa saja terlihat tidak aktif.

```
25   1 <ion-list-header>
26   2   <ion-label>New This Week</ion-label>
27   3   <ion-button>See All</ion-button>
28   4 </ion-list-header>
```

Kode 2.60: Kode Program untuk List Header

30           – Menu

31    Fungsi swipeEnable() telah dihapus di Angular, sebagai gantinya menggunakan swipeGesture(). Nilai *left* dan *right* telah dihapus, dan menggunakan *start* dan *end* sebagai gantinya.

33    Terdapat penghapusan atribut utama, sebagai gantinya yaitu dengan menggunakan content-id  
34    (untuk vanilla JS atau Vue) dan contentId (untuk Angular atau React) (Kode 2.61).

```
35   1 <ion-menu content-id="main"></ion-menu>
36   2 <ion-content id="main">...</ion-content>
```

Kode 2.61: Kode Program untuk Menu

1            - Select Option

2            Properti selected telah dihapus. Sebagai gantinya harus mengatur properti nilai pada ion-select induk agar sesuai dengan opsi terpilih yang diinginkan (Kode 2.62).

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

Kode 2.62: Kode Program untuk Select Option

10          - Toast

11          Properti close button seperti showCloseButton dan closeButtonText telah dihapus. Sebagai gantinya, gunakan buttons array untuk fungsi batal (Kode 2.63).

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

Kode 2.63: Kode Program untuk Toast

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

34          • *Package* dan *Dependencies*

35          Untuk memasang *package* dan *dependencies* pada Angular, dapat memanfaatkan npm pada CLI, dengan menjalankan pemasangan pada *package* ionic-angular (Kode 2.64). Namun jika ingin membuat proyek baru, dapat dibuat dari CLI dan aplikasi yang ada dapat dimigrasikan secara manual.

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

Kode 2.64: Kode untuk Memasang *Package* dan *Dependencies* pada Angular

---

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

1     • Warna

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

Tabel 2.1: Tabel Warna Bawaan di Ionic 5

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

3     • Events

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

6     3. Migrasi Ionic 5 ke Ionic 6

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

8       • Pembaruan Ionic dan Angular

9       Ionic 6 mendukung penggunaan Angular versi 12 dan yang lebih baru, dengan begitu perlu  
10      dilakukan perbaruan Angular ke versi yang terbaru (Kode 2.65).

11     1    npm install @ionic/angular@6

12       Kode 2.65: Kode untuk Memperbarui Versi Ionic 6 dengan versi Angular Terbaru

14       • Mengganti penggunaan Config.set() menjadi IonicModule.forRoot().

15       • *Icon*

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

18       • *Input, Select, dan Textarea*

19       Menggunakan “undefined” sebagai nilai untuk diteruskan ke porperti placeholder, dibandingkan  
20       dengan menggunakan “null”.

21       • *Modal dan Popover*

22       ion-modal (Kode 2.66) dan ion-popover (Kode 2.67) menggunakan Shadow DOM.

```
24     1 ion-modal::part(content) {
25       ...
26     }
27     4
28     5 ion-modal::part(backdrop) {
29       ...
30     }
```

31       Kode 2.66: Kode ion-modal menggunakan Shadow DOM pada CSS

```
1      1 ion-popover::part(arrow) {  
2          2     ...  
3      3 }  
4  
5      5 ion-popover::part(backdrop) {  
6          6     ...  
7      7 }  
8  
9      9 ion-popover::part(content) {  
10         10    ...  
11     11 }
```

Kode 2.67: Kode ion-popover menggunakan Shadow DOM pada CSS

14     • Radio

15       Menghapus semua penggunaan antarmuka RadioChangeEventDetail.



1

## BAB 3

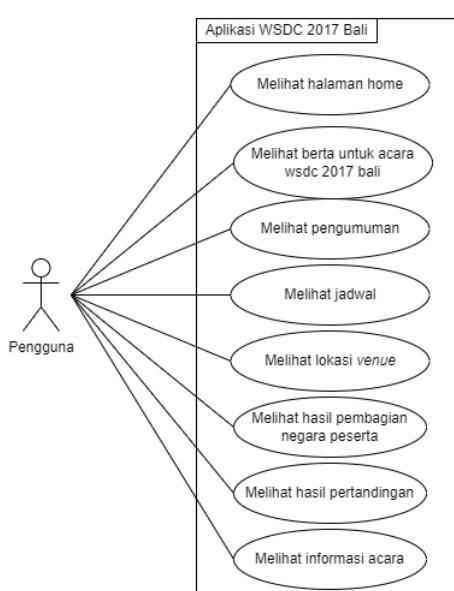
2

## ANALISIS

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

### 6 3.1 Analisis Sistem Kini dan Sistem Usulan

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



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

- 13 Terdapat *sidemenu* untuk pengguna agar dapat bernavigasi ke dalam menu-menu yang terdapat pada  
14 aplikasi WSDC 2017 Bali. Untuk mengakses *sidemenu*, pengguna dapat menekan tombol navigasi berada di  
15 sebelah kiri atas aplikasi WSDC 2017 Bali. Selain cara tersebut dapat juga dengan cara mengusap layar dari  
16 kiri ke kanan. Untuk menutup *sidemenu*, pengguna dapat menekan area di luar *sidemenu*, atau dengan cara

menekan tombol silang di sebelah kiri atas *sidemenu*. Terdapat fitur-fitur yang ada pada aplikasi WSDC 2017 Bali yang dapat diakses melalui *sidemenu*. Fitur-fitur tersebut adalah sebagai berikut :

#### 1. Home

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

Tabel 3.1: Tabel Skenario dari Halaman *Home*

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> .

#### 2. Newsletter

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

Tabel 3.2: Tabel Skenario dari *Newsletter*

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

#### 3. Announcement

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

Tabel 3.3: Tabel Skenario dari Halaman *Announcement*

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

#### 4. Schedule

Pada halaman ini, pengguna dapat melihat 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. Untuk mengakses halaman ini, dapat melalui *sidemenu* (Tabel 3.4).

Tabel 3.4: Tabel Skenario dari Halaman *Schedule*

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Schedule</i> pada <i>sidemenu</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.

1     5. *Venues*

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

Tabel 3.5: Tabel Skenario dari Halaman *Venues*

No	Aksi Aktor	Reaksi Sistem
1	Pengguna menekan tombol <i>Venues</i> pada <i>sidemenu</i>	Aplikasi WSDC 2017 Bali menampilkan halaman <i>Venues</i> yang berisi <i>Ceremony Venues</i> , <i>Competition Venues</i> , <i>Delegates Accomodation</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.

4     6. *Draw*

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

Tabel 3.6: Tabel Skenario dari Halaman *Draw*

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

8     7. *Result*

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

Tabel 3.7: Tabel Skenario dari Halaman *Result*

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

1    8. Info

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

Tabel 3.8: Tabel Skenario dari Halaman Info

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

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

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

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

```

24 @NgModule({
25   declarations: [
26     MyApp, HomePage, AnnouncementsPage, SchedulePage, VenuesPage, VenuesMapPage,
27     DrawPage, ResultPage, InfoPage
28   ],
29   imports: [
30     BrowserModule, HttpClientModule, IonicModule.forRoot(MyApp), IonicModule.forRoot(
31       CloudModule.forRoot(cloudSettings)
32     ),
33     bootstrap: [IonicApp],
34     entryComponents: [
35       MyApp, HomePage, AnnouncementsPage, SchedulePage, VenuesPage, VenuesMapPage,
36       DrawPage, ResultPage, InfoPage
37     ]
38   ]
39 }
```

```

11  ],
12  providers: [
13    StatusBar, SplashScreen, InAppBrowser, {provide: ErrorHandler, useClass:
14      IonicErrorHandler}, Geolocation,
15    ]
16  })
17
18 export class AppModule {}

```

Kode 3.1: NgModule pada app.module.ts

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

```

18
19 <ion-menu [content] = "content">
20   <ion-header>
21     <ion-toolbar>
22       <ion-title>
23         <button menuClose id = "menu-close-btn">
24           <ion-icon menu-close ios = "ios-close-circle-outline" md = "md-close-circle"></
25           ion-icon>
26         </button>
27         <span class = "text">Menu</span>
28       </ion-title>
29     </ion-toolbar>
30   </ion-header>
31
32   <ion-content>
33     <ion-list>
34       <button class = "title-sidemenu" menuClose ion-item *ngFor = "let p of pages" (click
35 ) = "openPage(p)">
36         <ion-icon [ios] = p.iosicon [md] = p.mdicon></ion-icon>
37         <span class = "text">{{p.title}}</span>
38       </button>
39     </ion-list>
40   </ion-content>
41
42 </ion-menu>
43
44 <!-- Disable swipe-to-go-back because it's poor UX to combine STGB with side menus -->
45 <ion-nav [root] = "rootPage" #content swipeBackEnabled = "false"></ion-nav>

```

Kode 3.2: Source Code File app.html

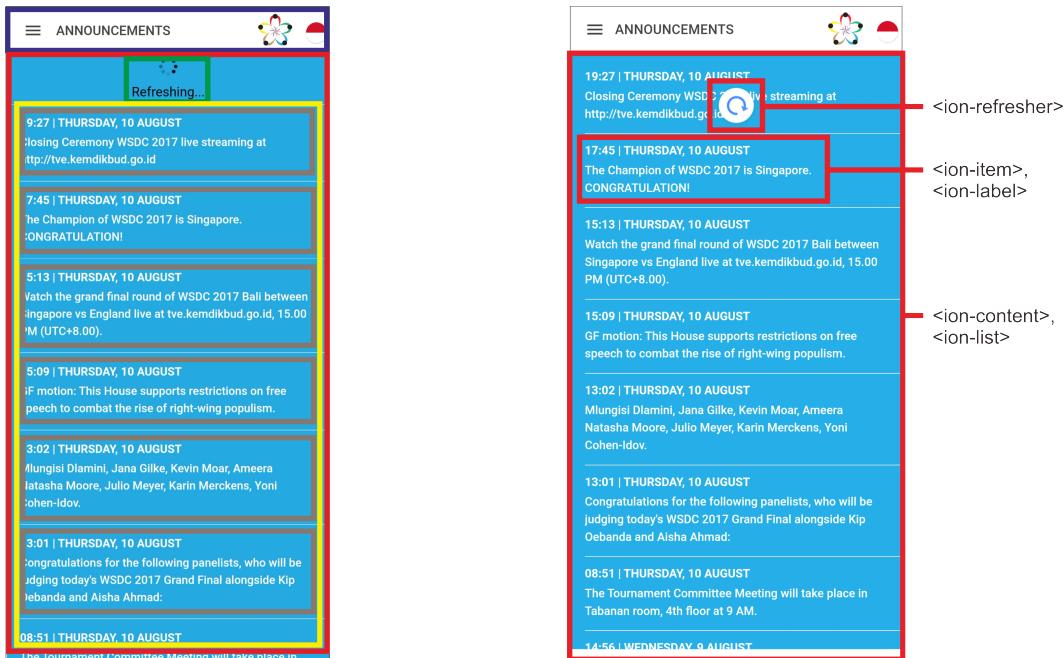
1 Selain komponen *root*, terdapat beberapa komponen lain yang berisi halaman-halaman yang ada di  
 2 aplikasi WSDC 2017 Bali. Masing-masing komponen akan mengimpor Component dari @angular/core, Na-  
 3 vController dari ionic-angular, dan Storage dari @ionic/storage. Mengimpor Component dari @angular/core  
 4 berfungsi untuk menambahkan sebuah komponen ke dalam *module*. Dengan begitu, komponen tersebut bisa  
 5 terlihat di seluruh aplikasi, dan dapat digunakan oleh komponen lain. NavController merupakan *base class*  
 6 untuk mengatur komponen navigasi. Ini berguna agar aplikasi dapat berpindah antar halaman. Sedangkan  
 7 Storage berfungsi untuk menyimpan pasangan *key/value* dan sebuah objek JSON.

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

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

### 13 1. Komponen *Announcement*

14 Komponen *Announcement* digunakan untuk melihat pengumuman terkait acara WSDC 2017 Bali.



(a) Wireframe Komponen Announcements Aplikasi WSDC 2017 Bali terdahulu

(b) UI Component Komponen Announcements Aplikasi WSDC 2017 Bali terbaru

Gambar 3.2: Komponen Announcements pada Aplikasi WSDC 2017 Bali

### 15 (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

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

```

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

```

Kode 3.3: @Component pada annoncement.ts

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

- `ionViewDidLoad()`

*Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman *announcement* telah dimuat. *Event* ini hanya berjalan satu kali, jadi jika sudah melakukan chache 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.

- `presentConnectionAlert()`

*Method* ini digunakan ketika method `doRefresh()` mengalami *error*, yang kemudian memunculkan *toast*.

- `formatDatetime(sqlDatetime: string)`

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

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

- *Header*

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

```

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

```

Kode 3.4: *Header* pada Halaman *Announcement*

- *Content*

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

```

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

```

Kode 3.5: *Content* pada Halaman *Announcement*

(b) Perancangan Aplikasi WSDC 2017 Bali terbaru

Pada komponen *announcement*, terdapat beberapa UI Component yang akan diimplementasikan seperti pada gambar 3.2b, 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 *announcement*. UI Component *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

1     • *Refresher*

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

5     • *List*

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

10    • *Item*

11    *Item* dengan tag `<ion-item>` sejak Ionic 4 mengalami perubahan dibandingkan pada  
12    Ionic 3, yaitu wajib menambahkan *label* dengan tag `<ion-label>`. Pada aplikasi WSDC  
13    2017 Bali saat ini yang menggunakan Ionic 3, tidak terdapat tag `<ion-label>` pada  
14    `<ion-item>` (Kode 3.6). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi  
15    yang akan dibangun yang menggunakan Ionic 6, akan menggunakan tag `<ion-label>` di  
16    dalam `<ion-item>` (Kode 3.7).

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

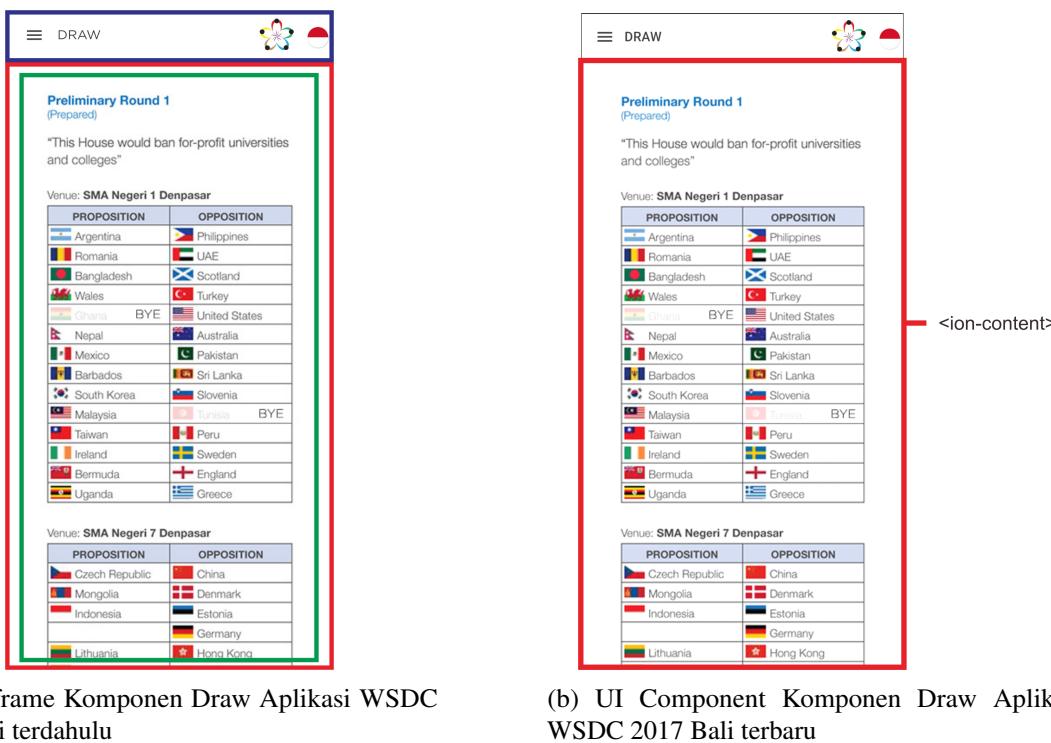
Kode 3.6: Tag `<ion-item>` dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

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

Kode 3.7: Tag `<ion-item>` dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan dibuat

32 2. Komponen *Draw*

33    Komponen *Draw* digunakan untuk melihat pembagian kubu proposisi dan oposisi dari tiap-tiap negara  
34    peserta.



(a) Wireframe Komponen Draw Aplikasi WSDC 2017 Bali terdahulu

(b) UI Component Komponen Draw Aplikasi WSDC 2017 Bali terbaru

Gambar 3.3: Komponen Draw pada Aplikasi WSDC 2017 Bali

1           (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

2           Terdapat *file* TypeScript, draw.ts, yang berfungsi untuk mengatur keseluruhan halaman. Di  
3           dalam *file* tersebut terdapat *decorator* @Component (Kode 3.8) dan *decorator* @ViewChild (Kode 3.9). Pada *decorator* @Component, terdapat CSS *selector* untuk memilih CSS mana yang  
4           akan digunakan, serta templateUrl untuk mendefinisikan ekxternal HTML template halaman  
5           Draw yang akan digunakan, yaitu draw.html. @ViewChild digunakan untuk memanggil elemen  
6           dari DOM untuk meamanggil komponen API ke dalam TypeScript, yaitu pada komponen draw  
7           adalah drawIFrame yang berada di *file* draw.html.  
8

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

Kode 3.8: @Component pada draw.ts

```
1  @ViewChild('drawIFrame') drawIFrame: ElementRef;
```

Kode 3.9: @ViewChild pada draw.ts

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

- 20           • ionViewDidLoad()

21           *Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman draw 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 berguna  
24           untuk melakukan pengaturan awal, yaitu untuk memuat data draw dari storage, kemudian

1 data tersebut dimasukkan ke dalam *child* drawIFrame. Terakhir, method ini akan memanggil  
 2 method presentLoading().

- 3 • presentLoading()

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

- 8 • onDrawIframeLoad()

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

12 Terdapat *file* draw.html yang digunakan untuk menampilkan tata letak dari halaman *draw*. Ter-  
 13 dapat beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan ke  
 14 dalam halaman *draw*. Diantaranya adalah sebagai berikut:

- 15 • *Header*

16 *Header* dari halaman *draw* seperti pada gambar 3.3a menggunakan *tag* <ion-header>  
 17 (*Kode 3.10*). *Tag* tersebut merupakan komponen *parent* yang menampung komponen  
 18 navbar yang ditandai dengan kotak berwarna biru pada gambar 3.3a. Di dalam navbar  
 19 tersebut, terdapat sebuah *tag* <button> untuk memunculkan sidemenu, <ion-icon>  
 20 untuk menampilkan icon dari tombol pada *tag button*, dan *tag* <ion-title> sebagai judul  
 21 dari halaman.

```
22
23 1 <ion-header>
24 2   <ion-navbar>
25 3     <button ion-button menuToggle>
26 4       <ion-icon name="menu"></ion-icon>
27 5     </button>
28 6     <ion-title>Draw</ion-title>
29 7   </ion-navbar>
30 8 </ion-header>
```

Kode 3.10: *Header* pada draw.html

- 32 • *Content*

33 *Content* dari halaman *draw* seperti pada gambar 3.3a menggunakan *tag* <ion-content>  
 34 (*Kode 3.11*) yang ditandai menggunakan kotak berwarna merah. Di dalam *tag* ini terdapat  
 35 sebuah *tag* <iframe> yang berisi hasil pengundian grup untuk peserta WSDC 2017 Bali,  
 36 ditandai menggunakan kotak berwarna hijau. *Tag* <iframe> menampilkan hasil dari  
 37 *method* onDrawIframeLoad() pada draw.ts.

```
38
39 1 <ion-content>
40 2   <iframe #drawIFrame (load)="onDrawIframeLoad()" class="iframe-
41 3     fullscreen"></iframe>
42 4 </ion-content>
```

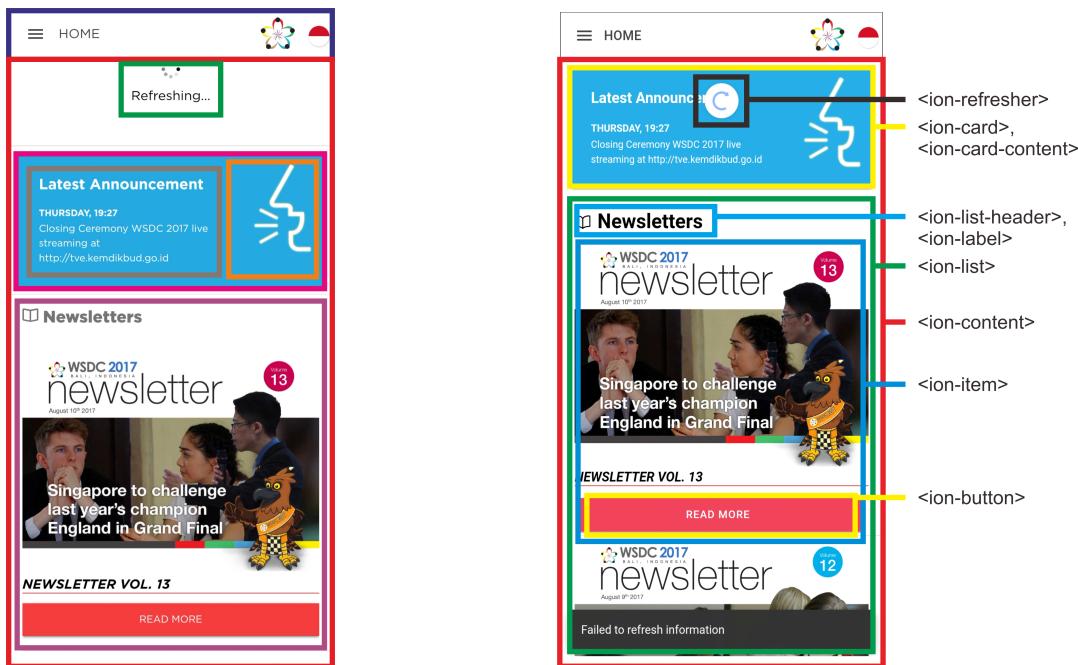
Kode 3.11: *Content* pada draw.html

1                   (b) Perancangan Aplikasi WSDC 2017 Bali terbaru

2                   Pada komponen *draw*, terdapat sebuah UI Component, yaitu *Content* seperti pada gambar 3.3b.  
 3                   Komponen *content* akan digunakan sebagai penyedia area konten yang digunakan untuk meng-  
 4                   ontrol area yang dapat digulir dan menampilkan isi konten dari halaman *draw*. UI Component  
 5                   *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubah-  
 6                   an dari Ionic Framework versi 3.

7                   3. Komponen *Home*

8                   Komponen *Home* digunakan untuk menampilkan halaman utama aplikasi WSDC 2017 Bali yang berisi  
 9                   pengumuman terbaru dari acara WSDC 2017 Bali, dan berita-berita terkait acara WSDC 2017 Bali.



(a) Wireframe Komponen Home Aplikasi WSDC 2017 Bali terdahulu

(b) UI Component Komponen Home Aplikasi WSDC 2017 Bali terbaru

Gambar 3.4: Komponen Home pada Aplikasi WSDC 2017 Bali

10                  (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

11                  Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam  
 12                  *file* `home.ts` terdapat sebuah *decorator* `@Component` untuk komponen (Kode 3.12). Di dalam  
 13                  decorator ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta *templateUrl*  
 14                  untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template HTML* yang  
 15                  digunakan adalah *file* `home.html`.

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

Kode 3.12: `@Component` pada `home.ts`

1 Komponen *Home* merupakan komponen yang menjadi rootPage dari aplikasi ini, yang dimasukan  
2 di dalam *file* app.component.ts. Maka dari itu, saat pertama kali aplikasi dijalankan, komponen  
3 *home*-lah yang pertama kali ditampilkan di dalam layar. rootPage di dalam *file* app.component.ts  
4 akan memanggil komponen *home*, yang kemudian *file* home.ts akan berjalan.

5 Di dalam *file* ini terdapat sebuah kelas HomePage yang berisi beberapa *method*, diantaranya  
6 adalah sebagai berikut:

- 7 • ionViewDidLoad()

8 *Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman *home* telah  
9 dimuat. *Event* ini hanya berjalan satu kali, jadi jika sudah melakukan chache terhadap  
10 halaman ini, maka ionViewDidLoad() tidak akan berjalan lagi. *Lifecycle event* ini berguna  
11 untuk melakukan pengaturan awal, yaitu untuk mengambil keseluruhan data aplikasi  
12 yang ada di dalam penyimpanan. Dengan memanfaatkan fitur *storage* yang dimiliki oleh  
13 Ionic Framework, *method* ini akan mengecek apakah sudah ada data dengan format .json  
14 yang berisi keseluruhan data aplikasi di dalam penyimpanan. Data tersebut berisi data  
15 *announcements*, *newsletters*, *schedule*, *venues*, *draws*, dan *info*. Jika data tersebut tidak  
16 ditemukan, maka akan diambil dari *file* wsdc-data.json dari aset lokal menggunakan HTTP  
17 API yang disediakan oleh Angular, yaitu HttpClient, kemudian dimasukan ke dalam  
18 penyimpanan. Hal ini bertujuan jika pengguna tidak memiliki koneksi internet pada saat  
19 pemasangan aplikasi, aplikasi masih bisa dijalankan dan menampilkan halaman-halaman  
20 yang tidak kosong karena data diambil dari aset lokal.

21 Setelah itu, *method* ini mengambil data terbaru dari server dengan menggunakan Angular  
22 HttpClient. Jika sudah melewati batas waktu, dan aplikasi belum terhubung dengan server,  
23 maka *method* ini akan memanggil *method* showToast() yang akan menampilkan Toast yang  
24 berisi teks ‘Failed to refresh information’. Jika mengambil data dari server berhasil, maka  
25 data yang didapatkan dari server akan dimasukan ke dalam penyimpanan menggantikan data  
26 yang sudah ada di penyimpanan sebelumnya. Hal ini dilakukan dengan asumsi bahwa data  
27 yang terdapat di server merupakan data terbaru, sehingga data yang ada pada penyimpanan  
28 merupakan data lama dan harus diganti dengan data terbaru.

- 29 • launch(url: string)

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

- 35 • formatDatetime(sqlDatetime: string)

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

1       • doRefresh(refresher)

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

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

5       *Method* ini berfungsi untuk memunculkan sebuah *native Toast*, yaitu sebuah *popup* teks,  
 6       dengan memanfaatkan *UI Component* milik Ionic Framework. *Method* ini menerima pa-  
 7       rameter berupa sebuah *string*, yang berisi pesan yang akan dimunculkan ke dalam sebuah  
 8       *Toast*, dan memiliki sebuah parameter *duration* yang berisi lama waktu

9       • onAnnouncementClick()

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

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

16       • *Header*

17      Halaman *home* memiliki *header* dengan *tag <ion-header>* (Kode 3.13) seperti pada  
 18     gambar 3.4a yang ditandai dengan kotak berwarna biru. *Tag* tersebut merupakan komponen  
 19     *parent* yang menampung komponen *navbar* yang ditandai dengan kotak berwarna biru.  
 20     Di dalam *navbar* tersebut, terdapat sebuah *tag <button>* untuk memunculkan *sideme-  
 21     nu*, *<ion-icon>* untuk menampilkan icon, dan *tag <ion-title>* sebagai judul dari  
 22     halaman.

```
31 1 <ion-header>
32 2   <ion-navbar>
33 3     <button ion-button menuToggle>
34 4       <ion-icon name="menu"></ion-icon>
35 5     </button>
36 6     <ion-title>Home</ion-title>
37 7   </ion-navbar>
38 8 </ion-header>
```

Kode 3.13: *Header* pada *home.html*

40       • *Content*

41      *Content* pada halaman *home* dengan *tag <ion-content>* (Kode 3.14 pada gambar 3.4a  
 42     ditandai dengan kotak berwarna merah. Di dalam *tag <ion-content>* terdapat beberapa  
 43     *tag* lainnya. Pertama yaitu sebuah *tag <ion-refresher>* yang digunakan untuk menam-  
 44     pilkian simbol *refresh* saat pengguna menyegarkan halaman dengan cara melakukan *swipe*

1 dari atas ke bawah layar, ditandai dengan kotak berwarna hijau. Terdapat tag `<ion-card>`  
2 yang digunakan sebagai tempat untuk pengumuman terkait acara WSDC 2017 Bali disimpan.  
3 Penggunaan *card* ditantai dengan kotak berwarna merah muda. Di dalam tag `<ion-card>`  
4 terdapat tag `<ion-grid>` untuk mengatur tata letak dari penyusunan isi dari suatu *card*. Di  
5 dalam *grid* tersebut terdapat satu baris dengan tag `<ion-row>` dan dua kolom dengan tag  
6 `<ion-col>`. Kolom pertama ditandai dengan kotak berwarna coklat, berisi tanggal beserta  
7 pengumuman, dan kolom kedua ditandai dengan kotak berwarna oranye berisi gambar.  
8 Selanjutnya terdapat sebuah tag `<ion-list>` untuk menyimpan berita-berita terkait acara  
9 WSDC 2017 Bali, yang ditandai dengan warna ungu. Di dalam *list* tersebut terdapat tag  
10 `<ion-list-header>` sebagai judul dari *list*, dan tag `<ion-item>` untuk menyimpan  
11 berita-berita terkait acara WSDC 2017 Bali. Di dalam tag `<ion-item>` terdapat tag  
12 `<button>` yang apabila ditekan oleh pengguna, maka akan mengarahkan pengguna untuk  
13 melihat berita tertentu sesuai dengan *item* yang dipilih dengan memanggil *method* `launch()`  
14 yang ada di `home.ts`.

```
1 <ion-content>
2   <ion-refresher (ionRefresh)="doRefresh($event)">
3     <ion-refresher-content pullingIcon="arrow-dropdown" pullingText="Pull
4       to refresh" refreshingSpinner="circles" refreshingText="Refreshing
5       ...">
6     </ion-refresher-content>
7   </ion-refresher>
8   <ion-card (click)="onAnnouncementClick()">
9     <ion-grid>
10    <ion-row>
11      <ion-col col-9>
12        <ion-card-header text-wrap>
13          Latest Announcement
14        </ion-card-header>
15        <ion-card-content>
16          <h3>{{formatDatetime(wsdcData?.announcements[0].localtime)
17            }}</h3>
18          <p>{{wsdcData?.announcements[0].message}}</p>
19        </ion-card-content>
20      </ion-col>
21      <ion-col col-3>
22        
23      </ion-col>
24    </ion-row>
25  </ion-grid>
26 </ion-card>
27 <ion-list>
28   <ion-list-header>
29     <ion-icon ios="ios-book-outline" md="md-book"></ion-icon>
30     Newsletters
31   </ion-list-header>
32   <ion-item *ngFor="let wsdcNews of wsdcData?.newsletters">
33     
35     <h2 text-wrap>{{wsdcNews.title}}</h2>
36   </ion-item>
37 </ion-list>
```

```

1   32      <button ion-button full block color="danger" (click)="launch(
2     33        wsdcNews.url)">Read More</button>
3   34    </ion-item>
4   35  </ion-list>
5  </ion-content>

```

Kode 3.14: *Content* pada home.html

7 (b) Perancangan Aplikasi WSDC 2017 Bali terbaru

8 Pada komponen *Home*, terdapat beberapa UI Component yang akan diimplementasikan seperti  
9 pada gambar 3.4b, diantaranya adalah sebagai berikut:

10 • Content

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

15 • *Refresher*

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

19 • *Card*

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

24 • *Grid*

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

29 • *List Header*

30 *List Header* dengan tag `<ion-list-header>` sejak Ionic 4 diwajibkan untuk selalu  
31 menambahkan tag `<ion-label>`. Pada aplikasi WSDC 2017 Bali saat ini yang meng-  
32 gunakan Ionic 3, tidak terdapat tag `<ion-label>` pada `<ion-list-header>` (Ko-  
33 de 3.15). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi yang akan  
34 dibangun yang menggunakan Ionic 6, akan menggunakan tag `<ion-label>` di dalam  
35 `<ion-item-header>` (Kode 3.16).

```

36 1  <ion-list-header>
37 2    <ion-icon ios="ios-book-outline" md="md-book"></ion-icon>
38 3    Newsletters
39 4  </ion-list-header>
40

```

Kode 3.15: Tag `<ion-list-header>` dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

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

Kode 3.16: Tag `<ion-list-header>` dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan dibuat

- *Icon*

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

- *Button*

Di dalam halaman *home*, komponen ini merupakan sebuah komponen yang dapat diklik untuk mengarahkan pengguna ke URL yang berisi berita terkait WSDC 2017 Bali. Pada aplikasi WSDC 2017 Bali saat ini yang menggunakan Ionic 3, komponen ini ditulis menggunakan tag `<button>` (Kode 3.17). Sejak Ionic Framework versi 4, terjadi perubahan dengan mengganti tag tersebut menjadi `<ion-button>` pada aplikasi yang akan dibangun yang menggunakan Ionic 6 (Kode 3.18).

```
18 1 <button ion-button full block color="danger" (click)="launch(wsdcNews.url
19 2   )">Read More</button>
20
21
```

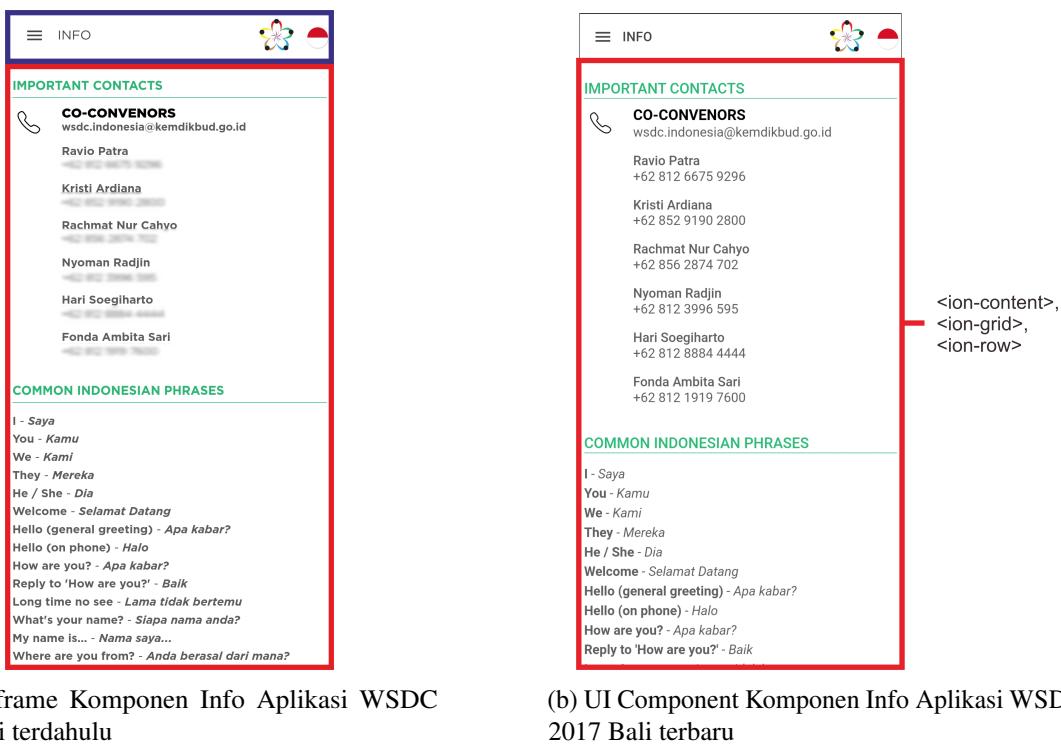
Kode 3.17: *Button* dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```
22 1 <ion-button full block color="danger" (click)="launch(wsdcNews.url)">Read
23 2   More</ion-button>
24
```

Kode 3.18: *Button* dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan dibuat

#### 4. Komponen Info

Komponen Info digunakan untuk menampilkan informasi yang berisi kontak penting untuk acara WSDC 2017 Bali, kosa-kata dalam Bahasa Indonesia, serta *credits* kepada pengembang aplikasi.



(a) Wireframe Komponen Info Aplikasi WSDC 2017 Bali terdahulu

(b) UI Component Komponen Info Aplikasi WSDC 2017 Bali terbaru

Gambar 3.5: Komponen Info pada Aplikasi WSDC 2017 Bali

### 1 (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

2 Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam  
 3 *file* info.ts terdapat sebuah *decorator* @Component untuk komponen (Kode 3.19). Di dalam  
 4 *decorator* ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta **templateUrl**  
 5 untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template HTML* yang  
 6 digunakan adalah *file* info.html.

```
7   1 @Component ({  

8     2   selector: 'page-info',  

9     3   templateUrl: 'info.html'  

10    4 })
```

Kode 3.19: @Component pada info.ts

13 Terdapat kelas InfoPage pada komponen info. Kelas ini hanya berisi *constructor* yang digunakan  
 14 untuk menginisialisai halaman yang akan digunakan. *Constructor* sendiri berfungsi untuk memuat  
 15 data info dari penyimpanan dan memasukannya ke dalam sebuah variabel lokal. Terdapat *file*  
 16 info.html yang digunakan untuk menampilkan tata letak dari halaman info. Terdapat beberapa  
 17 komponen yang disediakan oleh Ionic Framework, yang diimplementasikan ke dalam halaman  
 18 info. Diantaranya adalah sebagai berikut:

19 • *Header*

20 Halaman info memiliki *header* dengan tag <ion-header> (Kode 3.20) seperti pada  
 21 gambar 3.5a. Tag tersebut merupakan komponen *parent* yang menampung komponen navbar  
 22 yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat sebuah tag  
 23 <button> untuk memunculkan sidemenu dan tag <ion-icon> untuk menampilkan icon  
 24 dari tombol pada tag button. Terdapat tag <ion-title> sebagai judul dari halaman, yaitu

1           “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 3.20: *Header* pada info.html

- *Content*

*Content* pada halaman info memiliki tag `<ion-content>` (Kode 3.21) yang pada gambar 3.5a 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 1 <ion-content>
2 2   <ion-grid>
3 3     <ion-row>
4 4       <div [innerHTML]=wsdcInfoData>
5 5     </div>
6 6   </ion-row>
7 7 </ion-grid>
8 8 </ion-content>
```

Kode 3.21: *Content* pada info.html

28 (b) Perancangan Aplikasi WSDC 2017 Bali terbaru

29 Pada komponen info, terdapat beberapa UI Component yang akan diimplementasikan seperti  
30 pada gambar 3.5b, diantaranya adalah sebagai berikut:

- *Content*

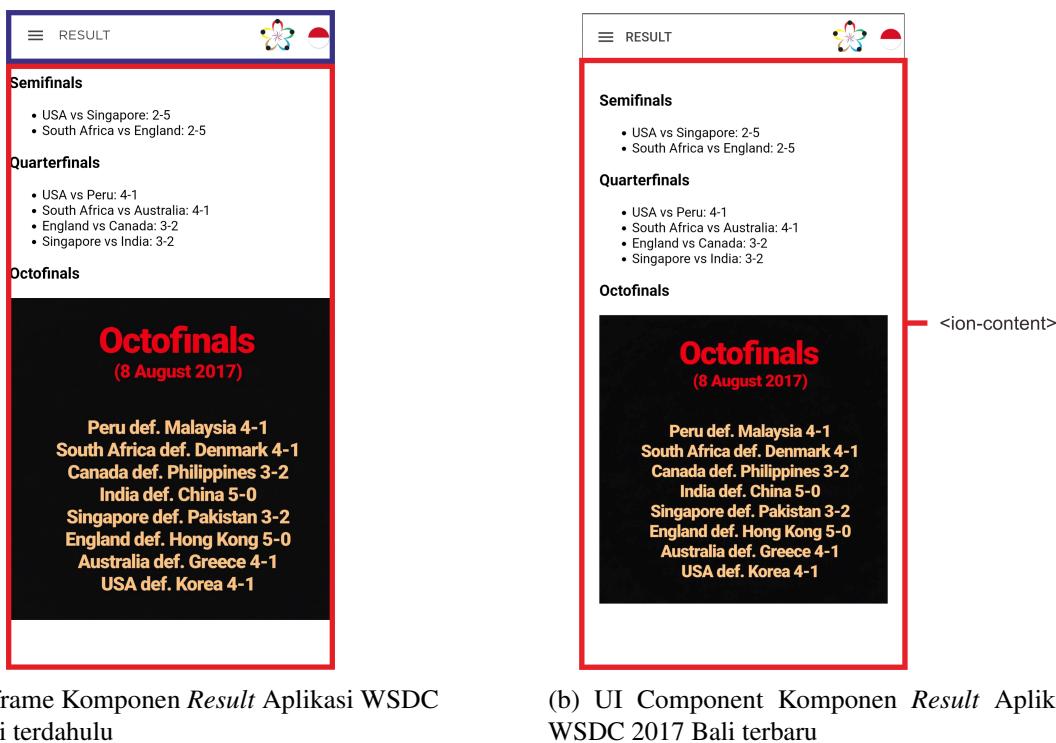
32 Komponen ini akan digunakan sebagai penyedia area konten yang digunakan untuk mengontrol area yang dapat digulir dan menampilkan isi konten dari halaman info. UI Component  
33 *Content* dengan tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Framework versi 3.

- *Grid*

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

40 5. Komponen *Result*

41 Komponen *Result* digunakan untuk melihat hasil dari pertandingan yang berisi pemenang dan skornya  
42 dari masing-masing babak pertandingan.

Gambar 3.6: Komponen *Result* pada Aplikasi WSDC 2017 Bali

#### 1 (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

2 Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam  
 3 *file* result.ts terdapat sebuah *decorator* @Component untuk komponen (Kode 3.22) dan *decorator*  
 4 @ViewChild (Kode 3.27). Di dalam *decorator* ini terdapat CSS *selector* untuk memilih CSS yang  
 5 akan digunakan, serta templateUrl untuk mendefinisikan ekxternal HTML *template* yang akan  
 6 digunakan. *Template* HTML yang digunakan adalah *file* info.html. @ViewChild digunakan untuk  
 7 memanggil elemen dari DOM untuk memanggil komponen API ke dalam TypeScript, yaitu pada  
 8 komponen *result* adalah resultIFrame yang berada di *file* result.html.

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

Kode 3.22: @Component pada result.ts

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

Kode 3.23: @ViewChild pada result.ts

18 Terdapat kelas ResultPage dengan beberapa *method* yang digunakan, yaitu:  
 19

- ionViewDidLoad()

20 *Method* ini merupakan sebuah *lifecycle event* yang dijalankan saat halaman *result* telah  
 21 dimuat. *Event* ini hanya berjalan satu kali, jadi jika sudah melakukan cache terhadap  
 22 halaman ini, maka ionViewDidLoad() tidak akan berjalan lagi. *Lifecycle event* ini berguna  
 23 untuk melakukan pengaturan awal, yaitu untuk memuat data *result* yang sudah disimpan

1 di dalam penyimpanan internal. Setelah berhasil memuat data *result*, data tersebut akan  
 2 dimasukan ke dalam *child* resultIFrame. Terakhir, akan dipanggil *method* presentLoading().

- 3 • presentLoading()

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

- 8 • onResultIframeLoad()

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

12 Terdapat *file* result.html yang digunakan untuk menampilkan tata letak dari halaman *result*.  
 13 Terdapat beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan  
 14 ke dalam halaman *result*. Diantaranya adalah sebagai berikut:

- 15 • *Header*

16 Halaman *result* memiliki *header* dengan tag <ion-header> (Kode 3.24) seperti pada  
 17 gambar 3.6a. Tag tersebut merupakan komponen *parent* yang menampung komponen navbar  
 18 yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat sebuah tag  
 19 <button> untuk memunculkan sidemenu, dan <ion-icon> untuk menampilkan icon  
 20 dari tombol pada tag *button*. Terdapat tag <ion-title> sebagai judul dari halaman, yaitu  
 21 “Result”.

```
22
23 1 <ion-header>
24 2   <ion-navbar>
25 3     <button ion-button menuToggle>
26 4       <ion-icon name="menu"></ion-icon>
27 5     </button>
28 6     <ion-title>Result</ion-title>
29 7   </ion-navbar>
30 8 </ion-header>
```

Kode 3.24: *Header* pada result.html

- 32 • *Content*

33 *Content* pada halaman *result* memiliki tag <ion-content> (Kode 3.25) yang pada gam-  
 34 bar 3.6a dengan kotak berwarna merah. Di dalam tag <ion-content> terdapat tag  
 35 <iframe>. Tag tersebut berisi informasi mengenai daftar pemenang acara WSDC 2017  
 36 bali yang di dapatkan pada *method* onResultIframeLoad() di kelas ResultPage pada *file*  
 37 result.ts.

```
38
39 1 <ion-content>
40 2   <iframe #resultIFrame (load)="onResultIframeLoad()" class="iframe-
41 3     fullscreen"></iframe>
42 4 </ion-content>
```

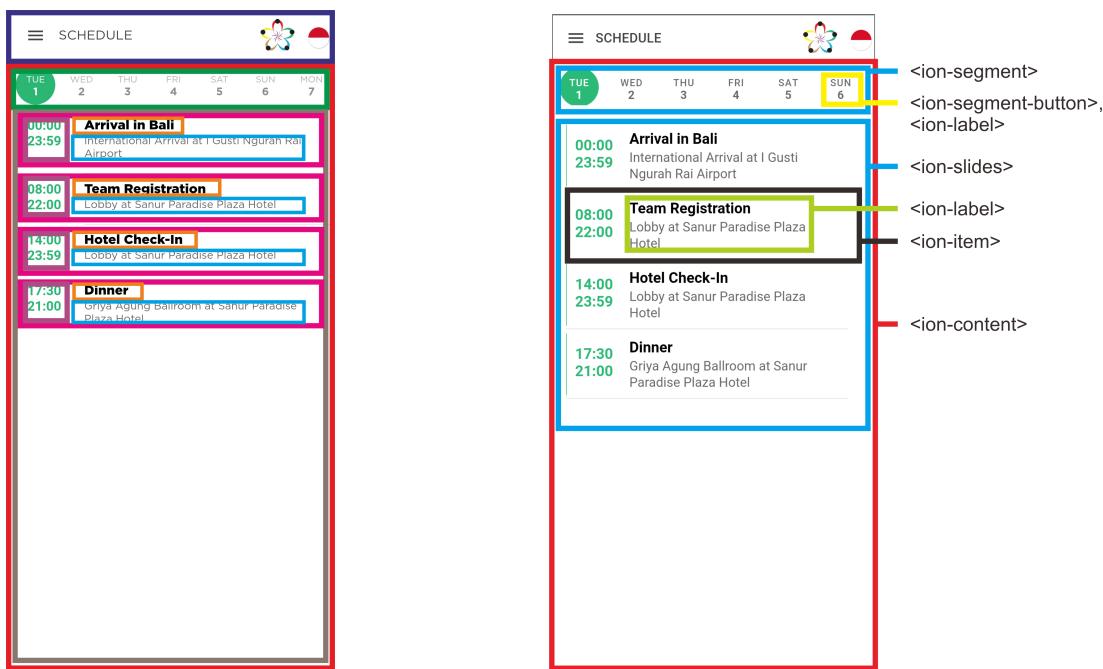
Kode 3.25: *Content* pada result.html

1                   (b) Perancangan Aplikasi WSDC 2017 Bali terbaru

2                   Pada komponen *Result*, terdapat sebuah UI Component, yaitu *Content* seperti pada gambar 3.6b.  
 3                   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  
 4                   tag `<ion-content>` pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic Fra-  
 5                   mework versi 3.

6. Komponen *Schedule*

8                   Komponen *Schedule* digunakan untuk melihat jadwal pertandingan yang berisi waktu, tanggal, hari,  
 9                   dan lokasi untuk setiap jadwal pertandingan WSDC 2017 Bali.



(a) Wireframe Komponen *Schedule* Aplikasi WSDC 2017 Bali terdahulu

(b) UI Component Komponen *Schedule* Aplikasi WSDC 2017 Bali terbaru

Gambar 3.7: Komponen *Schedule* pada Aplikasi WSDC 2017 Bali

10                 (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

11                 Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam  
 12                 *file* `schedule.ts` terdapat sebuah *decorator* `@Component` untuk komponen (Kode 3.26) dan dua  
 13                 *decorator* `@ViewChild` (Kode 3.27). Di dalam *decorator* ini terdapat CSS *selector* untuk memilih  
 14                 CSS yang akan digunakan, serta `templateUrl` untuk mendefinisikan eksternal HTML *template*  
 15                 yang akan digunakan. *Template* HTML yang digunakan adalah *file* `info.html`. `@ViewChild`  
 16                 digunakan untuk memanggil elemen dari DOM untuk memanggil komponen API ke dalam  
 17                 TypeScript, yaitu pada komponen *result* adalah `scheduleSlider` dan `segmentContainer` yang  
 18                 berada di *file* `result.html`. `scheduleSlider` berfungsi untuk menyimpan konten dari sebuah *slide*.  
 19                 Sedangkan `segmentContainer` berfungsi untuk menyimpan konten dari sebuah *segment*.

```

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

```

Kode 3.26: @Component pada schedule.ts

```

7 1 @ViewChild('scheduleSlider') slider: Slides;
8 2 @ViewChild('segmentContainer') segmentContainer: ElementRef;

```

Kode 3.27: @ViewChild pada schedule.ts

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. 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.

- onSlideChanged()

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

- getDayName(sqlDate: string)

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

- getDate(sqlDate: string)

*Method* ini bergunci untuk mengembalikan tanggal dari parameter.

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

- *Header*

Halaman *schedule* memiliki *header* dengan tag `<ion-header>` (Kode 3.28) seperti pada gambar 3.7a. Tag tersebut merupakan komponen *parent* yang menampung komponen navbar yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat sebuah tag `<button>` untuk memunculkan sidemenu dan `<ion-icon>` untuk menampilkan icon dari tombol pada tag *button*. Terdapat tag `<ion-title>` sebagai judul dari halaman, yaitu

```

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

```

Kode 3.28: *Header* pada schedule.html

- *Content*

*Content* pada halaman result memiliki tag `<ion-content>` (Kode 3.29) yang pada gambar 3.7a dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat dua buah tag `<div>` yang masing masing berisi tag `<ion-segment>` dan tag `<ion-slides>`. Tag `<ion-segment>` digunakan untuk tampilan hari, seperti pada gambar 3.7a yang ditandai dengan kotak berwarna hijau. Tag `<ion-slides>` digunakan untuk tampilan jadwal di dalam satu hari, seperti yang ditandai dengan kotak berwarna coklat.

Setiap jadwal yang berada di tag `<ion-slides>` dibungkus dengan tag `<ion-list>` seperti pada kotak berwarna muda di gambar 3.7a. Dalam satu tag `<ion-list>` terdapat tag `<ion-note>` yang berisi waktu mulai dan waktu selesai dari satu jadwal seperti yang ditandai dengan kotak berwarna ungu, tag `<h3>` berisi nama acara seperti yang ditandai dengan kotak berwarna oranye, dan tag `<p>` yang berisi tempat acara tersebut diadakan ditandai dengan kotak berwarna biru muda.

```

25
26   1 <ion-content>
27     2   <div id="schedulesContainer">
28       3     <div id="schedulesSegments">
29         4       <ion-segment #segmentContainer *ngIf="schedules" [(ngModel)]="selectedSegmentIdx" (ionChange)="onSegmentChanged($event)">
30           5         <ion-segment-button *ngFor="let schedule of schedules; let i = index" [value]="i">
31             6           <div class="day">{{getDayName(schedule.date)}}</div>
32             7           <div class="date">{{getDate(schedule.date)}}</div>
33             8         </ion-segment-button>
34           9       </ion-segment>
35         10     </div>
36       11     <div id="schedulesSlides">
37         12       <ion-slides #scheduleSlider (ionSlideDidChange)="onSlideChanged()">
38           13         <ion-slide *ngFor="let schedule of schedules">
39             14               <ion-list>
40               15                 <ion-item text-wrap *ngFor="let agenda of schedule.agenda">
41                 16                   <ion-note item-start>
42                   17                     {{agenda.start}}<br/>
43                     18                     {{agenda.end}}
44                   19                 </ion-note>
45                 20                   <h3>{{agenda.title}}</h3>
46                 21                   <p>{{agenda.subtitle}}</p>
47               22             </ion-item>

```

```

1      23      </ion-list>
2      24      </ion-slide>
3      25      </ion-slides>
4      26      </div>
5      27      </div>
6      28  </ion-content>

```

Kode 3.29: *Content* pada schedule.html

- Perancangan Aplikasi WSDC 2017 Bali terbaru

Pada komponen *schedule*, terdapat beberapa UI Component yang akan diimplementasikan seperti pada gambar 3.7b, 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 3.30). 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 3.31).

```

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

```

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

```

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

```

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

1            - *List*

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

6            - *Segment*

7        Komponen ini akan digunakan untuk pengguna agar dapat berpindah tampilan di dalam  
 8        halaman yang sama. Seperti pada tampilan halaman jadwal yang ada pada aplikasi  
 9        WSDC 2017 Bali saat ini, dimana pengguna dapat berpindah hari untuk mengetahui  
 10      jadwal kegiatan pada hari tertentu yang dipilih oleh pengguna, namun masih berada  
 11      di halaman yang sama, yaitu halaman Schedule. UI Component *Segment* dengan tag  
 12      <ion-segment> pada Ionic Framework versi 6 tidak mengalami perubahan dari Ionic  
 13      Framework versi 3. Sedangkan tag <ion-segment-button> yang berada di dalam  
 14      tag <ion-segment> sejak Ionic 4 mengalami perubahan yaitu wajib menambahkan  
 15      *label* dengan tag <ion-label>. Pada aplikasi WSDC 2017 Bali saat ini yang  
 16      menggunakan Ionic 3, tidak terdapat tag <ion-label> pada <ion-item> (Kode 3.32). Sedangkan sesuai dengan ketentuan yang berlaku, pada aplikasi yang akan  
 17      dibangun yang menggunakan Ionic 6, akan menggunakan tag <ion-label> di dalam  
 18      <ion-item> (Kode 3.33).

```
20
21 1 <ion-segment-button *ngFor="let schedule of schedules; let i = index"
22   [value]="i">
23   <div class="day">{{getDayName(schedule.date)}}</div>
24   <div class="date">{{getDate(schedule.date)}}</div>
25 </ion-segment-button>
```

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

```
27
28 1 <ion-segment-button *ngFor="let schedule of schedules; let i = index"
29   [value]="i">
30   <ion-label>
31     <div class="day">{{getDayName(schedule.date)}}</div>
32     <div class="date">{{getDate(schedule.date)}}</div>
33   </ion-label>
34 </ion-segment-button>
```

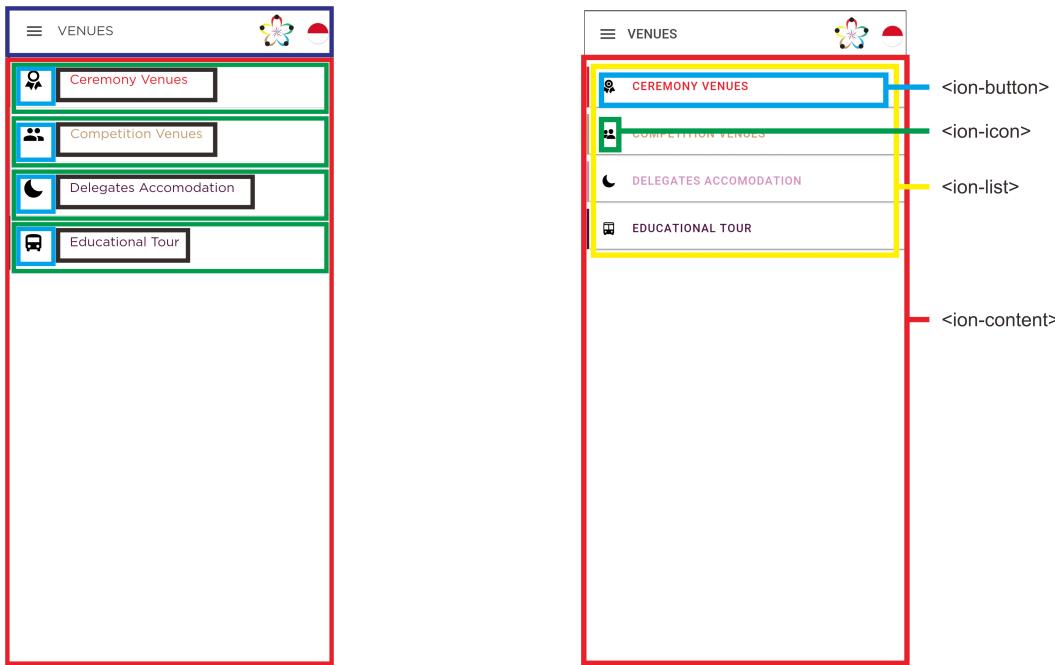
Kode 3.33: Tag <ion-segment-button> dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan Dibuat

36            - *Slides*

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

1      7. Komponen *Venues*

2      Komponen *Venues* digunakan untuk menampilkan halaman *venues* yang berisi kategori dari masing-  
3      masing *venues*. Tiap-tiap kategori dapat ditekan untuk menampilkan halaman *venues maps*.



(a) Wireframe Komponen *Venues* Aplikasi WSDC 2017 Bali terdahulu

(b) UI Component Komponen *Venues* Aplikasi WSDC 2017 Bali terbaru

Gambar 3.8: Komponen *Venues* pada Aplikasi WSDC 2017 Bali

4      (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

5      Komponen ini memiliki sebuah *file* TypeScript untuk mengatur keseluruhan halaman. Di dalam  
6      *file* *venues.ts* terdapat sebuah *decorator* `@Component` untuk komponen (Kode 3.34). Di dalam  
7      *decorator* ini terdapat CSS *selector* untuk memilih CSS yang akan digunakan, serta **templateUrl**  
8      untuk mendefinisikan eksternal HTML *template* yang akan digunakan. *Template* HTML yang  
9      digunakan adalah *file* *venues.html*

```
10     1 @Component ({  
11       2   selector: 'page-venues',  
12       3   templateUrl: 'venues.html'  
13       4 })
```

Kode 3.34: `@Component` pada *venues.ts*

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

21      Terdapat *file* *venues.html* yang digunakan untuk menampilkan halaman *venues*. Terdapat beberapa  
22      komponen yang disediakan oleh Ionic Framework, yang diimplementasikan ke dalam halaman  
23      *venues*. Diantaranya adalah sebagai berikut:

1       • *Header*

2       Halaman *venues* memiliki *header* dengan tag `<ion-header>` (Kode 3.35) seperti pada  
 3       gambar 3.8a. Tag tersebut merupakan komponen *parent* yang menampung komponen navbar  
 4       yang ditandai dengan kotak berwarna biru. Di dalam navbar tersebut, terdapat sebuah tag  
 5       `<button>` untuk memunculkan *sidemenu*. Terdapat tag `<ion-title>` sebagai judul dari  
 6       halaman, yaitu “Venues”.

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

Kode 3.35: *Header* pada *venues.html*

17     • *Content*

18     *Content* pada halaman *venues* memiliki tag `<ion-content>` (Kode 3.36) yang pada gam-  
 19     bar 3.8a dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat sebuah tag  
 20     `<ion-grid>` dan sebuah tag `<ion-row>`. Di dalam tag `<ion-row>` terdapat sebuah  
 21     tag `<ion-list>` yang berisi tag `<ion-button>` yang ditandai dengan kotak berwarna  
 22     hijau pada gambar 3.8a. Masing-masing tag `<ion-button>` berisi tag `<ion-icon>`  
 23     yang ditandai dengan kotak berwarna biru muda, dan tag `<span>` berisi nama *venues* yang  
 24     ditandai dengan kotak berwarna hitam.

```
25
26   1 <ion-content>
27   2   <ion-grid>
28   3     <ion-row>
29   4       <ion-list style="width: 100%;" no-lines>
30   5         <button ion-item id="{{wsdcVenue.id}}" *ngFor="let wsdcVenue
31   6           of venuesData" (click)="itemTapped($event, wsdcVenue)">
32   7           <ion-icon ios="ios-{{wsdcVenue.icon}}-outline" md="md-{{
33   8             wsdcVenue.icon}}" item-start></ion-icon>
34   9           <span>{{wsdcVenue.name}}</span>
35   10          </button>
36   11        </ion-list>
37   12      </ion-row>
38   13    </ion-grid>
39   14  </ion-content>
```

Kode 3.36: *Content* pada *venues.html*

41     Terdapat kelas *VenuesPage* pada *venues.ts*. Kelas ini memiliki satu *constructor*. *Constructor*  
 42     kelas ini berfungsi untuk mengambil data *venues* yang berada di dalam penyimpanan. Data  
 43     tersebut kemudian disimpan ke dalam variabel lokal *venuesData*, yang berisi *id*, *name*, *icon*,  
 44     *geojson*, dan *colorIdx*. Terdapat sebuah *method* yaitu *itemTapped(event, wsdcVenue)*. *Method*  
 45     ini memiliki dua buah parameter, *event* yang berisi *event* pada tag *button*, dan *wsdcVenue* yang  
 46     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.

(b) Perancangan Aplikasi WSDC 2017 Bali terbaru

Pada komponen *venues*, terdapat beberapa UI Component yang akan diimplementasikan seperti pada gambar 3.8b, diantaranya adalah sebagai berikut:

- Content

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

- Grid

Komponen ini akan digunakan untuk membangun tata letak kustom pada halaman info. UI Component *Grid* dengan *tag* <ion-grid> 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 3.39). Sejak Ionic Framework versi 4, terjadi perubahan dengan mengganti *tag* tersebut menjadi <ion-button> pada aplikasi yang akan dibangun yang menggunakan Ionic 6 (Kode 3.40).

```

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

Kode 3.37: *Button* dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

```

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

Kode 3.38: *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.

(c) Perancangan Aplikasi WSDC 2017 Bali terbaru

Pada komponen *venues*, terdapat beberapa UI Component yang akan diimplementasikan seperti pada gambar 3.8b, diantaranya adalah sebagai berikut:

- Content

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

- *Grid*

Komponen ini akan digunakan untuk membangun tata letak kustom pada halaman info. UI Component *Grid* dengan tag `<ion-grid>` 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 3.39). Sejak Ionic Framework versi 4, terjadi perubahan dengan mengganti tag tersebut menjadi `<ion-button>` pada aplikasi yang akan dibangun yang menggunakan Ionic 6 (Kode 3.40).

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

Kode 3.39: *Button* dengan Ionic 3 di Aplikasi WSDC 2017 Bali Saat Ini

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

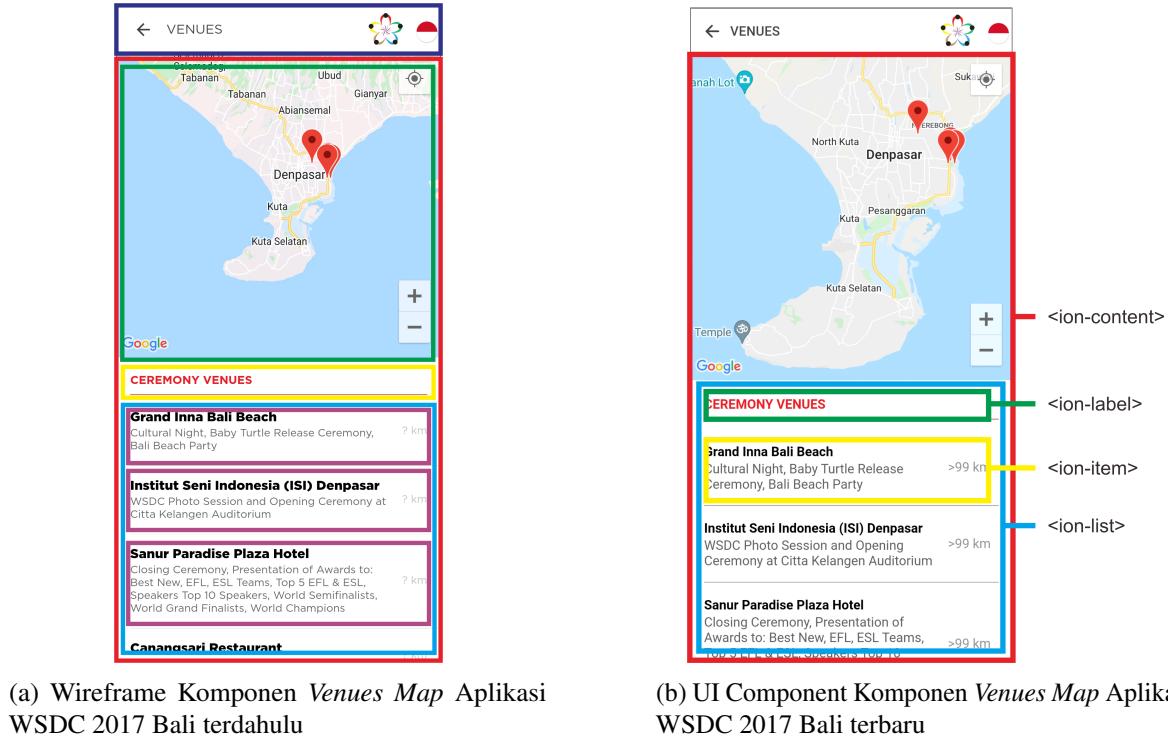
Kode 3.40: *Button* dengan Ionic 6 di Aplikasi WSDC 2017 Bali yang Akan Dibuat

- *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. Komponen *Venues Map*

Komponen *Venues Map* digunakan untuk menampilkan halaman *Venues Map* yang berisi peta untuk masing-masing lokasi *venue*, serta jarak dari posisi pengguna ke tiap-tiap lokasi *venue*.



Gambar 3.9: Komponen *Venues Map* pada Aplikasi WSDC 2017 Bali

### (a) Analisis pada Aplikasi WSDC 2017 Bali terdahulu

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 3.41). 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 3.41: `@Component` pada *venues\_map.ts*

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

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

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

- 8 • ngAfterViewInit()

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

- 12 • loadMap()

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

- 22 • loadMarkers()

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

- 27 • featTapped(event, index)

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

- 32 • computeDistance(p1, p2)

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

37 Terdapat *file* venues\_map.html yang digunakan untuk menampilkan halaman *venues map*. Ter-  
38 dapat beberapa komponen yang disediakan oleh Ionic Framework, yang diimplementasikan ke  
39 dalam halaman. Diantaranya adalah sebagai berikut:

1     • *Header*

2     Halaman *venues* memiliki *header* dengan tag `<ion-header>` (Kode 3.42) seperti pada  
 3     gambar 3.9a. Tag tersebut merupakan komponen *parent* yang menampung komponen navbar  
 4     seperti yang ditandai dengan kotak berwarna biru. Di dalam navbar, terdapat sebuah tag  
 5     `<button>` untuk memunculkan *sidemenu* dan `<ion-icon>` untuk menampilkan icon.  
 6     Terdapat tag `<ion-title>` sebagai judul dari halaman, yaitu “Venues”.

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

Kode 3.42: *Header* pada *venues\_map.html*

17    • *Content*

18    *Content* pada halaman *venues* dibungkus oleh tag `<ion-content>` (Kode 3.43) yang pada  
 19    gambar 3.8a dengan kotak berwarna merah. Di dalam tag `<ion-content>` terdapat  
 20    sebuah tag `<div>` dengan id bernilai *map*, untuk menampilkan peta lokasi dari *venues*  
 21    seperti yang ditandai dengan kotak berwarna hijau pada gambar 3.9a. Untuk judul dari *venues*  
 22    ditandai dengan kotak berwarna kuning dengan menggunakan tag `<h3>`. Terdapat sebuah  
 23    tag `<ion-scroll>` seperti yang ditandai dengan kotak berwarna biru muda, berfungsi  
 24    untuk menampilkan sebuah konten yang dapat digulir. Di dalam tag `<ion-scroll>`  
 25    terdapat sebuah tag `<ion-list>` dan `<ion-item>` seperti yang ditandai dengan kotak  
 26    berwarna ungu, berisi nama, deskripsi, serta jarak pengguna ke tempat *venues* berada. Tag  
 27    `<ion-item>` akan melakukan perulangan dengan menggunakan `*ngFor` yang tersedia  
 28    pada Angular untuk menampilkan daftar *venues*.

```
29  1 <ion-content>
30  2   <div #map id="map"></div>
31  3   <h3 #pagetitle>
32  4     {{selectedItem.name}}
33  5   </h3>
34  6   <ion-scroll scrollY="true">
35  7     <ion-list>
36  8       <ion-item text-wrap *ngFor="let feature of items; let i=index" (
37  9         click)="featTapped($event, i)">
38 10         <h2>{{feature.name}}</h2>
39 11         <p>{{feature.description}}</p>
40 12         <ion-note item-end>
41 13           {{feature.distance}}
42 14         </ion-note>
43 15       </ion-item>
44 16     </ion-list>
45 17   </ion-scroll>
46 18 </ion-content>
```

Kode 3.43: *Content* pada *venues\_map.html*

1                   (b) Perancangan Aplikasi WSDC 2017 Bali terbaru

2                   Pada komponen *venues map*, terdapat beberapa UI Component yang akan diimplementasikan  
 3                   seperti pada gambar 3.9b, diantaranya adalah sebagai berikut:

4                   • Content

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

9                   • Scroll

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

13                  • List

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

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

24                  Dengan digunakannya Capacitor, maka akan digunakan Capacitor *plugin* diantaranya yaitu:

25                  • Capacitor Browser API

26                  Plugin ini berfungsi untuk menjalankan kemampuan *in-app browser* pada aplikasi WSDC 2017  
 27                  Bali, yaitu untuk membuka berita terkait acara WSDC 2017 Bali. Untuk membuka *in-app browser*,  
 28                  digunakan *method* open() dengan properi url sebagai url yang dituju, seperti pada kode 3.44.

```
29
30 1 launch(newsUrl: string) {
31 2   Browser.open({ url: newsUrl });
32 3 }
```

Kode 3.44: *Method open()* Pada Browser API

34                  • Capacitor Community Google Maps

35                  Plugin ini berfungsi untuk menampilkan peta Google Maps secara *native* pada perangkat pengguna.  
 36                  Peta tersebut berisi lokasi dari seluruh acara WSDC 2017 Bali dilaksanakan yang ditandai dengan  
 37                  marker, dan juga menampilkan posisi dari perangkat pengguna. Seperti yang sudah dijelaskan pada sub  
 38                  bab 2.3.1.1, Capacitor Community Google Maps membutuhkan sebuah kontainer pada HTML untuk  
 39                  ditempati. Contoh kontainer tersebut seperti pada gambar 3.10a yang ditandai dengan kotak berwarna  
 40                  merah. Kode HTML untuk kontainer seperti pada kode 3.45 dibuat dengan ukuran lebar sesuai dengan  
 41                  layar ponsel pengguna, dan tinggi sebesar 400px. Kemudian mengisi *option* boundingRect pada  
 42                  *method* createMap() seperti pada kode 3.46 dengan nilai yang diambil dari kontainer pada HTML yang  
 43                  sudah dibuat sebelumnya. Dengan begitu, Google Maps sudah memiliki tempat di dalam kontainer

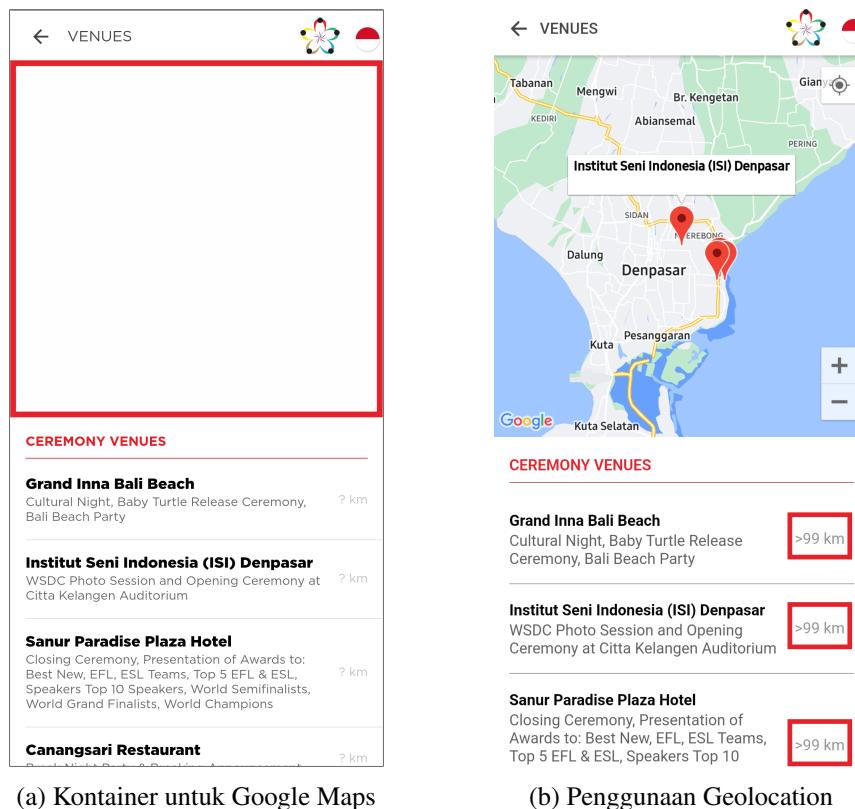
1 pada HTML. Peta Google Maps menampilkan lokasi awal Kuta, Bali dengan memasukan *latitude*  
2 dan *longitude* dari Kuta, Bali ke dalam *option cameraPosition* properti target. Selain properti target,  
3 digunakan juga properti zoom yang digunakan untuk seberapa besar peta ditampilkan. Semakin besar  
4 nilai zoom, maka peta ditampilkan semakin dekat, begitu pula sebaliknya.

```
5 1 <div id="container">  
6 2   <div id="mapContainer"></div>  
7 3 </div>
```

Kode 3.45: Kontainer pada HTML untuk Peta Capacitor CommunityGoogle Maps

```
10 1 const result = await CapacitorGoogleMaps.createMap({  
11 2   boundingRect: {  
12 3     width: Math.round(boundingRect.width),  
13 4       height: Math.round(boundingRect.height),  
14 5       x: Math.round(boundingRect.x),  
15 6       y: Math.round(boundingRect.y),  
16 7     },  
17 8     cameraPosition: {  
18 9       target: {  
20 10         latitude: -8.722396,  
21 11           longitude: 115.17671,  
22 12         },  
23 13       zoom: 11,  
24 14     },  
25 15     preferences: {  
26 16       controls: {  
27 17         isCompassButtonEnabled: true,  
28 18           isMyLocationButtonEnabled: true,  
29 19             isZoomButtonsEnabled: true,  
30 20         },  
31 21       appearance: {  
32 22         isMyLocationDotShown: true,  
33 23       },  
34 24     },  
35 25   });
```

Kode 3.46: Method *createMap()* Pada Capacitor Community Google Maps



Gambar 3.10: Penggunaan Community Google Maps dan Geolocation pada Aplikasi WSDC 2017 Bali

Pada Capacitor Community Google Maps terdapat *option* preferences seperti pada kode 3.46 yang digunakan untuk menambah fungsi pada Google Maps, diantaranya yaitu:

1. `isCompassButtonEnabled`: Digunakan untuk menampilkan tombol kompas sebagai petunjuk arah mata angin.
2. `isMyLocationButtonEnabled`: Digunakan untuk menampilkan tombol gps. Ketika tombol tersebut ditekan, posisi peta mengarah pada posisi ponsel pengguna. Penggunaan `isMyLocationButtonEnabled` adalah pada gambar 3.11a yang ditunjukan dengan kotak berwarna hijau.
3. `isZoomButtonsEnabled`: Digunakan untuk menampilkan tombol *zoom in* dan *zoom out*. Ketika tombol tersebut ditekan, maka peta akan melakukan *zoom in* dan *zoom out* sesuai dengan tombol yang ditekan. Penggunaan `isMyLocationButtonEnabled` adalah pada gambar 3.11a yang ditunjukan dengan kotak berwarna kuning.
4. `isMyLocationDotShown`: Digunakan untuk menampilkan titik biru sebagai tanda lokasi pengguna saat ini. Titik ini akan muncul ketika peta mengarah pada posisi ponsel pengguna. Penggunaan `isMyLocationButtonEnabled` adalah pada gambar 3.11b yang ditunjukan dengan kotak berwarna biru.

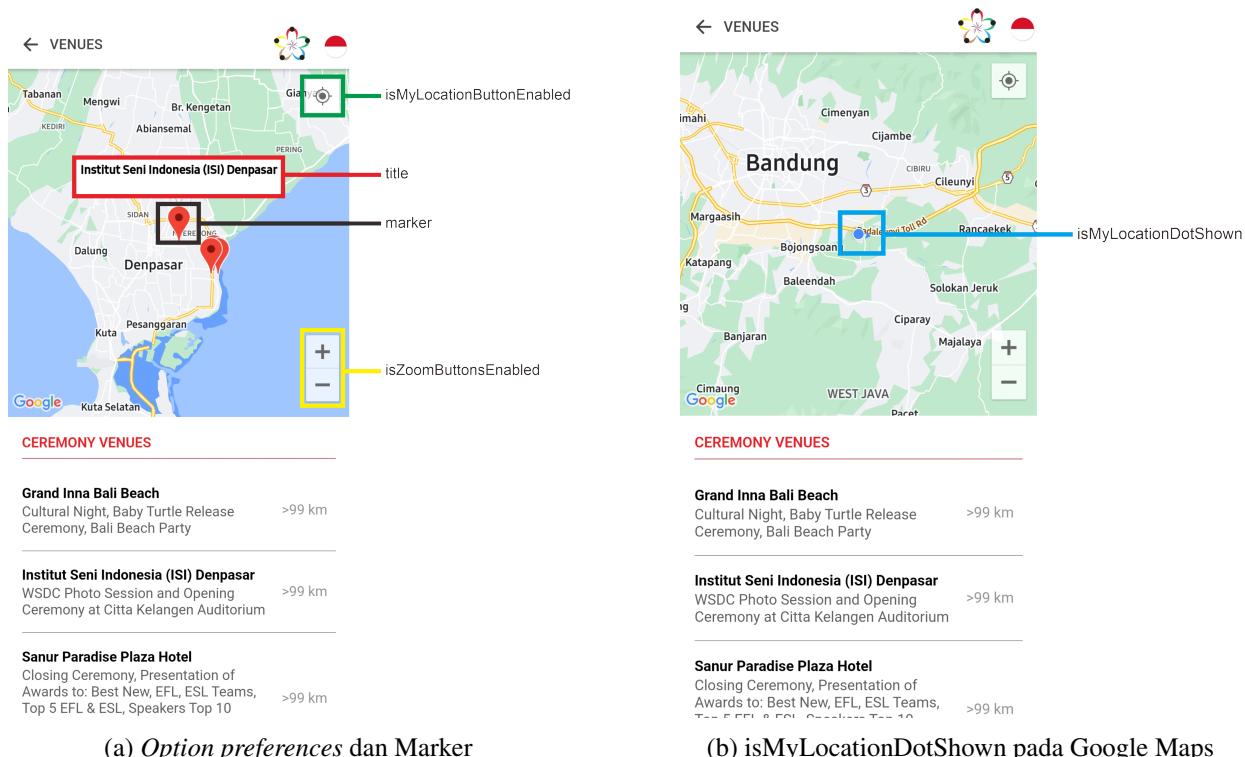
Di dalam peta Google Maps menampilkan *marker* untuk setiap lokasi *venue* seperti pada gambar 3.11a yang ditandai dengan kotak warna hitam. Untuk menampilkan *marker* digunakan *method* `addMarker()` seperti pada kode 3.47. *Marker* ditempatkan pada lokasi *venue* sesuai dengan koordinat *latitude* dan *longitude* masing-masing *venue*. Lokasi tersebut ditambahkan pada *option* `position`. Terdapat juga *option* preferences yang memiliki properti `title`, digunakan untuk menampilkan nama *venue* diatas *marker* ketika *marker* ditekan seperti pada gambar 3.11a yang ditandai dengan kotak berwarna merah.

```

1 CapacitorGoogleMaps.addMarker({
2   mapId: result.googleMap.mapId,
3   position: {
4     latitude: koor[1],
5     longitude: koor[0],
6   },
7   preferences: {
8     title: venuesMarker.properties.Name,
9   },
10 });
11 );

```

Kode 3.47: Method addMarker() Pada Capacitor Community Google Maps



Gambar 3.11: Penggunaan Community Google Maps pada Aplikasi WSDC 2017 Bali

13 • Capacitor Geolocation API

14 Capacitor Geolocation API digunakan untuk mengetahui posisi dari pengguna. *Plugin* ini mendapatkan koordinat berupa *latitude* dan *longitude* ponsel pengguna berada saat ini menggunakan *Global Positioning System (GPS)* dengan menggunakan *method* *getCurrentPosition()* seperti pada kode 3.48.

15 Setelah mendapatkan koordinat ponsel pengguna, koordinat tersebut digunakan untuk menghitung

16 jarak dari lokasi pengguna ke lokasi *venue* seperti pada gambar 3.10b yang ditandai dengan kotak

17 berwarna merah. Jika lokasi pengguna dengan *venue* lebih dari 99 KM dari lokasi *venue*, maka akan

18 menampilkan >99km. Jika lokasi pengguna dengan *venue* kurang dari 99 KM, maka akan menampilkan

19 jarak yang sesuai.

20

21

```

1 const printCurrentPosition = async () => {
2   const coordinates = await Geolocation.getCurrentPosition();
3   this.userCoordinatesLat = coordinates.coords.latitude;
4   this.userCoordinatesLng = coordinates.coords.longitude;
5 };

```

Kode 3.48: *Method getCurrentPosition( Pada Geolocation API*

- Capacitor Splash Screen API

Capacitor Splash Screen API digunakan untuk menampilkan *splash screen* yang tampil pada saat aplikasi pertama kali dibuka. Seperti yang sudah dijelaskan pada sub-bab 2.3.1.1, Splash Screen secara *default* otomatis ditutup setelah 500ms sejak pertama kali Splash Screen dibuka. Namun jika hal itu terjadi, maka akan ada kemungkinan ketika Splash Screen ditutup, komponen-komponen pada aplikasi belum selesai dimuat, mengakibatkan pengguna melihat halaman putih kosong. Untuk menghindari hal tersebut, Splash Screen ditutup ketika komponen Home, komponen yang pertama kali ditampilkan, selesai dimuat dan ditampilkan ke layar. Hal tersebut dapat terjadi dengan menjalankan *method* hide() yang dimiliki Capacitor Splash Screen pada *method* ionViewDidEnter() di dalam komponen Home seperti pada kode 3.49. *Method* ionViewDidEnter() sendiri merupakan sebuah Ionic Life Cycle yang dijalankan ketika komponen selesai dijalankan dan sudah ditampilkan ke layar.

```

1 ionViewDidEnter() {
2   SplashScreen.hide()
3 }

```

Kode 3.49: *Method hide() Pada Splash Screen API*

Pada sistem usulan, digunakan *libraries* yang dimiliki Angular, yaitu Angular Routing dan Angular HTTPClient. Angular Routing digunakan untuk menangani navigasi dari satu tampilan ke tampilan lain, kemudian menampilkannya di layar. Angular HTTPClient digunakan untuk melakukan komunikasi dengan server dengan menggunakan protokol HTTP. HTTPClient mendapatkan data dari server, kemudian menyimpannya ke dalam penyimpanan. HTTPClient digunakan pada setiap komponen untuk mendapatkan data dari penyimpanan. Pada komponen Home dan Announcements digunakan juga untuk mengambil data dari server seperti pada kode 3.50 yang merupakan penggunaan HTTPClient pada komponen Home untuk sistem kini. Pertama, Ionic Storage mengambil data terlebih dahulu dari penyimpanan. Jika aplikasi pertama kali dijalankan setelah instalasi, maka data tersebut tidak akan ada di dalam penyimpanan. Jika data tidak ada data pada penyimpanan, HTTPClient mengambil data dari aset lokal dengan *method* get(), dan menyimpannya ke dalam penyimpanan dengan Ionic Storage. Jika data tidak berhasil didapatkan, maka akan menampilkan sebuah Toast. Kemudian *method* get() dijalankan untuk mengambil data terbaru dari server. Data terbaru tersebut kemudian menggantikan data sebelumnya yang didapat dari lokal aset. Mengambil data pertama kali dari aset lokal bertujuan agar jika pengguna tidak terhubung ke koneksi internet pada saat pengguna melakukan instalasi aplikasi, aplikasi tetap dapat dibuka, dijalankan, dan menampilkan data-data yang seharusnya ditampilkan. Pengguna tetap mendapatkan data terbaru dari server ketika pengguna terhubung ke koneksi internet, karena *method* get pada HTTPClient selalu dijalankan pada saat pengguna membuka aplikasi untuk selalu mendapatkan data terbaru dari server.

```
1 this.storage.get('wsdcDataStorage').then((data) => {
2     if(data == null){
3         this.http.get('../assets/json/wsdc_data.json').subscribe((data: any) => {
4             this.wsdcData = data;
5             this.storage.set('wsdcDataStorage',data);
6         },
7         error => {
8             this.showToast('Failed to refresh information from local storage');
9         });
10    }else{
11        this.wsdcData = data;
12    }
13    setTimeout(() => {
14        this.http.get('https://wsdc.dnartworks.com/wsdc_data.json')
15        .subscribe((data: any) => {
16            this.storage.set('wsdcDataStorage', data);
17            this.wsdcData = data;
18        },
19        error => {
20            this.showToast('Failed to refresh information');
21        });
22    }, 1000);
23 })
24 })
```

Kode 3.50: HttpClient pada Komponen Home

Perubahan-perubahan yang terjadi dari aplikasi WSDC 2017 Bali terdahulu ke aplikasi WSDC 2017 Bali terbaru meliputi perubahan UI Component, Native API dan penggunaan *plugin* dari Native API yaitu seperti pada tabel 3.9. Selain hal-hal yang berkaitan dengan perubahan-perubahan tersebut, pembuatan aplikasi WSDC 2017 Bali dengan Ionic 6 menggunakan aset yang sama dengan aplikasi WSDC 2017 Bali terdahulu, seperti gambar, file HTML, kelas TypeScript, file SCSS untuk setiap komponen, dan kode warna.

Tabel 3.9: Tabel Perubahan yang Dilakukan dari Ionic 3 ke Ionic 6

Jenis Perubahan	Perubahan	
	Ionic 3	Ionic 6
UI Component	<ion-item>	Menambahkan <ion-label>
	<ion-list-header>	Menambahkan <ion-label>
	<button>	<ion-button>
	<ion-segment-button>	Menambahkan <ion-label>
	<ion-scroll>	Dihapus, hanya menggunakan <ion-content> sudah cukup
Native API	Cordova	Capacitor
Plugin Native API	Cordova Google Maps API	Capacitor Community Google Maps API
	Cordova Geolocation	Capacitor Geolocation API
	Cordova Splash Screen	Capacitor Splash Screen API
	Cordova In-App Browser	Capacitor Browser API

## 3.2 Tantangan Pengembangan Sistem Usulan

Saat sedang melakukan proses migrasi aplikasi WSDC 2017 Bali dari Ionic Framework versi 3 ke Ionic Framework versi 6, 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 6 terlebih dahulu melakukan migrasi dari Ionic Framework versi 3 ke Ionic Framework versi 4, yang selanjutnya dari Ionic versi 4 ke Ionic versi 5 dan dari Ionic versi 5 ke Ionic versi 6. 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 penggeraan skripsi, halaman Draw dan Result pada aplikasi WSDC 2017 Bali tidak dapat diakses karena terjadi kesalahan konfigurasi pada server. Kemudian 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 penggeraan 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<sup>1</sup>. Maka dari itu, atas saran dan masukan dosen pembimbing dan dosen pengujii, maka peneliti melakukan migrasi tidak lagi sampai Ionic versi 5, melainkan sampai dengan Ionic versi 6 untuk masa dukungan Ionic Framework yang lebih lama.
- Pada Capacitor Community Google Maps, terdapat *option preferences isCompassButtonEnabled* yang digunakan untuk menampilkan tombol kompas pada peta. Namun setelah aplikasi dijalankan, tombol tersebut tidak muncul pada layar. Maka hal ini menyebabkan adanya perbedaan tampilan pada peta antara aplikasi WSDC 2017 Bali terdahulu dan terbaru, yaitu tidak ada tombol kompas pada aplikasi WSDC 2017 Bali terbaru.

---

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

1

## BAB 4

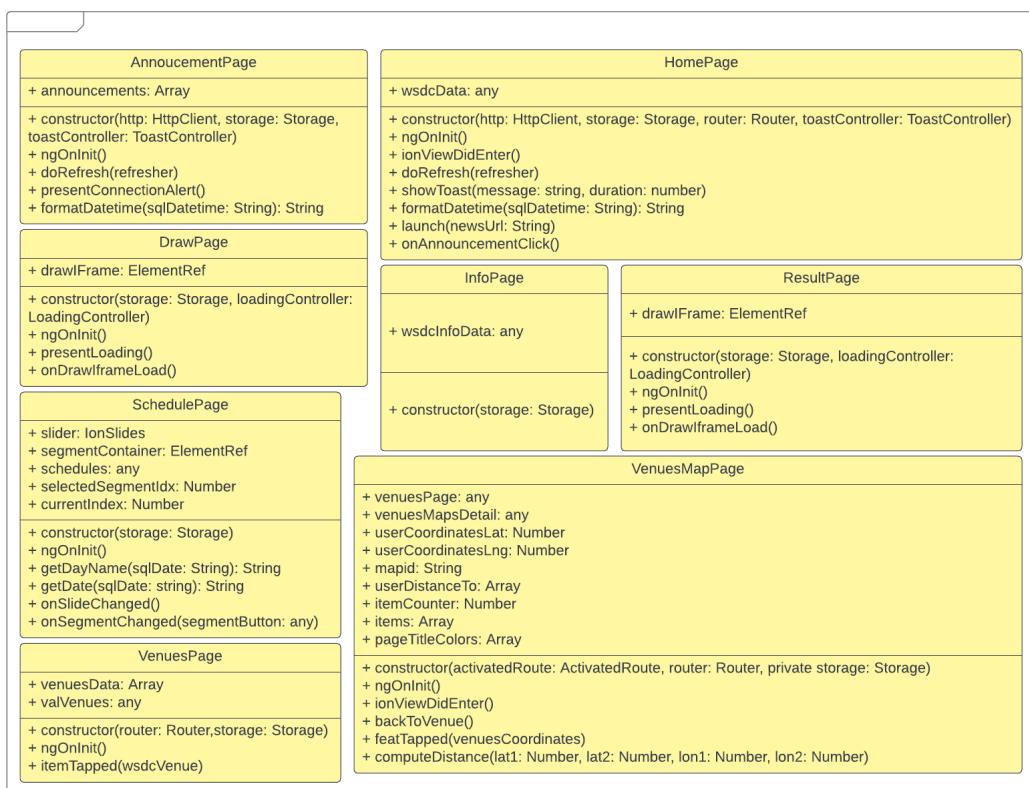
2

# PERANCANGAN

- 3 Pada bab ini menjelaskan mengenai perancangan aplikasi yang akan dibangun meliputi perancangan kelas pada masing-masing komponen beserta dengan deskripsi dan fungsinya, dan perancangan struktur HTML.
- 5 Untuk antarmuka pada aplikasi yang dibangun memiliki antarmuka yang sama dengan aplikasi WSDC 2017 Bali terdahulu. Penjelasan terkait dengan antarmuka telah dibahas pada bagian 3.1.

### 7 4.1 Perancangan Kelas

- 8 Pada aplikasi WSDC 2017 Bali yang akan dibangun menggunakan struktur kelas yang sama dengan aplikasi WSDC 2017 Bali terdahulu, dengan beberapa penyesuaian terkait dengan pembaruan yang dilakukan.
- 10 Diagram kelas secara keseluruhan dapat dilihat pada gambar 4.1.



Gambar 4.1: Diagram Kelas Keseluruhan

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

2 1. Komponen *Announcements*

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

- 9 • **constructor(private http: HttpClient, private storage: Storage, public toastController: ToastController)**

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

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

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

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

14 **Kembalian**: tidak ada.

- 15 • **ngOnInit()**

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

18 **Parameter**: tidak ada.

19 **Kembalian**: tidak ada.

- 20 • **doRefresh(refresher)**

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

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

29 **Kembalian**: tidak ada.

- 30 • **presentConnectionAlert()**

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

34 **Parameter**: tidak ada.

35 **Kembalian**: tidak ada.

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

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

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

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

6     2. Komponen *Draw*

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

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

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

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

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

16       – **loadingController:** Parameter ini digunakan untuk menyimpan API LoadingController.

17       **Kembalian:** tidak ada.

18     • **ngOnInit()**

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

22       **Parameter:** tidak ada.

23       **Kembalian:** tidak ada.

24     • **async presentLoading()**

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

26       **Parameter:** tidak ada.

27       **Kembalian:** tidak ada.

28     • **onDrawIframeLoad()**

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

32       **Parameter:** tidak ada.

33       **Kembalian:** tidak ada.

34     3. Komponen *Home*

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

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

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

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

- 2       – **http**: Parameter ini digunakan untuk menyimpan API HttpClient.
- 3       – **storage**: Parameter ini digunakan untuk menyimpan API Storage.
- 4       – **router**: Parameter ini digunakan untuk menyimpan API Router.
- 5       – **toastController**: Parameter ini digunakan untuk menyimpan API ToastController.

6       **Kembalian:** tidak ada.

7       • **ionViewDidEnter()**

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

10      **Parameter:** tidak ada.

11      **Kembalian:** tidak ada.

12      • **ngOnInit()**

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

20      **Parameter:** tidak ada.

21      **Kembalian:** tidak ada.

22      • **doRefresh(refresher)**

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

29      **Parameter:** tidak ada.

30      **Kembalian:** tidak ada.

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

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

34      **Parameter:**

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

37      **Kembalian:** tidak ada.

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

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

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

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

- 1     • **launch(newsUrl: string)**
- 2         *Method* ini berfungsi untuk membuka url berita sesuai dengan url yang ada di dalam parameter.
- 3         **Parameter:** newsUrl: *string* url dari sebuah berita.
- 4         **Kembalian:** tidak ada.
- 5     • **onAnnouncementClick()**
- 6         *Method* ini berfungsi untuk berpindah halaman ke halaman *announcements*.
- 7         **Parameter:** tidak ada.
- 8         **Kembalian:** tidak ada.

#### 9     4. Komponen Info

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

#### 17    5. Komponen *Result*

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

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

- 24     • **constructor(private storage: Storage, public loadingController: LoadingController)**
- 25         *Method* constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat. **Parameter:**
  - 26             – **storage:** Parameter ini digunakan untuk menyimpan API Storage.
  - 27             – **loadingController:** Parameter ini digunakan untuk menyimpan API LoadingController.
- 28         **Kembalian:** tidak ada.
- 29     • **ngOnInit()**
- 30         *Method* ini berfungsi untuk mengambil data *result* yang terdapat di dalam *storage*. Data tersebut  
31         lalu disimpan ke dalam *child* resultIFrame. Setelah itu *method* ini akan memanggil *method*  
32         presentLoading().
- 33         **Parameter:** tidak ada.
- 34         **Kembalian:** tidak ada.
- 35     • **async presentLoading()**
- 36         *Method* ini berfungsi untuk menampilkan sebuah indikator *loading* dengan pesan “Please wait...”.
- 37         **Parameter:** tidak ada.
- 38         **Kembalian:** tidak ada.

1     • **onResultIframeLoad()**

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

5       **Parameter:** tidak ada.

6       **Kembalian:** tidak ada.

7     6. Komponen *Schedule*

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

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

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

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

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

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

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

26       **Kembalian:** tidak ada.

27     • **ngOnInit()**

28       *Method* ini berfungsi untuk mengambil data *schedule* yang terdapat di dalam *storage*. Data  
 29       tersebut lalu disimpan ke dalam atribut *schedules*.

30       **Parameter:** tidak ada.

31       **Kembalian:** tidak ada.

32     • **getDayName(sqlDate: string)**

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

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

35       **Kembalian:** *string* nama hari.

36     • **getDate(sqlDate: string)**

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

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

39       **Kembalian:** *string* tanggal.

40     • **onSlideChanged()**

41       *Method* ini dipanggil saat *slides* dipindahkan dengan cara digeser ke kanan atau ke kiri. *Method*  
 42       ini akan mengubah atribut currentIndex menjadi index *slides* saat ini, kemudian mengubah atribut

selectedSegmentIdx menjadi index *slides* saat ini. Hal ini bertujuan agar indeks dari *segment* yang aktif dapat diganti sesuai dengan indeks *slides* yang aktif. Dengan begitu tampilan *segment* dan *slides* yang aktif akan sesuai.

**Parameter:** tidak ada.

**Kembalian:** tidak ada.

- **onSegmentChanged(segmentButton)**

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

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

**Kembalian:** tidak ada.

## 7. Komponen *Venues*

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

Kelas ini memiliki beberapa atribut, yaitu:

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

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

Kelas ini memiliki beberapa *method*, diantaranya yaitu:

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

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

**Parameter:**

- **router:** Parameter ini berfungsi untuk menyimpan API Router.

- **storage:** Parameter ini berfungsi untuk menyimpan API Storage.

**Kembalian:** tidak ada.

- **ngOnInit()**

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

**Parameter:** tidak ada.

**Kembalian:** tidak ada.

- **itemTapped(wsdcVenue)**

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

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

**Kembalian:** tidak ada.

1    8. Komponen *Venues Maps*

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

- 5    • **venuesPage** : Atribut ini digunakan untuk menyimpan data yang dikirimkan dari komponen  
6    *Venues*.
- 7    • **venuesMapsDetail** : Atribut ini digunakan untuk menyimpan data *venues* dari *storage* sesuai  
8    dengan kategori yang sedang dipilih.
- 9    • **userCoordinatesLat** : Atirbut ini digunakan untuk menyimpan koordinat latitude dari perangkat  
10   pengguna.
- 11   • **userCoordinatesLng** : Atirbut ini digunakan untuk menyimpan koordinat longitude dari perang-  
12   kat pengguna.
- 13   • **mapid** : Atribut ini digunakan untuk menyimpan id dari peta.
- 14   • **userDistanceTo** : Atribut ini digunakan untuk menyimpan jarak dari posisi perangkat pengguna  
15   ke posisi *venues*.
- 16   • **itemCounter** : Atribut ini digunakan untuk menyimpan berapa banyak *venues* dalam sebuah  
17   kategori.
- 18   • **items** : Atribut ini digunakan untuk menyimpan id dan id warna dari sebuah kategori *venues*.
- 19   • **pageTitleColors** : Atribut ini merupakan sebuah *array* yang berisi kode warna untuk masing-  
20   masing judul kategori *venues*.

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

- 22   • **constructor(private activatedRoute: ActivatedRoute, private router: Router, private stor-  
23   age: Storage)**

25   *Method* constructor berfungsi untuk memberikan nilai awal pada saat kelas dibuat. *Constructor*  
26   juga digunakan untuk mengambil data dari *storage* dan memasukannya ke atribut *venuesMapsDe-  
27   tail*.

28   **Parameter:**

- 29   – **activatedRoute** : Parameter ini digunakan untuk menyimpan API *ActivatedRoute*.
- 30   – **router** : Parameter ini digunakan untuk menyimpan API *Router*.
- 31   – **storage** : Parameter ini digunakan untuk menyimpan API *Storage*.

32   **Kembalian:** tidak ada.

- 33   • **ngOnInit()**

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

36   **Parameter:** tidak ada.

37   **Kembalian:** tidak ada.

- 38   • **ionViewDidEnter()**

39   *Method* ini dijalankan ketika halaman sudah masuk sepenuhnya. *Method* ini digunakan untuk  
40   menginisialisasi peta menggunakan *plugin* Google Maps yang disediakan oleh Capacitor. Peta  
41   tersebut menampilkan Pulau Bali, lebih tepatnya di Kecamatan Kuta dengan latitude -8.722396  
42   dan longitude 115.17671. *Method* ini juga membuat *marker* yang menandai setiap lokasi *venues*

1 pada satu kategori *venues*. *Method* ini juga digunakan untuk menyimpan jarak antara posisi  
2 perangkat pengguna ke masing-masing posisi *venues*, yang kemudian disimpan ke dalam atibut  
3 *userDistanceTo*.

4 **Parameter:** tidak ada.

5 **Kembalian:** tidak ada.

6 • **backToVenue()**

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

8 **Parameter:** tidak ada.

9 **Kembalian:** tidak ada.

10 • **featTapped(venuesCoordinates)**

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

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

15 **Kembalian:** tidak ada.

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

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

19 **Parameter:**

- 20 – **lat1**: koordinat latitude dari pengguna.
- 21 – **lat2**: koordinat latitude dari *venues*.
- 22 – **lon1**: koordinat longitude dari pengguna.
- 23 – **lon2**: koordinat longitude dari *venues*.

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

25 

## 4.2 Perancangan Struktur HTML

26 Struktur HTML pada masing-masing komponen mengambil struktur yang sama dengan aplikasi WSDC  
27 2017 Bali terdahulu, namun dengan beberapa perubahan. Perubahan-perubahan tersebut telah dibahas pada  
28 bagian [2.3.3](#), serta analisis penggunaannya di sistem usulan pada bagian [3.1](#).

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

31 1. *Header*

32 *Header* untuk setiap komponen pada umumnya memiliki struktur yang serupa namun dibedakan dengan  
33 judul dari setiap halaman. *Header* digunakan untuk menampilkan judul dari sebuah halaman, serta  
34 menyediakan sebuah *menu button* sebagai salah satu cara untuk membuat *sidemenu* untuk melakukan  
35 navigasi antar halaman. *Header* dibungkus oleh tag `<ion-header>` yang didalamnya terdapat tag  
36 `<ion-toolbar>` yang disediakan Ionic Framework. Kemudian untuk *menu button* dibuat oleh tag  
37 `<ion-menu-button>` pada tag `<ion-buttons>`. Selanjutnya untuk judul dari sebuah halaman  
38 dibungkus oleh tag `<ion-title>`. Judul akan berbeda beda tergantung dengan halamannya. Salah  
39 satu contoh dari penggunaan *header* adalah pada gambar [3.4a](#) yang ditandai dengan kotak berwarna  
40 biru.

1    2. *Content*

2    *Content* untuk setiap komponen memiliki struktur yang berbeda-beda tergantung dengan isi dari halaman tersebut. Namun *content* untuk setiap komponen dibungkus oleh sebuah *tag* <ion-content>. Salah satu contoh dari penggunaan *content* adalah pada gambar 3.4a yang ditandai dengan warna merah. Struktur *content* untuk masing-masing halaman adalah sebagai berikut:

3    • Halaman *Announcements*

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

5    • Halaman *Draw*

6    *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 3.3a yang ditandai dengan kotak berwarna hijau.

7    • Halaman *Home*

8    *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 3.4a yang ditandai dengan kotak berwarna hijau. Terdapat sebuah *card* dengan *tag* <ion-card> seperti yang ditandai dengan kotak berwarna merah muda yang digunakan untuk menampilkan sebuah pengumuman terbaru, berikut dengan waktu dan tanggal serta pesan dari pengumuman tersebut. Di dalam *card* terdapat *grid* untuk *layout* dari *card*. Di dalam sebuah *grid* terdapat sebuah baris dengan *tag* <ion-row>. Di dalam baris tersebut terdapat dua buah kolom dengan *tag* <ion-col>. Masing-masing kolom memiliki ukuran, kolom pertama yang ditandai dengan kotak berwarna coklat berukuran sembilan digunakan untuk menyimpan *header* dari *card* dengan *title* yaitu “Latest Announcement”. *Header* ini dibungkus di dalam *tag* <ion-card-header>, dan *title* dengan *tag* <ion-card-title>. Selain *header* untuk *card*, terdapat pula *content* untuk *card* dengan *tag* <ion-card-content> yang berisi waktu dan tanggal, serta pesan dari pengumuman. Untuk kolom selanjutnya dengan ukuran tiga berisi sebuah gambar seperti yang ditandai dengan kotak berwarna jingga.

9    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 3.4a. Di dalam list terdapat sebuah *header* dengan *tag* <ion-list-header> yang berisi judul dari list yaitu “Newsletters” yang berada di dalam *tag* <ion-label>. Untuk masing-masing *thumbnail* berita berada di dalam *tag* <ion-item>. Untuk masing-masing item

terdapat sebuah gambar *thumbnail* dari sebuah berita, judul dari berita tersebut, serta sebuah tombol dengan tag `<ion-button>` yang jika ditekan akan mengarahkan pengguna untuk melihat berita tertentu sesuai dengan item yang dipilih.

- Halaman Info

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

- Halaman *Result*

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

- Halaman *Schedule*

*Content* pada halaman *schedule* berisi *segment* dengan tag `<ion-segment>` dan sebuah *slides* dengan tag `<ion-slides>`. *Segment* digunakan untuk menampilkan tanggal dan hari, serta berfungsi untuk memindahkan *slides* ke hari yang dipilih seperti yang ditandai dengan kotak berwarna hijau pada gambar 3.7a. Untuk melakukan hal tersebut, di dalam *segment* terdapat sebuah *button* dengan tag `<ion-segment-button>` yang berisi tag `<ion-label>` untuk menampung tanggal dan hari. Sedangkan *slides* digunakan untuk menampilkan jadwal acara WSDC 2017 Bali pada hari yang sesuai dengan *segment* yang terpilih seperti yang ditandai dengan kotak berwarna coklat. Untuk menampilkan jadwal menggunakan *list* dengan tag `<ion-list>` yang ditandai dengan warna merah muda. Di dalam *list* terdapat sebuah *item* dengan tag `<ion-item>`. Untuk setiap *item* berisi waktu mulai dan selesai sebuah acara dengan tag `<ion-note>` yang ditandai dengan warna ungu, serta nama acara dengan tag `<h3>` yang ditandai dengan kotak berwarna jingga dan lokasi acara dengan tag `<p>` yang ditandai dengan kotak berwarna biru muda. Nama dan lokasi acara dibungkus dengan tag `<ion-label>`.

- Halaman *Venues*

*Content* pada halaman *venues* berisi *grid* dengan tag `<ion-grid>` dengan satu baris menggunakan tag `<ion-row>`. Di dalamnya terdapat sebuah *list* dengan tag `<ion-list>`. *List* tersebut berisi tombol dengan tag `<ion-button>` yang ditandai dengan kotak berwarna hijau pada gambar 3.8a. Masing-masing tombol digunakan untuk melakukan navigasi ke halaman *venues map*. Setiap tombol berisi ikon dengan tag `<ion-icon>` yang ditandai dengan kotak berwarna biru muda pada gambar 3.8a dan sebuah tag `<span>` yang berisi nama dari *venues*, ditandai dengan kotak berwarna hitam.

- Halaman *Venues Map*

*Content* pada halaman *venues map* berisi tag `<div>` yang digunakan untuk menampung peta dari sebuah kategori *venues* seperti yang ditandai dengan kotak berwarna hijau pada gambar 3.9a. Terdapat sebuah label dengan tag `<ion-label>` yang berisi nama kategori *venues* yang ditandai dengan kotak berwarna kuning. Kemudian untuk menampilkan nama dan deskripsi sebuah *venues*, serta jarak antara pengguna dengan *venues*, menggunakan *list* dengan tag `<ion-list>` yang ditandai dengan kotak berwarna biru muda.



1

## BAB 5

2

### IMPLEMENTASI DAN PENGUJIAN

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

7

#### 5.1 Implementasi

8

##### 5.1.1 Lingkugan Implementasi

9 Implementasi perangkat lunak ini dilakukan di komputer penulis dengan spesifikasi berikut:

- 10 1. Sistem Operasi: Windows 10 version 21H2
- 11 2. Versi Android Development Kit (SDK): API 30 (Android 11 (R))
- 12 3. Versi Ionic CLI: 6.20.1
- 13 4. Versi Capacitor: 3.4.3

14

##### 5.1.2 Hasil Implementasi

15 Hasil implementasi berupa sebuah aplikasi android WSDC 2017 Bali. Sebelum halaman dimuat, ditampilkan

16 sebuah *splash screen* terlebih dahulu yang menampilkan logo WSDC, logo WSDC 2017 Bali, dan logo

17 Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi Republik Indonesia. Tangkapan layar *splash*

18 *screen* dapat dilihat pada Gambar 5.1a. Aplikasi WSDC 2017 Bali terdiri dari 8 halaman yang dapat diakses

19 melalui *sidemenu*. Tangkapan layar *sidemenu* dapat dilihat pada Gambar 5.2a. Halaman-halaman yang ada

20 pada aplikasi WSDC 2017 Bali tersebut yaitu:

- 21 1. Halaman *Home*

22 Halaman *home* menjadi halaman pertama yang dimasuki oleh pengguna di aplikasi WSDC 2017 Bali.

23 Pada halaman ini pengguna dapat melihat pengumuman terbaru terkait dengan acara WSDC 2017

24 Bali, yang berisi hari, jam, dan pesan dari pengumuman tersebut, yang dapat diklik dan mengarahkan

25 pengguna ke halaman *announcements*. Pengguna dapat melihat *headline* berita-berita terkait dengan

26 acara WSDC 2017 Bali. Untuk melihat berita tersebut secara penuh, disediakan sebuah tombol yang

27 akan mengarahkan pengguna untuk melihat dan mengunduh berita terkait acara WSDC 2017 Bali.

28 Tangkapan layar halaman *home* dapat dilihat pada Gambar 5.3a. Sebagai perbandingan, tangkapan

29 layar halaman *home* pada aplikasi WSDC 2017 Bali terdahulu dapat dilihat pada Gambar 5.3b.

1    2. Halaman *Announcements*

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

6    3. Halaman *Draw*

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

11   4. Halaman Info

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

16   5. Halaman *Result*

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

21   6. Halaman *Schedule*

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

28   7. Halaman *Venues*

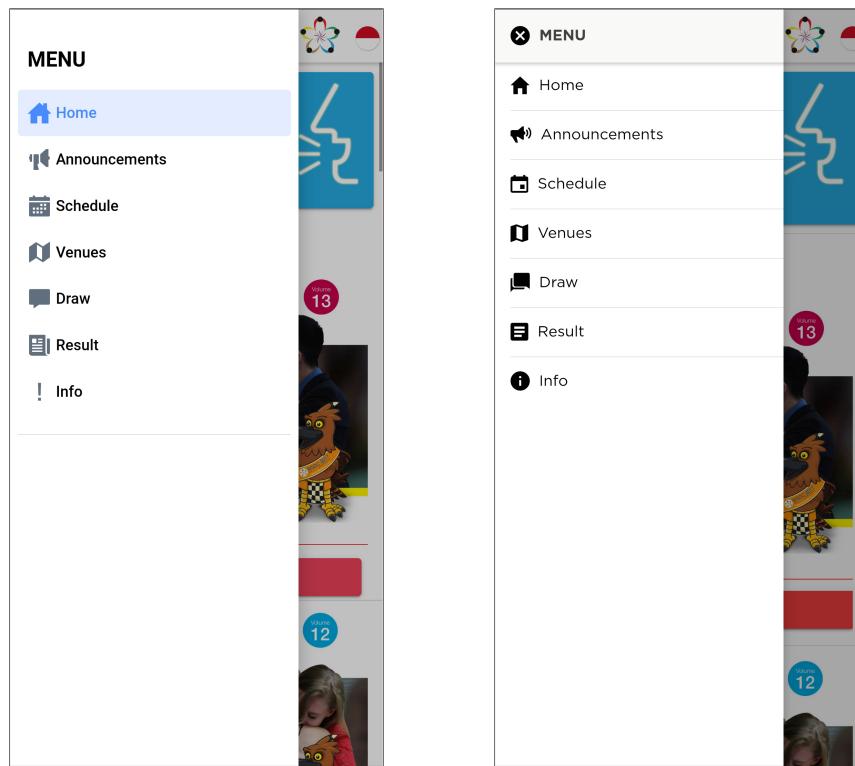
29   Halaman *venues* berisi kategori *venues* WSDC 2017 Bali. Setiap kategori yang ditampilkan merupakan  
30   sebuah tombol yang dapat diklik untuk mengarahkan pengguna ke halaman *venues map*. Tangkapan  
31   layar halaman *venues* dapat dilihat pada Gambar 5.9a. Lalu sebagai perbandingan, tangkapan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

33   8. Halaman *Venues Map*

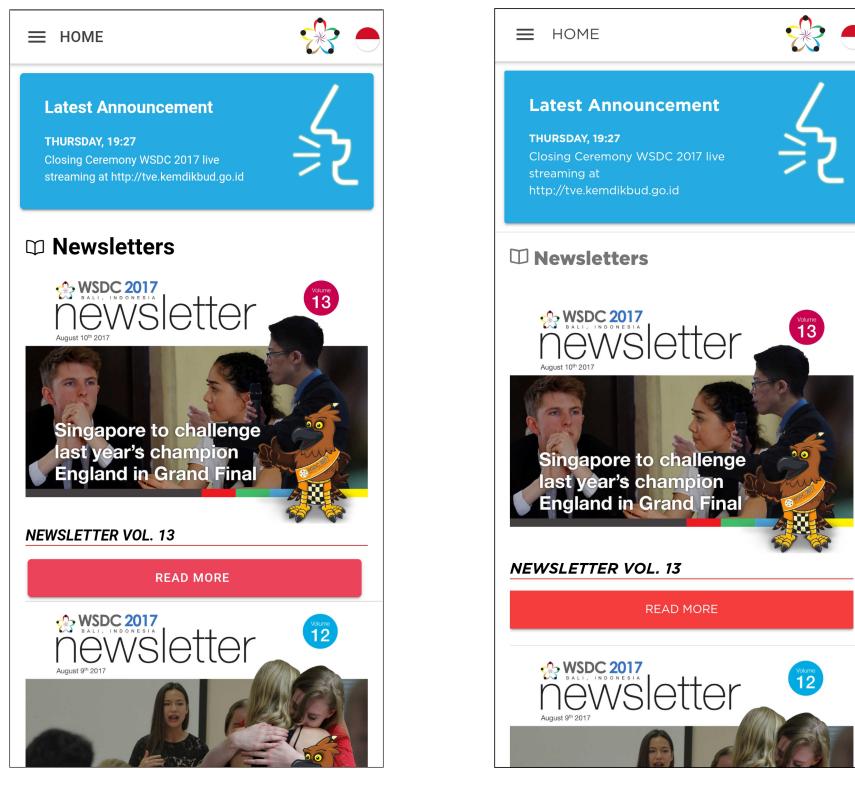
34   Halaman *venues map* berisi lokasi *venues* yang digunakan oleh WSDC 2017 Bali. Lokasi tersebut  
35   ditampilkan dengan peta, dan detail dari lokasi ditampilkan dengan *list* yang berisi nama dan  
36   lokasi *venues*, serta jarak dari pengguna ke lokasi *venues*. Tangkapan layar dari halaman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

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

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

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

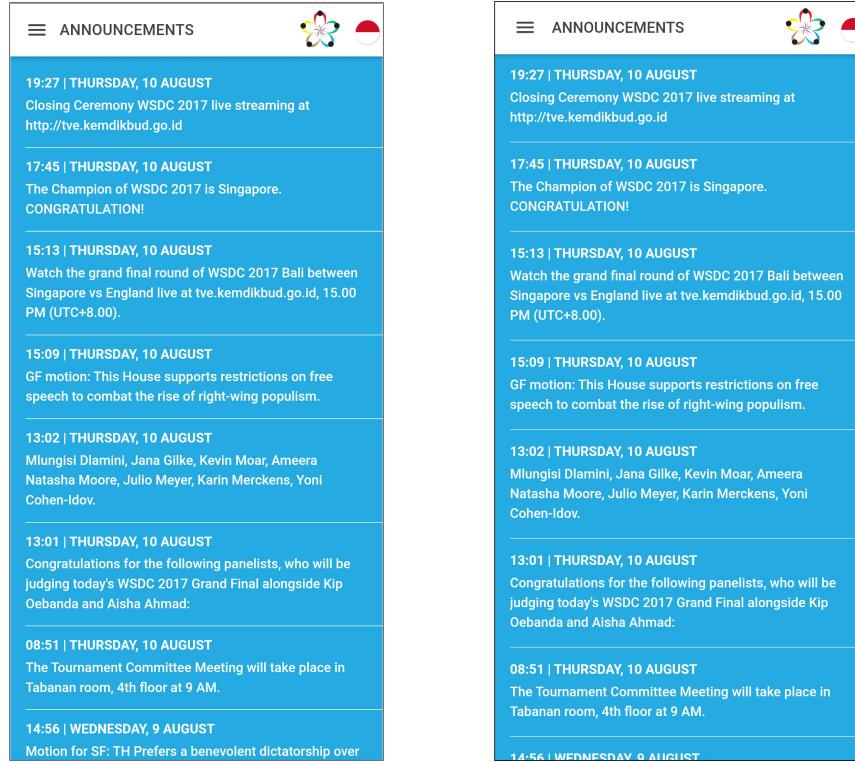
Gambar 5.2: Tangkapan Layar Sidemenu Aplikasi WSDC 2017 Bali



(a) Home Page Terbaru

(b) Home Page Terdahulu

Gambar 5.3: Tangkapan Layar Halaman Home Aplikasi WSDC 2017 Bali



(a) Announcements Page Terbaru

(b) Announcements Page Terdahulu

Gambar 5.4: Tangkapan Layar Halaman Announcements Aplikasi WSDC 2017 Bali

**(a) Draw Page Terbaru**

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

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

**(b) Draw Page Terdahulu**

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

Gambar 5.5: Tangkapan Layar Halaman Draw Aplikasi WSDC 2017 Bali

**(a) Info Page Terbaru**

CO-CONVENORS
wsdc.indonesia@kemdikbud.go.id
Ravio Patra
Kristi Ardiana
Rachmat Nur Cahyo
Nyoman Radjin
Hari Soegiharto
Fonda Ambita Sari

COMMON INDONESIAN PHRASES
I - Saya
You - Kamu
We - Kami
They - Mereka
He / She - Dia
Welcome - Selamat Datang
Hello (general greeting) - Apa kabar?
Hello (on phone) - Halo
How are you? - Apa kabar?
Reply to 'How are you?' - Baik

**(b) Info Page Terdahulu**

CO-CONVENORS
wsdc.indonesia@kemdikbud.go.id
Ravio Patra
Kristi Ardiana
Rachmat Nur Cahyo
Nyoman Radjin
Hari Soegiharto
Fonda Ambita Sari

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

Gambar 5.6: Tangkapan Layar Halaman Info Aplikasi WSDC 2017 Bali

**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**

**Octofinals**  
(8 August 2017)

Peru def. Malaysia 4-1  
South Africa def. Denmark 4-1  
Canada def. Philippines 3-2  
India def. China 5-0  
Singapore def. Pakistan 3-2  
England def. Hong Kong 5-0  
Australia def. Greece 4-1  
USA def. Korea 4-1

(a) Result Page Terbaru

**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**

**Octofinals**  
(8 August 2017)

Peru def. Malaysia 4-1  
South Africa def. Denmark 4-1  
Canada def. Philippines 3-2  
India def. China 5-0  
Singapore def. Pakistan 3-2  
England def. Hong Kong 5-0  
Australia def. Greece 4-1  
USA def. Korea 4-1

(b) Result Page Terdahulu

Gambar 5.7: Tangkapan Layar Halaman *Result* Aplikasi WSDC 2017 Bali

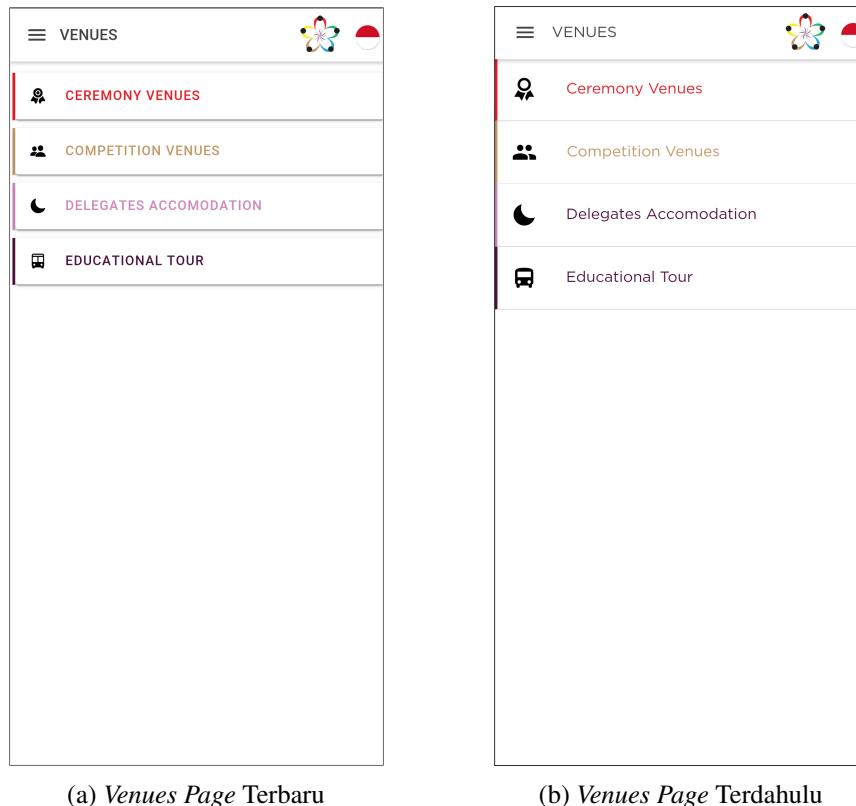
TUE	WED	THU	FRI	SAT	SUN	MON
1	2	3	4	5	6	
<b>00:00</b> <b>Arrival in Bali</b> <b>23:59</b> International Arrival at I Gusti Ngurah Rai Airport						
<b>08:00</b> <b>Team Registration</b> <b>22:00</b> Lobby at Sanur Paradise Plaza Hotel						
<b>14:00</b> <b>Hotel Check-In</b> <b>23:59</b> Lobby at Sanur Paradise Plaza Hotel						
<b>17:30</b> <b>Dinner</b> <b>21:00</b> Griya Agung Ballroom at Sanur Paradise Plaza Hotel						

(a) Schedule Page Terbaru

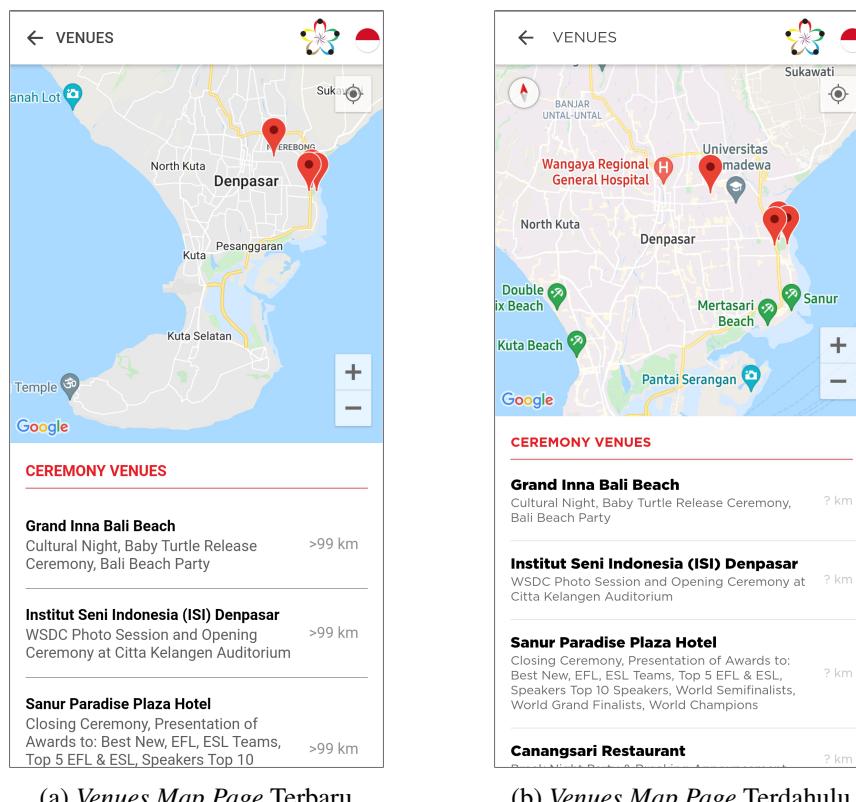
TUE	WED	THU	FRI	SAT	SUN	MON
1	2	3	4	5	6	7
<b>00:00</b> <b>Arrival in Bali</b> <b>23:59</b> International Arrival at I Gusti Ngurah Rai Airport						
<b>08:00</b> <b>Team Registration</b> <b>22:00</b> Lobby at Sanur Paradise Plaza Hotel						
<b>14:00</b> <b>Hotel Check-In</b> <b>23:59</b> Lobby at Sanur Paradise Plaza Hotel						
<b>17:30</b> <b>Dinner</b> <b>21:00</b> Griya Agung Ballroom at Sanur Paradise Plaza Hotel						

(b) Schedule Page Terdahulu

Gambar 5.8: Tangkapan Layar Halaman *Schedule* Aplikasi WSDC 2017 Bali



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



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

## 5.2 Pengujian

### 5.2.1 Pengujian Fungsional

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

Tabel 5.1: Tabel Pengujian Fungsional

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

Tabel 5.2: Lanjutan Tabel Pengujian Fungsional dari Halaman Sebelumnya

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

### **5.2.2 Pengujian Eksperimental**

Pengujian eksperimental dilakukan terhadap pengguna *smartphone* dengan sistem operasi Android. Metode pengujian dilakukan dengan cara menyebarkan aplikasi yang dapat diunduh melalui Google Drive<sup>1</sup>. Responden yang dipilih merupakan sembilan orang mahasiswa dari berbagai jurusan di beberapa universitas di Indonesia dengan rentang usia 21 sampai 23 tahun. Kemudian, pengguna diminta untuk mengunduh dan menjalankan aplikasi tersebut. Pengguna juga diminta untuk mengunduh aplikasi WSDC 2017 Bali terdahulu melalui Google Play Store<sup>2</sup> dan menjalankannya. Setelah itu pengguna diminta untuk membandingkan kedua aplikasi tersebut, dan mengisi beberapa pertanyaan terkait pengalaman menggunakan aplikasi WSDC 2017 Bali melalui Google Form. Berikut ini merupakan pertanyaan dan rangkuman jawaban dari hasil pengujian eksperimental terhadap sembilan responden sebagai berikut:

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

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

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

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

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

Semua responden menjawab tidak ada halaman yang tidak memiliki isi.

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

Semua responden menjawab semua tombol berfungsi dengan normal.

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

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

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

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

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

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

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

14   Sebanyak empat orang responden menjawab bahwa tidak ada perubahan negatif pada aplikasi WSDC  
15   2017 Bali terbaru. Seorang responden menjawab *font* tulisan lebih kaku, dan halaman *draw* kualitasnya  
16   terlihat lebih rendah dibandingkan aplikasi terdahulu. Lalu seorang responden menjawab tampilan  
17   pada aplikasi yang terbaru dari menu yang ada di masing-masing *Venues* sedikit aneh, karena nama  
18   tempat dan alamatnya saling berdekatan tanpa ada jarak spasi. Dan seorang responden menjawab  
19   pemakaian aplikasi terbaru lebih boros.

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

22   Seorang responden menjawab netral dengan skala 3, enam orang responden menjawab dengan skala  
23   4, dan dua orang responden menjawab aplikasi WSDC 2017 Bali lebih baik dibandingkan aplikasi  
24   terdahulu dengan skala 5.

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

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

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

30   Terdapat beberapa kritik dan saran dari responden sebagai berikut:

- 31   (a) Dapat memilih font yang lebih menarik dan nyaman untuk dibaca.
- 32   (b) Pada halaman *Result* diharapkan agar memiliki tampilan lebih menarik lagi.
- 33   (c) Ukuran aplikasi bisa dikecilkan.
- 34   (d) Pada halaman *venues* dimana tulisan headline dari list map dijauhkan sedikit dari tulisan body  
35   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 sebagai  
5 berikut:
- 6 1. Telah berhasil melakukan pembaruan aplikasi WSDC 2017 Bali dengan Ionic Framework versi 6 yang  
7 sebelumnya menggunakan Ionic Framework versi 3.
  - 8 2. Aplikasi WSDC 2017 Bali telah dapat dijalankan pada perangkat dengan sistem operasi Android.

#### 9 6.2 Saran

- 10 Dari hasil penelitian dan pengujian termasuk dengan pengujian terhadap responden, berikut ini merupakan  
11 beberapa saran untuk pengembangan lebih lanjut:
- 12 1. Dapat memilih font yang lebih menarik dan nyaman untuk dibaca.
  - 13 2. Dapat memperbaiki halaman *Result* agar memiliki tampilan lebih menarik.
  - 14 3. Dapat mengecilkan ukuran aplikasi.
  - 15 4. Pada halaman *venues* dimana tulisan headline dari list map dijauhkan sedikit dari tulisan body karena  
16 terlalu dekat.
- 17 Namun menurut pengamatan penulis terhadap kritik dan saran pada poin 3, bahwa ukuran aplikasi WSDC  
18 2017 Bali terbaru sudah cukup kecil, yaitu 25MB. Sedangkan untuk poin 1 dan 2 bersifat subjektif, sehingga  
19 tidak akan diimplementasikan oleh penulis.



## DAFTAR REFERENSI

- [1] Emmit A. Scott, J. (2015) *SPA Design and Architecture: Understanding single-page web applications*, 1st edition. Manning Publications, New York, USA.
- [2] Wargo, J. M. (2014) *Apache Cordova API Cookbook*, 1st edition. Pearson Education, Inc., New Jersey, USA.
- [3] World Schools Debate Championship (2021) WSDC. <https://wsdcdebate.org/history>. [Online; diakses 8-Juli-2021].
- [4] Waranashiwar, J. dan Ukey, M. (2018) Ionic framework with angular for hybrid app development. *International Journal of New Technology and Research*, **4**, 01–02.
- [5] Yusuf, S. (2016) *Ionic Framework By Example*, 1st edition. Pact Publishing Ltd., Birmingham, UK.
- [6] Griffith, C. (2017) *Mobile App Development with Ionic : Cross-Platform Apps with Ionic, Angular and Cordova*, 1st edition. O'Reilly Media, Inc., California, USA.
- [7] Grønli, T.-M., Biørn-Hansen, A., dan Majchrzak, T. A. (2019) Median trajectories using well-visited regions and shortest paths software development for mobile computing the internet of things and wearable devices: Inspecting the past to understand the future. *Proceedings of the 52nd Hawaii International Conference on System Sciences*, Grand Wailea, Hawaii, 8–11 January, pp. 7451–7460. University of Hawaii, Manoa.
- [8] 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.
- [9] Moiseev, A. dan Fain, Y. (2018) *Angular Development with TypeScript*, 2nd edition. Manning Publications, New York, USA.
- [10] Prusty, N. (2015) *Learning ECMAScript 6*, 1st edition. Pact Publishing Ltd., Birmingham, UK.
- [11] Kunz, G. (2018) *Mastering Angular Components: Build component-based user interfaces using Angular*, 2nd edition. Pact Publishing Ltd., Birmingham, UK.
- [12] Savkin, V. (2017) *Angular Router*, 1st edition. Pact Publishing Ltd., Birmingham, UK.
- [13] 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.
- [14] 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 toastController
16     : 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",
56   "Saturday"];
57   var monthNames = ["January", "February", "March", "April", "May", "June", "July",
58   "August", "September", "October", "November", "December"];
59   var dayOfWeek = date.getDay();
60   var day = date.getDate();
61   var monthIndex = date.getMonth();
62   var time=sqlDatetime.substring(16,4)
63   return time.substring(7) + ' | ' + dayNames[dayOfWeek] + ', ' + day + ' ' +
monthNames[monthIndex];
}
}

```

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 =
index">
16       <ion-label class="ion-text-wrap">
17         <h3>{{ formatDatetime(announcement.localtime) }}</h3>
18         <p>{{ announcement.message }}</p>
19       </ion-label>
20     </ion-item>
21   </ion-list>
22 </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 implements OnInit {
12   @ViewChild('drawIFrame') drawIFrame: ElementRef;
13   constructor(private storage: Storage, public loadingController: LoadingController) {
14     }
15   ngOnInit() {
16     this.storage.get('wsdcDataStorage').then((data) => {
17       this.drawIFrame.nativeElement.contentWindow.location.assign(data.draws);
18     });
19     this.presentLoading();
20   }
21
22   async presentLoading() {
23     const loading = await this.loadingController.create({
24       message: 'Please wait...',
25       backdropDismiss: true // If true, the loading indicator will be dismissed when
26       the backdrop is clicked.
27     );
28     await loading.present();
29     setTimeout(() => {
30       loading.dismiss();
31     }, 1000);
32   }
33
34   onDrawIframeLoad() {
35     let doc = this.drawIFrame.nativeElement.contentWindow.document;
36     let elements = [
37       doc.getElementById('header'),
38       doc.getElementById('page_header'),
39       doc.getElementById('footer')
40     ];
41     elements.forEach(function (element) {
42       if (element) {
43         element.style.display = 'none';
44       }
45     })
46     this.drawIFrame.nativeElement.style.display = 'block';
47   }
48 }s

```

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
19   ,public toastController: ToastController) { }
20
21   ionViewDidEnter(){
22     SplashScreen.hide()
23   }
24
25   ngOnInit(){
26     this.storage.get('wsdcDataStorage').then((data) => {
27
28       if(data == null){
29         //Default from asset
30         this.http.get('../assets/json/wsdc_data.json').subscribe((data: any) => {
31           this.wsdcData = data;
32           this.storage.set('wsdcDataStorage',data);
33         },
34         error => {
35           this.showToast('Failed to refresh information from local storage');
36         });
37       }else{
38         this.wsdcData = data;
39       }
40
41       // Refresh data
42       setTimeout(() => {
43         this.http.get('http://wsdc.dnartworks.com/wsdc_data.json')
44         .subscribe((data: any) => {
45           this.storage.set('wsdcDataStorage', data);
46           this.wsdcData = data;
47         },
48         error => {
49           this.showToast('Failed to refresh information');
50         });
51       }, 1000);
52     })
53   }
54
55   doRefresh(refresher) {
56     this.http.get('https://wsdc.dnartworks.com/wsdc_data.json').subscribe((data: any)
57     => {
58       this.storage.clear();
59       this.storage.set('wsdcDataStorage',data);
60       this.wsdcData = data;
61     })
62   }
63
64   showToast(message) {
65     this.toastController.show(message, '3000');
66   }
67
68   ionViewWillLeave() {
69     this.wsdcData = null;
70   }
71
72   ionViewDidEnter() {
73     this.wsdcData = this.storage.get('wsdcDataStorage');
74   }
75
76   ionViewWillEnter() {
77     this.wsdcData = this.storage.get('wsdcDataStorage');
78   }
79
80   ionViewDidLeave() {
81     this.wsdcData = null;
82   }
83
84   ionViewWillLeave() {
85     this.wsdcData = null;
86   }
87
88   ionViewDidEnter() {
89     this.wsdcData = this.storage.get('wsdcDataStorage');
90   }
91
92   ionViewWillEnter() {
93     this.wsdcData = this.storage.get('wsdcDataStorage');
94   }
95
96   ionViewDidLeave() {
97     this.wsdcData = null;
98   }
99
100   ionViewWillLeave() {
101     this.wsdcData = null;
102   }
103
104   ionViewDidEnter() {
105     this.wsdcData = this.storage.get('wsdcDataStorage');
106   }
107
108   ionViewWillEnter() {
109     this.wsdcData = this.storage.get('wsdcDataStorage');
110   }
111
112   ionViewDidLeave() {
113     this.wsdcData = null;
114   }
115
116   ionViewWillLeave() {
117     this.wsdcData = null;
118   }
119
120   ionViewDidEnter() {
121     this.wsdcData = this.storage.get('wsdcDataStorage');
122   }
123
124   ionViewWillEnter() {
125     this.wsdcData = this.storage.get('wsdcDataStorage');
126   }
127
128   ionViewDidLeave() {
129     this.wsdcData = null;
130   }
131
132   ionViewWillLeave() {
133     this.wsdcData = null;
134   }
135
136   ionViewDidEnter() {
137     this.wsdcData = this.storage.get('wsdcDataStorage');
138   }
139
140   ionViewWillEnter() {
141     this.wsdcData = this.storage.get('wsdcDataStorage');
142   }
143
144   ionViewDidLeave() {
145     this.wsdcData = null;
146   }
147
148   ionViewWillLeave() {
149     this.wsdcData = null;
150   }
151
152   ionViewDidEnter() {
153     this.wsdcData = this.storage.get('wsdcDataStorage');
154   }
155
156   ionViewWillEnter() {
157     this.wsdcData = this.storage.get('wsdcDataStorage');
158   }
159
160   ionViewDidLeave() {
161     this.wsdcData = null;
162   }
163
164   ionViewWillLeave() {
165     this.wsdcData = null;
166   }
167
168   ionViewDidEnter() {
169     this.wsdcData = this.storage.get('wsdcDataStorage');
170   }
171
172   ionViewWillEnter() {
173     this.wsdcData = this.storage.get('wsdcDataStorage');
174   }
175
176   ionViewDidLeave() {
177     this.wsdcData = null;
178   }
179
180   ionViewWillLeave() {
181     this.wsdcData = null;
182   }
183
184   ionViewDidEnter() {
185     this.wsdcData = this.storage.get('wsdcDataStorage');
186   }
187
188   ionViewWillEnter() {
189     this.wsdcData = this.storage.get('wsdcDataStorage');
190   }
191
192   ionViewDidLeave() {
193     this.wsdcData = null;
194   }
195
196   ionViewWillLeave() {
197     this.wsdcData = null;
198   }
199
200   ionViewDidEnter() {
201     this.wsdcData = this.storage.get('wsdcDataStorage');
202   }
203
204   ionViewWillEnter() {
205     this.wsdcData = this.storage.get('wsdcDataStorage');
206   }
207
208   ionViewDidLeave() {
209     this.wsdcData = null;
210   }
211
212   ionViewWillLeave() {
213     this.wsdcData = null;
214   }
215
216   ionViewDidEnter() {
217     this.wsdcData = this.storage.get('wsdcDataStorage');
218   }
219
220   ionViewWillEnter() {
221     this.wsdcData = this.storage.get('wsdcDataStorage');
222   }
223
224   ionViewDidLeave() {
225     this.wsdcData = null;
226   }
227
228   ionViewWillLeave() {
229     this.wsdcData = null;
230   }
231
232   ionViewDidEnter() {
233     this.wsdcData = this.storage.get('wsdcDataStorage');
234   }
235
236   ionViewWillEnter() {
237     this.wsdcData = this.storage.get('wsdcDataStorage');
238   }
239
240   ionViewDidLeave() {
241     this.wsdcData = null;
242   }
243
244   ionViewWillLeave() {
245     this.wsdcData = null;
246   }
247
248   ionViewDidEnter() {
249     this.wsdcData = this.storage.get('wsdcDataStorage');
250   }
251
252   ionViewWillEnter() {
253     this.wsdcData = this.storage.get('wsdcDataStorage');
254   }
255
256   ionViewDidLeave() {
257     this.wsdcData = null;
258   }
259
260   ionViewWillLeave() {
261     this.wsdcData = null;
262   }
263
264   ionViewDidEnter() {
265     this.wsdcData = this.storage.get('wsdcDataStorage');
266   }
267
268   ionViewWillEnter() {
269     this.wsdcData = this.storage.get('wsdcDataStorage');
270   }
271
272   ionViewDidLeave() {
273     this.wsdcData = null;
274   }
275
276   ionViewWillLeave() {
277     this.wsdcData = null;
278   }
279
280   ionViewDidEnter() {
281     this.wsdcData = this.storage.get('wsdcDataStorage');
282   }
283
284   ionViewWillEnter() {
285     this.wsdcData = this.storage.get('wsdcDataStorage');
286   }
287
288   ionViewDidLeave() {
289     this.wsdcData = null;
290   }
291
292   ionViewWillLeave() {
293     this.wsdcData = null;
294   }
295
296   ionViewDidEnter() {
297     this.wsdcData = this.storage.get('wsdcDataStorage');
298   }
299
300   ionViewWillEnter() {
301     this.wsdcData = this.storage.get('wsdcDataStorage');
302   }
303
304   ionViewDidLeave() {
305     this.wsdcData = null;
306   }
307
308   ionViewWillLeave() {
309     this.wsdcData = null;
310   }
311
312   ionViewDidEnter() {
313     this.wsdcData = this.storage.get('wsdcDataStorage');
314   }
315
316   ionViewWillEnter() {
317     this.wsdcData = this.storage.get('wsdcDataStorage');
318   }
319
320   ionViewDidLeave() {
321     this.wsdcData = null;
322   }
323
324   ionViewWillLeave() {
325     this.wsdcData = null;
326   }
327
328   ionViewDidEnter() {
329     this.wsdcData = this.storage.get('wsdcDataStorage');
330   }
331
332   ionViewWillEnter() {
333     this.wsdcData = this.storage.get('wsdcDataStorage');
334   }
335
336   ionViewDidLeave() {
337     this.wsdcData = null;
338   }
339
340   ionViewWillLeave() {
341     this.wsdcData = null;
342   }
343
344   ionViewDidEnter() {
345     this.wsdcData = this.storage.get('wsdcDataStorage');
346   }
347
348   ionViewWillEnter() {
349     this.wsdcData = this.storage.get('wsdcDataStorage');
350   }
351
352   ionViewDidLeave() {
353     this.wsdcData = null;
354   }
355
356   ionViewWillLeave() {
357     this.wsdcData = null;
358   }
359
360   ionViewDidEnter() {
361     this.wsdcData = this.storage.get('wsdcDataStorage');
362   }
363
364   ionViewWillEnter() {
365     this.wsdcData = this.storage.get('wsdcDataStorage');
366   }
367
368   ionViewDidLeave() {
369     this.wsdcData = null;
370   }
371
372   ionViewWillLeave() {
373     this.wsdcData = null;
374   }
375
376   ionViewDidEnter() {
377     this.wsdcData = this.storage.get('wsdcDataStorage');
378   }
379
380   ionViewWillEnter() {
381     this.wsdcData = this.storage.get('wsdcDataStorage');
382   }
383
384   ionViewDidLeave() {
385     this.wsdcData = null;
386   }
387
388   ionViewWillLeave() {
389     this.wsdcData = null;
390   }
391
392   ionViewDidEnter() {
393     this.wsdcData = this.storage.get('wsdcDataStorage');
394   }
395
396   ionViewWillEnter() {
397     this.wsdcData = this.storage.get('wsdcDataStorage');
398   }
399
400   ionViewDidLeave() {
401     this.wsdcData = null;
402   }
403
404   ionViewWillLeave() {
405     this.wsdcData = null;
406   }
407
408   ionViewDidEnter() {
409     this.wsdcData = this.storage.get('wsdcDataStorage');
410   }
411
412   ionViewWillEnter() {
413     this.wsdcData = this.storage.get('wsdcDataStorage');
414   }
415
416   ionViewDidLeave() {
417     this.wsdcData = null;
418   }
419
420   ionViewWillLeave() {
421     this.wsdcData = null;
422   }
423
424   ionViewDidEnter() {
425     this.wsdcData = this.storage.get('wsdcDataStorage');
426   }
427
428   ionViewWillEnter() {
429     this.wsdcData = this.storage.get('wsdcDataStorage');
430   }
431
432   ionViewDidLeave() {
433     this.wsdcData = null;
434   }
435
436   ionViewWillLeave() {
437     this.wsdcData = null;
438   }
439
440   ionViewDidEnter() {
441     this.wsdcData = this.storage.get('wsdcDataStorage');
442   }
443
444   ionViewWillEnter() {
445     this.wsdcData = this.storage.get('wsdcDataStorage');
446   }
447
448   ionViewDidLeave() {
449     this.wsdcData = null;
450   }
451
452   ionViewWillLeave() {
453     this.wsdcData = null;
454   }
455
456   ionViewDidEnter() {
457     this.wsdcData = this.storage.get('wsdcDataStorage');
458   }
459
460   ionViewWillEnter() {
461     this.wsdcData = this.storage.get('wsdcDataStorage');
462   }
463
464   ionViewDidLeave() {
465     this.wsdcData = null;
466   }
467
468   ionViewWillLeave() {
469     this.wsdcData = null;
470   }
471
472   ionViewDidEnter() {
473     this.wsdcData = this.storage.get('wsdcDataStorage');
474   }
475
476   ionViewWillEnter() {
477     this.wsdcData = this.storage.get('wsdcDataStorage');
478   }
479
480   ionViewDidLeave() {
481     this.wsdcData = null;
482   }
483
484   ionViewWillLeave() {
485     this.wsdcData = null;
486   }
487
488   ionViewDidEnter() {
489     this.wsdcData = this.storage.get('wsdcDataStorage');
490   }
491
492   ionViewWillEnter() {
493     this.wsdcData = this.storage.get('wsdcDataStorage');
494   }
495
496   ionViewDidLeave() {
497     this.wsdcData = null;
498   }
499
500   ionViewWillLeave() {
501     this.wsdcData = null;
502   }
503
504   ionViewDidEnter() {
505     this.wsdcData = this.storage.get('wsdcDataStorage');
506   }
507
508   ionViewWillEnter() {
509     this.wsdcData = this.storage.get('wsdcDataStorage');
510   }
511
512   ionViewDidLeave() {
513     this.wsdcData = null;
514   }
515
516   ionViewWillLeave() {
517     this.wsdcData = null;
518   }
519
520   ionViewDidEnter() {
521     this.wsdcData = this.storage.get('wsdcDataStorage');
522   }
523
524   ionViewWillEnter() {
525     this.wsdcData = this.storage.get('wsdcDataStorage');
526   }
527
528   ionViewDidLeave() {
529     this.wsdcData = null;
530   }
531
532   ionViewWillLeave() {
533     this.wsdcData = null;
534   }
535
536   ionViewDidEnter() {
537     this.wsdcData = this.storage.get('wsdcDataStorage');
538   }
539
540   ionViewWillEnter() {
541     this.wsdcData = this.storage.get('wsdcDataStorage');
542   }
543
544   ionViewDidLeave() {
545     this.wsdcData = null;
546   }
547
548   ionViewWillLeave() {
549     this.wsdcData = null;
550   }
551
552   ionViewDidEnter() {
553     this.wsdcData = this.storage.get('wsdcDataStorage');
554   }
555
556   ionViewWillEnter() {
557     this.wsdcData = this.storage.get('wsdcDataStorage');
558   }
559
560   ionViewDidLeave() {
561     this.wsdcData = null;
562   }
563
564   ionViewWillLeave() {
565     this.wsdcData = null;
566   }
567
568   ionViewDidEnter() {
569     this.wsdcData = this.storage.get('wsdcDataStorage');
570   }
571
572   ionViewWillEnter() {
573     this.wsdcData = this.storage.get('wsdcDataStorage');
574   }
575
576   ionViewDidLeave() {
577     this.wsdcData = null;
578   }
579
580   ionViewWillLeave() {
581     this.wsdcData = null;
582   }
583
584   ionViewDidEnter() {
585     this.wsdcData = this.storage.get('wsdcDataStorage');
586   }
587
588   ionViewWillEnter() {
589     this.wsdcData = this.storage.get('wsdcDataStorage');
590   }
591
592   ionViewDidLeave() {
593     this.wsdcData = null;
594   }
595
596   ionViewWillLeave() {
597     this.wsdcData = null;
598   }
599
600   ionViewDidEnter() {
601     this.wsdcData = this.storage.get('wsdcDataStorage');
602   }
603
604   ionViewWillEnter() {
605     this.wsdcData = this.storage.get('wsdcDataStorage');
606   }
607
608   ionViewDidLeave() {
609     this.wsdcData = null;
610   }
611
612   ionViewWillLeave() {
613     this.wsdcData = null;
614   }
615
616   ionViewDidEnter() {
617     this.wsdcData = this.storage.get('wsdcDataStorage');
618   }
619
620   ionViewWillEnter() {
621     this.wsdcData = this.storage.get('wsdcDataStorage');
622   }
623
624   ionViewDidLeave() {
625     this.wsdcData = null;
626   }
627
628   ionViewWillLeave() {
629     this.wsdcData = null;
630   }
631
632   ionViewDidEnter() {
633     this.wsdcData = this.storage.get('wsdcDataStorage');
634   }
635
636   ionViewWillEnter() {
637     this.wsdcData = this.storage.get('wsdcDataStorage');
638   }
639
640   ionViewDidLeave() {
641     this.wsdcData = null;
642   }
643
644   ionViewWillLeave() {
645     this.wsdcData = null;
646   }
647
648   ionViewDidEnter() {
649     this.wsdcData = this.storage.get('wsdcDataStorage');
650   }
651
652   ionViewWillEnter() {
653     this.wsdcData = this.storage.get('wsdcDataStorage');
654   }
655
656   ionViewDidLeave() {
657     this.wsdcData = null;
658   }
659
660   ionViewWillLeave() {
661     this.wsdcData = null;
662   }
663
664   ionViewDidEnter() {
665     this.wsdcData = this.storage.get('wsdcDataStorage');
666   }
667
668   ionViewWillEnter() {
669     this.wsdcData = this.storage.get('wsdcDataStorage');
670   }
671
672   ionViewDidLeave() {
673     this.wsdcData = null;
674   }
675
676   ionViewWillLeave() {
677     this.wsdcData = null;
678   }
679
680   ionViewDidEnter() {
681     this.wsdcData = this.storage.get('wsdcDataStorage');
682   }
683
684   ionViewWillEnter() {
685     this.wsdcData = this.storage.get('wsdcDataStorage');
686   }
687
688   ionViewDidLeave() {
689     this.wsdcData = null;
690   }
691
692   ionViewWillLeave() {
693     this.wsdcData = null;
694   }
695
696   ionViewDidEnter() {
697     this.wsdcData = this.storage.get('wsdcDataStorage');
698   }
699
700   ionViewWillEnter() {
701     this.wsdcData = this.storage.get('wsdcDataStorage');
702   }
703
704   ionViewDidLeave() {
705     this.wsdcData = null;
706   }
707
708   ionViewWillLeave() {
709     this.wsdcData = null;
710   }
711
712   ionViewDidEnter() {
713     this.wsdcData = this.storage.get('wsdcDataStorage');
714   }
715
716   ionViewWillEnter() {
717     this.wsdcData = this.storage.get('wsdcDataStorage');
718   }
719
720   ionViewDidLeave() {
721     this.wsdcData = null;
722   }
723
724   ionViewWillLeave() {
725     this.wsdcData = null;
726   }
727
728   ionViewDidEnter() {
729     this.wsdcData = this.storage.get('wsdcDataStorage');
730   }
731
732   ionViewWillEnter() {
733     this.wsdcData = this.storage.get('wsdcDataStorage');
734   }
735
736   ionViewDidLeave() {
737     this.wsdcData = null;
738   }
739
740   ionViewWillLeave() {
741     this.wsdcData = null;
742   }
743
744   ionViewDidEnter() {
745     this.wsdcData = this.storage.get('wsdcDataStorage');
746   }
747
748   ionViewWillEnter() {
749     this.wsdcData = this.storage.get('wsdcDataStorage');
750   }
751
752   ionViewDidLeave() {
753     this.wsdcData = null;
754   }
755
756   ionViewWillLeave() {
757     this.wsdcData = null;
758   }
759
760   ionViewDidEnter() {
761     this.wsdcData = this.storage.get('wsdcDataStorage');
762   }
763
764   ionViewWillEnter() {
765     this.wsdcData = this.storage.get('wsdcDataStorage');
766   }
767
768   ionViewDidLeave() {
769     this.wsdcData = null;
770   }
771
772   ionViewWillLeave() {
773     this.wsdcData = null;
774   }
775
776   ionViewDidEnter() {
777     this.wsdcData = this.storage.get('wsdcDataStorage');
778   }
779
780   ionViewWillEnter() {
781     this.wsdcData = this.storage.get('wsdcDataStorage');
782   }
783
784   ionViewDidLeave() {
785     this.wsdcData = null;
786   }
787
788   ionViewWillLeave() {
789     this.wsdcData = null;
790   }
791
792   ionViewDidEnter() {
793     this.wsdcData = this.storage.get('wsdcDataStorage');
794   }
795
796   ionViewWillEnter() {
797     this.wsdcData = this.storage.get('wsdcDataStorage');
798   }
799
800   ionViewDidLeave() {
801     this.wsdcData = null;
802   }
803
804   ionViewWillLeave() {
805     this.wsdcData = null;
806   }
807
808   ionViewDidEnter() {
809     this.wsdcData = this.storage.get('wsdcDataStorage');
810   }
811
812   ionViewWillEnter() {
813     this.wsdcData = this.storage.get('wsdcDataStorage');
814   }
815
816   ionViewDidLeave() {
817     this.wsdcData = null;
818   }
819
820   ionViewWillLeave() {
821     this.wsdcData = null;
822   }
823
824   ionViewDidEnter() {
825     this.wsdcData = this.storage.get('wsdcDataStorage');
826   }
827
828   ionViewWillEnter() {
829     this.wsdcData = this.storage.get('wsdcDataStorage');
830   }
831
832   ionViewDidLeave() {
833     this.wsdcData = null;
834   }
835
836   ionViewWillLeave() {
837     this.wsdcData = null;
838   }
839
840   ionViewDidEnter() {
841     this.wsdcData = this.storage.get('wsdcDataStorage');
842   }
843
844   ionViewWillEnter() {
845     this.wsdcData = this.storage.get('wsdcDataStorage');
846   }
847
848   ionViewDidLeave() {
849     this.wsdcData = null;
850   }
851
852   ionViewWillLeave() {
853     this.wsdcData = null;
854   }
855
856   ionViewDidEnter() {
857     this.wsdcData = this.storage.get('wsdcDataStorage');
858   }
859
860   ionViewWillEnter() {
861     this.wsdcData = this.storage.get('wsdcDataStorage');
862   }
863
864   ionViewDidLeave() {
865     this.wsdcData = null;
866   }
867
868   ionViewWillLeave() {
869     this.wsdcData = null;
870   }
871
872   ionViewDidEnter() {
873     this.wsdcData = this.storage.get('wsdcDataStorage');
874   }
875
876   ionViewWillEnter() {
877     this.wsdcData = this.storage.get('wsdcDataStorage');
878   }
879
880   ionViewDidLeave() {
881     this.wsdcData = null;
882   }
883
884   ionViewWillLeave() {
885     this.wsdcData = null;
886   }
887
888   ionViewDidEnter() {
889     this.wsdcData = this.storage.get('wsdcDataStorage');
890   }
891
892   ionViewWillEnter() {
893     this.wsdcData = this.storage.get('wsdcDataStorage');
894   }
895
896   ionViewDidLeave() {
897     this.wsdcData = null;
898   }
899
900   ionViewWillLeave() {
901     this.wsdcData = null;
902   }
903
904   ionViewDidEnter() {
905     this.wsdcData = this.storage.get('wsdcDataStorage');
906   }
907
908   ionViewWillEnter() {
909     this.wsdcData = this.storage.get('wsdcDataStorage');
910   }
911
912   ionViewDidLeave() {
913     this.wsdcData = null;
914   }
915
916   ionViewWillLeave() {
917     this.wsdcData = null;
918   }
919
920   ionViewDidEnter() {
921     this.wsdcData = this.storage.get('wsdcDataStorage');
922   }
923
924   ionViewWillEnter() {
925     this.wsdcData = this.storage.get('wsdcDataStorage');
926   }
927
928   ionViewDidLeave() {
929     this.wsdcData = null;
930   }
931
932   ionViewWillLeave() {
933     this.wsdcData = null;
934   }
935
936   ionViewDidEnter() {
937     this.wsdcData = this.storage.get('wsdcDataStorage');
938   }
939
940   ionViewWillEnter() {
941     this.wsdcData = this.storage.get('wsdcDataStorage');
942   }
943
944   ionViewDidLeave() {
945     this.wsdcData = null;
946   }
947
948   ionViewWillLeave() {
949     this.wsdcData = null;
950   }
951
952   ionViewDidEnter() {
953     this.wsdcData = this.storage.get('wsdcDataStorage');
954   }
955
956   ionViewWillEnter() {
957     this.wsdcData = this.storage.get('wsdcDataStorage');
958   }
959
960   ionViewDidLeave() {
961     this.wsdcData = null;
962   }
963
964   ionViewWillLeave() {
965     this.wsdcData = null;
966   }
967
968   ionViewDidEnter() {
969     this.wsdcData = this.storage.get('wsdcDataStorage');
970   }
971
972   ionViewWillEnter() {
973     this.wsdcData = this.storage.get('wsdcDataStorage');
974   }
975
976   ionViewDidLeave() {
977     this.wsdcData = null;
978   }
979
980   ionViewWillLeave() {
981     this.wsdcData = null;
982   }
983
984   ionViewDidEnter() {
985     this.wsdcData = this.storage.get('wsdcDataStorage');
986   }
987
988   ionViewWillEnter() {
989     this.wsdcData = this.storage.get('wsdcDataStorage');
990   }
991
992   ionViewDidLeave() {
993     this.wsdcData = null;
994   }
995
996   ionViewWillLeave() {
997     this.wsdcData = null;
998   }
999
1000   ionViewDidEnter() {
1001     this.wsdcData = this.storage.get('wsdcDataStorage');
1002   }
1003
1004   ionViewWillEnter() {
1005     this.wsdcData = this.storage.get('wsdcDataStorage');
1006   }
1007
1008   ionViewDidLeave() {
1009     this.wsdcData = null;
1010   }
1011
1012   ionViewWillLeave() {
1013     this.wsdcData = null;
1014   }
1015
1016   ionViewDidEnter() {
1017     this.wsdcData = this.storage.get('wsdcDataStorage');
1018   }
1019
1020   ionViewWillEnter() {
1021     this.wsdcData = this.storage.get('wsdcDataStorage');
1022   }
1023
1024   ionViewDidLeave() {
1025     this.wsdcData = null;
1026   }
1027
1028   ionViewWillLeave() {
1029     this.wsdcData = null;
1030   }
1031
1032   ionViewDidEnter() {
1033     this.wsdcData = this.storage.get('wsdcDataStorage');
1034   }
1035
1036   ionViewWillEnter() {
1037     this.wsdcData = this.storage.get('wsdcDataStorage');
1038   }
1039
1040   ionViewDidLeave() {
1041     this.wsdcData = null;
1042   }
1043
1044   ionViewWillLeave() {
1045     this.wsdcData = null;
1046   }
1047
1048   ionViewDidEnter() {
1049     this.wsdcData = this.storage.get('wsdcDataStorage');
1050   }
1051
1052   ionViewWillEnter() {
1053     this.wsdcData = this.storage.get('wsdcDataStorage');
1054   }
1055
1056   ionViewDidLeave() {
1057     this.wsdcData = null;
1058   }
1059
1060   ionViewWillLeave() {
1061     this.wsdcData = null;
1062   }
1063
1064   ionViewDidEnter() {
1065     this.wsdcData = this.storage.get('wsdcDataStorage');
1066   }
1067
1068   ionViewWillEnter() {
1069     this.wsdcData = this.storage.get('wsdcDataStorage');
1070   }
1071
1072   ionViewDidLeave() {
1073     this.wsdcData = null;
1074   }
1075
1076   ionViewWillLeave() {
1077     this.wsdcData = null;
1078   }
1079
1080   ionViewDidEnter() {
1081     this.wsdcData = this.storage.get('wsdcDataStorage');
1082   }
1083
1084   ionViewWillEnter() {
1085     this.wsdcData = this.storage.get('wsdcDataStorage');
1086   }
1087
1088   ionViewDidLeave() {
1089     this.wsdcData = null;
1090   }
1091
1092   ionViewWillLeave() {
1093     this.wsdcData = null;
1094   }
1095
1096   ionViewDidEnter() {
1097     this.wsdcData = this.storage.get('wsdcDataStorage');
1098   }
1099
1100   ionViewWillEnter() {
1101     this.wsdcData = this.storage.get('wsdcDataStorage');
1102   }
1103
1104   ionViewDidLeave() {
1105     this.wsdcData = null;
1106   }
1107
1108   ionViewWillLeave() {
1109     this.wsdcData = null;
1110   }
1111
1112   ionViewDidEnter() {
1113     this.wsdcData = this.storage.get('wsdcDataStorage');
1114   }
1115
1116   ionViewWillEnter() {
1117     this.wsdcData = this.storage.get('wsdcDataStorage');
1118   }
1119
1120   ionViewDidLeave() {
1121     this.wsdcData = null;
1122   }
1123
1124   ionViewWillLeave() {
1125     this.wsdcData = null;
1126   }
1127
1128   ionViewDidEnter() {
1129     this.wsdcData = this.storage.get('wsdcDataStorage');
1130   }
1131
1132   ionViewWillEnter() {
1133     this.wsdcData = this.storage.get('wsdcDataStorage');
1134   }
1135
1136   ionViewDidLeave() {
1137     this.wsdcData = null;
1138   }
1139
1140   ionViewWillLeave() {
1141     this.wsdcData = null;
1142   }
1143
1144   ionViewDidEnter() {
1145     this.wsdcData = this.storage.get('wsdcDataStorage');
1146   }
1147
1148   ionViewWillEnter() {
1149     this.wsdcData = this.storage.get('wsdcDataStorage');
1150   }
1151
1152   ionViewDidLeave() {
1153     this.wsdcData = null;
1154   }
1155
1156   ionViewWillLeave() {
1157     this.wsdcData = null;
1158   }
1159
1160   ionViewDidEnter() {
1161     this.wsdcData = this.storage.get('wsdcDataStorage');
1162   }
1163
1164   ionViewWillEnter() {
1165     this.wsdcData = this.storage.get('wsdcDataStorage');
1166   }
1167
1168   ionViewDidLeave() {
1169     this.wsdcData = null;
1170   }
1171
1172   ionViewWillLeave() {
1173     this.wsdcData = null;
1174   }
1175
1176   ionViewDidEnter() {
1177     this.wsdcData = this.storage.get('wsdcDataStorage');
1178   }
1179
1180   ionViewWillEnter() {
1181     this.wsdcData = this.storage.get('wsdcDataStorage');
1182   }
1183
1184   ionViewDidLeave() {
1185     this.wsdcData = null;
1186   }
1187
1188   ionViewWillLeave() {
1189     this.wsdcData = null;
1190   }
1191
1192   ionViewDidEnter() {
1193     this.wsdcData = this.storage.get('wsdcDataStorage');
1194   }
1195
1196   ionViewWillEnter() {
1197     this.wsdcData = this.storage.get('wsdcDataStorage');
1198   }
1199
1200   ionViewDidLeave() {
1201     this.wsdcData = null;
1202   }
1203
1204   ionViewWillLeave() {
1205     this.wsdcData = null;
1206   }
1207
1208   ionViewDidEnter() {
1209     this.wsdcData = this.storage.get('wsdcDataStorage');
1210   }
1211
1212   ionViewWillEnter() {
1213     this.wsdcData = this.storage.get('wsdcDataStorage');
1214   }
1215
1216   ionViewDidLeave() {
1217     this.wsdcData = null;
1218   }
1219
1220   ionViewWillLeave() {
1221     this.wsdcData = null;
1222   }
1223
1224   ionViewDidEnter() {
1225     this.wsdcData = this.storage.get('wsdcDataStorage');
1226   }
1227
1228   ionViewWillEnter() {
1229     this.wsdcData = this.storage.get('wsdcDataStorage');
1230   }
1231
1232   ionViewDidLeave() {
1233     this.wsdcData = null;
1234   }
1235
1236   ionViewWillLeave() {
1237     this.wsdcData = null;
1238   }
1239
1240   ionViewDidEnter() {
1241     this.wsdcData = this.storage.get('wsdcDataStorage');
1242   }
1243
1244   ionViewWillEnter() {
1245     this.wsdcData = this.storage.get('wsdcDataStorage');
1246   }
1247
1248   ionViewDidLeave() {
1249     this.wsdcData = null;
1250   }
1251
1252   ionViewWillLeave() {
1253     this.wsdcData = null;
1254   }
1255
1256   ionViewDidEnter() {
1257     this.wsdcData = this.storage.get('wsdcDataStorage');
1258   }
1259
1260   ionViewWillEnter() {
1261     this.wsdcData = this.storage.get('wsdcDataStorage');
1262   }
1263
1264   ionViewDidLeave() {
1265     this.wsdcData = null;
1266   }
1267
1268   ionViewWillLeave() {
1269     this.wsdcData = null;
1270   }
1271
1272   ionViewDidEnter() {
1273     this.wsdcData = this.storage.get('wsdcDataStorage');
1274   }
1275
1276   ionViewWillEnter() {
1277     this.wsdcData = this.storage.get('wsdcDataStorage');
1278   }
1279
1280   ionViewDidLeave() {
1281     this.wsdcData = null;
1282   }
1283
1284   ionViewWillLeave() {
1285     this.wsdcData = null;
1286   }
1287
1288   ionViewDidEnter() {
1289     this.wsdcData = this.storage.get('wsdcDataStorage');
1290   }
1291
1292   ionViewWillEnter() {
1293     this.wsdcData = this.storage.get('wsdcDataStorage');
1294
```

```

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",
87     "Saturday"];
88     var dayOfWeek = date.getDay();
89     return dayNames[dayOfWeek] + ', ' + time.substring(7);
90   }
91
92   // ionic 6 : use capacitor in app browser
93   launch(newsUrl: string) {
94     Browser.open({ url: newsUrl });
95   }
96
97   // ionic 6 : use ionic angular router for change page
98   onAnnouncementClick() {
99     this.router.navigate(['announcement']);
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>titlecontent>
23         <h3>{{ formatDatetime(wsdcdData?.announcements[0].localtime) }}</h3>
24         <p>{{ wsdcdData?.announcements[0].message }}</p>
25     </ion-card>content>
26   </ion-col>
27   <ion-col size="3">
28     
29   </ion-col>
30 </ion-row>
31 </ion-grid>
32 </ion-card>
33
34 <ion-list>
35   <ion-list-header>
36     <ion-icon name="book-outline"></ion-icon>
37     <ion-label class="label_newsletters"><h1><b> Newsletters</b></h1></ion-label>
38   </ion-list-header>
39   <ion-item *ngFor="let wsdcNews of wsdcdData?.newsletters;">
40     <div class="newsletters" >
41       
42         <h2 text-wrap>{{wsdcNews.title}}</h2> <br>
43         <ion-button class="ion-padding-end" full block color="danger" (click)="launch(wsdcNews.url)">Read More</ion-button>
44     </div>
45   </ion-item>
46 </ion-list>
47 </ion-content>

```

Kode A.6: home.page.html

## A.4 Komponen Info

```

1 import { Component, ViewEncapsulation } from '@angular/core';
2 import { Storage } from '@ionic/storage';
3
4 @Component({
5   selector: 'app-info',
6   templateUrl: './info.page.html',
7   styleUrls: ['./info.page.scss'],
8   encapsulation: ViewEncapsulation.None,
9 })
10
11 export class InfoPage{
12   wsdcdInfoData: any;
13
14   constructor(private storage: Storage) {
15
16     this.storage.get('wsdcDataStorage').then((data) => {
17       this.wsdcdInfoData = data.info;
18       this.wsdcdInfoData = this.wsdcdInfoData.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
17   ngOnInit() {
18     this.storage.get('wsdcDataStorage').then((data) => {
19       this.resultIFrame.nativeElement.contentWindow.location.assign(data.results);
20     });
21     this.presentLoading();
22   }
23
24   async presentLoading() {
25     const loading = await this.loadingController.create({
26       message: 'Please wait...',
27       backdropDismiss: true
28     });
29
30     await loading.present();
31
32     setTimeout(() => {
33       loading.dismiss();
34     }, 500);
35   }
36
37   onResultIframeLoad() {
38     let doc = this.resultIFrame.nativeElement.contentWindow.document;
39     let elements = [

```

```

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

```

Kode A.9: result.page.ts

```

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

```

Kode A.10: result.page.html

## A.6 Komponen Schedule

```

1 import { Component, ElementRef } from '@angular/core';
2 import { ViewChild } from '@angular/core';
3 import { IonSlides } from '@ionic/angular';
4 import { Storage } from '@ionic/storage';
5
6 @Component({
7   selector: 'app-schedule',
8   templateUrl: './schedule.page.html',
9   styleUrls: ['./schedule.page.scss'],
10 })
11 export class SchedulePage implements OnInit{
12   @ViewChild('scheduleSlider', { static: false }) slider: IonSlides;
13   @ViewChild('segmentContainer', { static: false }) segmentContainer: ElementRef;
14   schedules: any;
15   slideOpts = {
16     initialSlide: 0,
17     speed: 400,
18   };
19   selectedSegmentIdx: any;
20   currentIndex: any;
21
22   constructor(private storage: Storage) {
23
24   }
25
26   ngOnInit() {
27     this.storage.get('wsdcDataStorage').then((val) => {
28       this.schedules = val.schedules;

```

```

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

```

Kode A.11: schedule.page.ts

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>Schedule</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <div id="schedulesContainer">
12     <div id="schedulesSegments" slot="fixed">
13       <ion-segment #segmentContainer scrollable [(ngModel)]="selectedSegmentIdx" (
14         ionChange)="onSegmentChanged($event)">
15         <ion-segment-button *ngFor="let schedule of schedules; let i = index;" [value
16           ]="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)="
onSlideChanged()">
26         <ion-slide *ngFor="let schedule of schedules;">
27           <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 implements OnInit{
11   venuesData: Array<{ id: string, name: string, icon: string, geojson: any, colorIdx: number }>;
12   valVenues: any;
13   constructor(private router: Router,private storage: Storage) {
14
15 }
16
17 ngOnInit() {
18   this.storage.get('wsdcDataStorage').then((val) => {
19     this.valVenues = val.venues;
20     this.venuesData = [];
21     let currColorIdx: number = 1;
22     for (let venue of this.valVenues) {
23       this.venuesData.push({
24         id: venue.id,
25         name: venue.name,
26         icon: venue.icon,
27         geojson: venue.geojson,
28         colorIdx: currColorIdx,
29       });
30       if (currColorIdx > 4) {
31         currColorIdx = 1;
32       } else {
33         currColorIdx++;
34       }
35     }
36   })
37 }
38
39   itemTapped(wsdcVenue) {
```

```

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

Kode A.13: venues.page.ts

```

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

Kode A.14: venues.page.html

## A.8 Komponen Venues Map

```

1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute, Router } from '@angular/router';
3 import { Storage } from '@ionic/storage';
4 import { CapacitorGoogleMaps } from "@capacitor-community/google-maps";
5 import { Geolocation } from '@capacitor/geolocation';
6
7 @Component({
8   selector: 'app-venues-map',
9   templateUrl: './venues-map.page.html',
10  styleUrls: ['./venues-map.page.scss'],
11 })
12 export class VenuesMapPage implements OnInit{
13   venuesPage: any;
14   venuesMapsDetail: any;
15   userCoordinatesLat: any;
16   userCoordinatesLng: any;
17   mapid: any;
18   userDistanceTo: string[];
19   itemCounter: number;
20   items: Array<{id: string, idcolor:number}>;
21   pageTitleColors: Array<string> = ["#ec1c24", "#c49a6c", "#d189bb", "#4d113f"];
22 }
```

```
23 constructor(private activatedRoute: ActivatedRoute, private router: Router,private
24 storage: Storage) {
25
26     //Get data from venues page.
27     this.items = [];
28     this.activatedRoute.queryParams.subscribe(params => {
29         if (this.router.getCurrentNavigation().extras.state) {
30             this.venuesPage = this.router.getCurrentNavigation().extras.state.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     //save user lat & lng location
43     const printCurrentPosition = async () => {
44         const coordinates = await Geolocation.getCurrentPosition();
45         this.userCoordinatesLat = coordinates.coords.latitude;
46         this.userCoordinatesLng = coordinates.coords.longitude;
47     };
48
49     printCurrentPosition();
50     Geolocation.requestPermissions();
51 }
52
53 ngOnInit() {
54     // Select data from storage
55     this.storage.get('wsdcDataStorage').then((data) => {
56         this.venuesMapsDetail = data.venues;
57         this.venuesMapsDetail = this.venuesMapsDetail.filter(d=> d.id==this.items[0].id);
58     });
59 }
60
61 ionViewDidEnter() {
62     // create map and add marker
63     const initializeMap = async () => {
64         this.itemCounter = 0;
65         await CapacitorGoogleMaps.initialize({
66             key: "YOUR_IOS_MAPS_API_KEY",
67             devicePixelRatio: window.devicePixelRatio,
68         });
69         const element = document.getElementById("mapContainer");
70         const boundingRect = element.getBoundingClientRect();
71         try {
72             const result = await CapacitorGoogleMaps.createMap({
73                 boundingRect: {
74                     width: Math.round(boundingRect.width),
75                     height: Math.round(boundingRect.height),
76                     x: Math.round(boundingRect.x),
77                     y: Math.round(boundingRect.y),
78                 },
79                 cameraPosition:{
80                     target:{ //Kuta, Bali -8.722396, 115.17671
81                         latitude:-8.722396,
82                         longitude: 115.17671
83                     },
84                     zoom:11
85                 }
86             });
87         }
88     };
89 }
```

```
83     },
84     preferences:{
85       controls:{
86         isCompassButtonEnabled:true,
87         isMyLocationButtonEnabled:true,
88         isZoomButtonsEnabled:true
89       },
90       appearance:{
91         isMyLocationDotShown:true
92       }
93     },
94   });
95
96   element.style.background = "";
97   element.setAttribute("data-maps-id", result.googleMap.mapId);
98   this.mapid = result.googleMap.mapId;
99   console.log(this.venuesMapsDetail);
100
101  //add marker to each location
102  for(let venue of this.venuesMapsDetail){
103    for(let venuesMarker of venue.geojson.features){
104      this.itemCounter +=1;
105      const koor = venuesMarker.geometry.coordinates;
106      CapacitorGoogleMaps.addMarker({
107        mapId:result.googleMap.mapId,
108        position:{
109          latitude: koor[1],
110          longitude: koor[0],
111        },
112        preferences:{
113          title: venuesMarker.properties.Name
114        },
115      });
116    }
117  }
118 } catch (e) {
119   alert("Map failed to load");
120 }
121
122 // Insert distance to each location
123 this.userDistanceTo = new Array(this.itemCounter);
124 let counter = 0;
125 for(let venue of this.venuesMapsDetail){
126   for(let venuesMarker of venue.geojson.features){
127     const koor = venuesMarker.geometry.coordinates;
128     this.userDistanceTo[counter] = this.computeDistance(this.userCoordinatesLat,
129     koor[1],this.userCoordinatesLng,koor[0]);
130     counter+=1;
131   }
132 }
133
134 (function () {
135   initializeMap();
136 })();
137 }
138
139 backToVenue() {
140   this.router.navigate(['venues']);
141 }
142
143 featTapped(venuesCoordinates){
144   const coordinates = venuesCoordinates;
```

```

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

```

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

```

1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start" (click)="backToVenue()">
4       <ion-icon name="arrow-back-outline"></ion-icon>
5     </ion-buttons>
6     <ion-title>Venues</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <div id="container">
11   <div id="mapContainer"></div>
12 </div>
13
14 <!-- //deprecated scroll in ionic 5 -->
15 <ion-content scrollX="true">
16   <ion-label>
17     <h3 #pagetitle id="pagetitle">{{venuesPage.name}}</h3>
18   </ion-label>
19
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.  
24        features; let j = index">  
25          <div class="venueDesc" (click)="featTapped(properties.geometry.coordinates)">  
26            <h2>{{properties.properties.Name}}</h2>  
27            <p>{{properties.properties.Description}}</p>  
28          </div>  
29          <div class="venuesDist" #distance>  
30            <p>{{userDistanceTo[j]}}</p>  
31          </div>  
32        </div>  
33      </ion-item>  
34    </ion-list>  
35 </ion-content>
```

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