Pemanfaatan Directive pada Framework Angular untuk Pengembangan Website Penerimaan Mahasiswa Baru

Wahyu Kartika Candra Kirana
Program Studi Informatika
Universitas Islam Indonesia
Yogyakarta, Indonesia
wahyu.kirana@students.uii.ac.id

Zainudin Zukhri
Program Studi Informatika
Universitas Islam Indonesia
Yogyakarta, Indonesia
zainudin@uii.ac.id

***Abstrak*— Terdapat ruang peningkatan efektivitas dan efisiensi pada proses penerimaan mahasiswa baru di Universitas Islam Indonesia dengan dilakukannya digitalisasi proses pendaftaran. Oleh karena itu, dikembangkan sebuah aplikasi berbasis *web* untuk menjembatani interaksi calon mahasiswa dengan proses pendaftaran. Aplikasi ini merupakan upaya modernisasi aplikasi UIIAdmisi versi lama yang rilis pada tahun 2016 dengan perancangan ulang seluruh fitur yang ditawarkan untuk meningkatkan pengalaman pengguna. Selebihnya, aplikasi UIIAdmisi yang baru memanfaatkan berbagai teknologi pengembangan yang lebih modern. Digunakan *framework* Angular untuk memaksimalkan efisiensi proses pengembangan aplikasi tersebut. Salah satu fitur utama Angular yang membantu optimasi pembangunan aplikasi yaitu *directive*. Pendekatan *scrum* pada metode *agile* digunakan untuk mengorganisir pengerjaan pembangunan aplikasi. Makalah ini akan memberikan gambaran dan pemahaman akan proses yang dilalui pengembang dalam implementasi berbagai fitur pada aplikasi berbasis *web* dengan memanfaatkan *directive* pada *framework* Angular. Setelah melalui masa pengembangan aplikasi UIIAdmisi ditemukan bahwa penggunaan berbagai fitur pada *framework* Angular khususnya *directive* menghadirkan berbagai kelebihan dibandingkan dengan *native development*. Beberapa kelebihan yang dirasakan berupa fleksibilitas pengembangan serta terjaminnya kerapian dan kecepatan penulisan kode.**

***Kata Kunci*— Aplikasi berbasis *web, framework,* Angular, *directive, scrum, agile*.**

I. PENDAHULUAN

Efektivitas penyebaran informasi melalui internet memunculkan tren digitalisasi seluruh aspek kehidupan. Dampak tren digitalisasi terhadap model bisnis terbukti telah mengalami perkembangan yang substansial [1]. Perguruan tinggi juga mengikuti tren ini dengan semakin banyaknya penggunaan *website* sebagai media penyebaran informasi dan penyediaan layanan. Salah satu bagian terpenting dalam sebuah perguruan tinggi yaitu penerimaan mahasiswa baru atau yang biasa disebut proses admisi. Proses penerimaan mahasiswa baru yang efektif dapat meningkatkan reputasi sebuah perguruan tinggi, memperluas jangkauan calon mahasiswa, dan meningkatkan jumlah mahasiswa baru yang diterima setiap tahunnya.

Dalam rangka meningkatkan efektivitas dan efisiensi proses penerimaan mahasiswa baru, Universitas Islam Indonesia (UII) melalui Badan Sistem Informasi (BSI) mengembangkan aplikasi berbasis web penerimaan mahasiswa baru pada tahun 2016 yang dinamakan "UIIAdmisi". *Website* ini mencakup seluruh proses bisnis penerimaan mahasiswa baru, mulai dari pengisian data diri pendaftar, pembelian formulir pendaftaran, sampai pengumuman status pendaftaran. UIIAdmisi meningkatkan aksesibilitas layanan penerimaan mahasiswa baru UII dengan memungkinkan interaksi pengguna secara daring. Walaupun demikian, tidak dapat dipungkiri bahwa teknologi pengembangan web terlah mengalami perkembangan yang signifikan sejak tahun 2016 [2]. Seiring dengan perkembangan tersebut, standar pengalaman pengguna telah meningkat dan aplikasi UIIAdmisi 2016 yang tadinya dirasa cukup, sekarang dinilai memerlukan perombakan dan pembaharuan. Masalah utama aplikasi UIIAdmisi 2016 yaitu struktur pemrograman aplikasi web tersebut masih belum memfasilitasi kemudahan integrasi antar komponen. Hal ini menghadirkan kesulitan dalam implementasi fitur baru. Dengan kata lain, skalabilitas aplikasi UIIAdmisi 2016 masih terbilang rendah. Selain itu, UIIAdmisi 2016 masih belum dapat memberikan pengalaman pengguna yang optimal, dengan tampilan dan alur aplikasi yang dapat dibilang kuno. Pengalaman pengguna memiliki keterkaitan yang kuat dengan aksesibilitas aplikasi web yang dirasakan pengguna [3].

Untuk menangani masalah skalabilitas dan pengalaman pengguna, dilakukan pengembangan aplikasi baru dengan nama yang sama, namun kali ini pengembangan dilakukan dengan mengadopsi berbagai teknologi. Dari segi *front-end*, UIIAdmisi memanfaatkan teknologi Angular. Angular merupakan *framework* pembangunan aplikasi berbasis web yang menyediakan berbagai kelebihan tersendiri. *Framework* ini membantu pengembang untuk menggunakan kembali berbagai komponen aplikasi dan mengembangkan fungsionalitasnya [4]. Untuk memastikan pengalaman pengguna yang lebih baik, Angular menyediakan fitur yang dinamakan *directive*. Dengan menggunakan *directive*, pengembang dapat melakukan manipulasi struktur *Document Object Model* (DOM) sebuah aplikasi. Sederhananya, *directive* memberikan kontrol kepada pengembang dalam penampilan informasi ke pengguna. Contohnya, pengembang dapat menyembunyikan elemen tertentu berdasarkan *input* dari pengguna.

Mengingat topik penelitian yang bersifat teknis, karya ilmiah ini akan menekankan pembahasan pada tahap

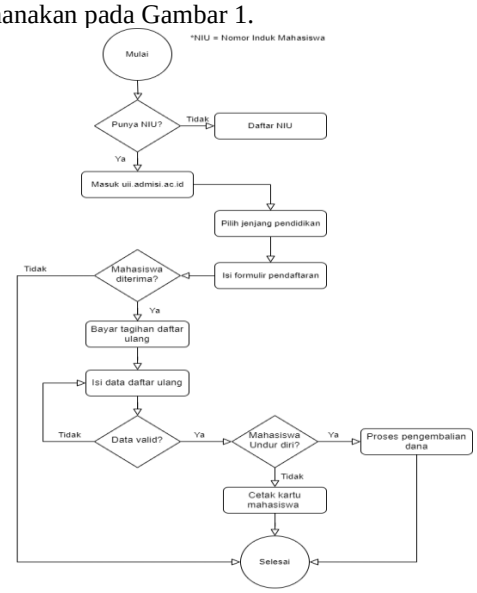
pengembangan. Karya ilmiah ini akan membahas mengenai pengembangan *website* UIIAdmisi dengan fokus pada pemanfaatan *framework* Angular, khususnya penggunaan *directive*. Makalah ini akan menguraikan berbagai jenis *directive* pada *framework* Angular dan proses penerapannya untuk melakukan manipulasi struktur DOM dalam pengembangan sebuah aplikasi berbasis web. Makalah ini juga akan mengemukakan beberapa kasus penggunaan *directive* pada pengembangan UIIAdmisi. Harapannya, makalah ini dapat memberikan pemahaman dan gambaran kepada pembaca mengenai kelebihan penggunaan *directive* dalam proses pengembangan aplikasi berbasis web dengan *framework* Angular, dilengkapi dengan proses implementasi pada sebuah proyek nyata.

II. Landasan Teori

A. UIIAdmisi

UIIAdmisi merupakan aplikasi berbasis web kembangan Badan Sistem Informasi UII. Aplikasi ini ditujukan untuk mempermudah proses pelaksanaan penerimaan mahasiswa baru di Universitas Islam Indonesia dengan proses digitalisasi proses bisnis yang mengedepankan kualitas pengalaman pengguna yang diharapkan akan menghadirkan berbagai keuntungan bagi institusi [5]. UIIAdmisi secara resmi diluncurkan seiring dengan pembukaan penerimaan mahasiswa baru yang jatuh pada bulan Desember 2022. UIIAdmisi terbagi menjadi dua versi, yaitu versi calon mahasiswa yang diluncurkan pada alamat uii.admisi.ac.id, dan versi petugas yang diwujudkan dalam bentuk sistem informasi yang dapat diakses menggunakan akun petugas. Kedua versi aplikasi berkomunikasi satu sama lain, di mana UIIAdmisi calon mahasiswa memungkinkan dilakukannya pendaftaran melalui berbagai pola seleksi, sedangkan UIIAdmisi petugas ditujukan kepada para petugas dan panitia admisi untuk menindaklanjuti pendaftaran dari calon mahasiswa.

Pada masa penerimaan mahasiswa baru tahun ajaran 2022/2023, pola seleksi pendaftaran yang ditawarkan UII terbagi menjadi 3 grup, *Computer Based Test* (CBT), Seleksi Berbasis Rapor (SIBER), dan Penelusuran Mahasiswa Berprestasi (PSB). Adapun proses bisnis UIIAdmisi dapat dilihat pada diagram alur pengguna yang telah disederhanakan pada Gambar 1.



Gambar 1. Diagram alur pengguna UIIAdmisi

B. Front-end Web Development

Istilah *front-end* merupakan salah satu bagian dari proses pengembangan aplikasi berbasis web. Istilah tersebut mengacu pada proses pengembangan perangkat lunak yang akan digunakan oleh pengguna akhir secara langsung. Tanggung jawab utama pengembang *front-end* berada pada implementasi antar muka aplikasi. Sederhananya, seluruh elemen yang dapat dilihat oleh pengguna merupakan tanggung jawab pengembang *front-end*. Pada umumnya, teknis pengembangan *front-end* mencakup 3 komponen utama, yaitu *HyperText Markup Language* (HTML), *Cascading Style Sheet* (CSS), dan JavaScript/TypeScript [6]. HTML merepresentasikan susunan elemen pada halaman aplikasi, CSS untuk menambahkan *styling* ke elemen tersebut, dan JavaScript/TypeScript untuk memberikan alur ke aplikasi melalui logika pemrograman . Utamanya, tujuan akhir pengembang *front-end* yakni menciptakan halaman web yang jelas, mudah, dan cepat melalui antarmuka yang dapat membuat pengguna mengerti dan peduli akan informasi yang disediakan. Oleh karena itu, seorang pengembang *front-end* perlu memiliki kemampuan dan kepekaan desain yang cukup untuk dapat mewujudkan tampilan web yang tidak berantakan [6].

C. Document Object Model

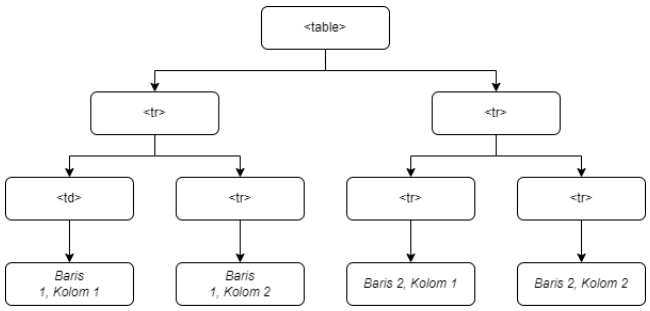
Document Object Model (DOM) merepresentasikan struktur sebuah halaman web yang disusun dari dokumen HTML [7]. DOM berperan sebagai jalur komunikasi antara dokumen tersebut dan kode logika pemrograman untuk dilaksanakannya proses manipulasi elemen-elemen dalam dokumen tersebut melalui kode. Hal ini memungkinkan interaksi dan pengubahan konten pada halaman web secara dinamis .

Pada umumnya, DOM divisualisasikan sebagai sebuah pohon yang membagi dokumen secara hierarkis, di mana objek dengan kepemilikan terbanyak diletakkan di bagian atas dan sebaliknya. Contohnya, untuk merepresentasikan elemen tabel pada dokumen HTML yang tertera pada Gambar 2.

```
<table>
    <tr>
        <td>Baris 1, Kolom 1</td>
        <td>Baris 1, Kolom 2</td>
    </tr>
    <tr>
        <td>Baris 2, Kolom 1</td>
        <td>Baris 2, Kolom 2</td>
    </tr>
</table>
```

Gambar 2. Elemen tabel pada contoh dokumen HTML

Dokumen HTML di atas menunjukkan sebuah tabel sederhana berisi dua baris dan dua kolom. Representasi DOM dari tabel berikut dapat dilihat pada visualisasi diagram pohon pada Gambar 3.



Gambar 3. Diagram pohon representasi DOM

Melalui DOM, pengembang dapat mengakses, mengubah, dan menghapus elemen-elemen dalam halaman web [8]. Mereka dapat mencari elemen berdasarkan *tag*, atribut, atau posisi dalam pohon DOM. Pengembang juga dapat mengubah konten teks, menambahkan atau menghapus elemen, serta mengubah atribut dan gaya CSS dari elemen. DOM juga memungkinkan pengembang untuk membuat animasi dan efek visual pada halaman web dengan mengubah atribut dan gaya CSS secara dinamis. Misalnya, pengembang dapat mengubah warna, posisi, atau ukuran elemen secara langsung melalui manipulasi DOM.

D. Framework Angular

Di dunia *front-end*, istilah *framework* pada umumnya merujuk ke kumpulan standar, konvensi, dan daftar fungsionalitas (*library*) yang membantu pengembang untuk membangun aplikasi web dengan lebih efisien [9]. *Framework* menyediakan kerangka kerja dan struktur siap guna yang fleksibel, sehingga pengembang tidak perlu membangun dari awal setiap kali memulai pembangunan sebuah aplikasi [10].

Angular merupakan salah satu *framework* yang memberikan berbagai kemudahan dalam proses pengembangan aplikasi web. Salah satu fitur utama Angular adalah penggunaan HTML sebagai *template*, yang dilengkapi dengan berbagai fungsionalitas khusus. Angular menggunakan pendekatan deklaratif untuk membangun antarmuka pengguna, di mana pengembang mendefinisikan struktur aplikasi dengan menghubungkan komponen bersama-sama. Sistem modul Angular memungkinkan pengembang untuk memecah kode menjadi modul-modul yang terpisah, sehingga mempermudah pengembangan dan pemeliharaan aplikasi yang besar. Modul-modul ini dapat digunakan untuk mengimpor fungsi dan komponen yang diperlukan ke dalam aplikasi [11].

Angular merupakan generasi terbaru dari *framework* AngularJS dengan berbagai peningkatan. AngularJS dibangun dengan bahasa JavaScript yang mendorong pemisahan data dengan antarmuka, sedangkan Angular dibangun dengan bahasa TypeScript yang mengadopsi arsitektur berbasis komponen yang memungkinkan penempelan data atau yang biasa disebut *data binding*. Selain karena Google menggunakan bahasa TypeScript dalam penciptaan *framework* Angular, alasan kuat pendorong penggunaan bahasa tersebut datang dalam peningkatan strukturisasi kode pemrograman dengan konsep *type safety* yang diadopsi [12]. Performa Angular telah diukur 7 kali lebih cepat dibanding AngularJS karena Angular memanipulasi struktur DOM secara langsung tanpa membuat dokumen HTML terlebih dahulu [13].

Secara keseluruhan, Angular adalah *framework* yang penuh dengan fitur dan pelengkap untuk membantu proses pembangunan aplikasi web. Dengan fitur-fitur canggihnya, Angular memungkinkan pengembang untuk mengembangkan aplikasi web yang solid, cepat, dan mudah dipelihara.

E. Directive pada Framework Angular

Directive merupakan salah satu fitur dari Angular yang digunakan untuk memperluas dan mengubah perilaku elemen HTML atau bagian-bagian lain dalam tampilan aplikasi. *Directive* memungkinkan pengembang untuk membuat kode yang dapat digunakan kembali dan mengisolasi logika dan tampilan terkait [14]. Ada beberapa jenis *directive* yang disediakan Angular:

1) *Component Directive*: *Directive* yang digunakan untuk menciptakan sebuah komponen. Dekorator "@Component" digunakan untuk mendefinisikan sebuah komponen dengan data yang diperlukan [14].

2) *Structural Directive: Structural directive* mengubah struktur DOM untuk menambahkan atau menghapus elemen. *Structural directive* yang paling sering digunakan adalah "ngIf" dan "ngFor". "ngIf" dapat menghapus atau menambahkan elemen dari DOM berdasarkan kondisi yang ditentukan, sedangkan "ngFor" digunakan untuk menampilkan elemen berulang kali sebanyak jumlah data dalam sebuah *array* [14].

3) *Attribute Directive: Attribute directive* memanipulasi tampilan atau perilaku elemen dengan mengubah nilai atributnya. Contohnya, "ng*Style*" digunakan untuk mengubah *style* elemen CSS berdasarkan kondisi atau ekspresi yang ditentukan pada definisi atribut, dan "ng*Class*" digunakan untuk menambahkan atau menghapus kelas CSS berdasarkan kondisi atau ekspresi serupa [14].

Selain *directive* bawaan Angular, pengembang juga dapat membuat yang *custom directive* sesuai kebutuhan aplikasi. *Directive* buatan pengembang ini memungkinkan ditambahkannya perilaku unik aplikasi ke elemen HTML yang tidak tersedia secara langsung. Untuk membuat *directive* sendiri, pengembang perlu menggunakan "*@Directive*" dan mengimplementasikan logika yang diinginkan di dalamnya [14].

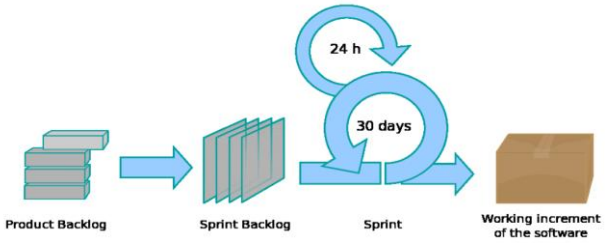
F. Metode Agile

Metode *agile* merupakan metode manajemen pengembangan proyek yang memungkinkan dilakukannya perubahan setiap saat [15]. Hal ini disebabkan oleh konsep siklus pengembangan yang menjadi pilar metode agile. Siklus ini terdiri dari tahap perencanaan, tahap perancangan, tahap pengembangan, tahap pengujian, tahap peluncuran, dan tahap evaluasi. Siklus tersebut diulang terus menerus selama masa pengembangan. Dengan begitu, pelaksanaan iterasi dan perubahan dapat dilakukan dengan mudah karena kebutuhan dan cakupan proyek ditentukan di setiap awal siklus, tidak seperti metode lain seperti metode *waterfall* yang penentuan kebutuhan proyek dilakukan di awal pengembangan [16].

G. Scrum

Scrum merupakan salah satu kerangka kerja yang menggunakan pendekatan metode *agile*. Model ini mengekspansi konsep siklus pada metode *agile*. Dalam *scrum*, terdapat istilah sprint yang merupakan siklus pendamping iterasi yang dilaksanakan setiap 24 jam. Satu siklus sprint pada *scrum* biasanya ditetapkan sepanjang 1-4

minggu. Di awal sprint, ditentukan sebuah *backlog* yang berisi berbagai kebutuhan proyek yang mungkin dikerjakan selama 1 sprint ke depan. Model *scrum* mengedepankan aspek kecepatan dan adaptabilitas pengembangan dengan dilakukannya pemeriksaan dan penyesuaian secara terus menerus terhadap kebutuhan proyek [15]. *Scrum* mendorong pengembang untuk membangun aplikasi sedikit demi sedikit untuk membatasi alokasi tenaga kerja ke hal-hal yang pasti dibutuhkan [17]. Ilustrasi model *scrum* dapat dilihat pada Gambar 4.



Gambar 4. Ilustrasi model *scrum* [18]

III. *METODOLOGI*

Dalam proses pengembangan UIIAdmisi, digunakan pendekatan manajemen proyek bermetode *agile*. Adapun model spesifik yang digunakan dalam pengembangan UIIAdmisi yaitu *scrum*. Pelaksanaan dilakukan melalui tahapan yang telah didefinisikan pada metode *agile* yang terdiri dari tahap perencanaan, tahap perancangan, tahap pengembangan, tahap pengujian, dan tahap peluncuran.

A. *Tahap Perencanaan*

Tahap perencanaan dimulai di luar iterasi sprint dengan mendefinisikan seluruh pekerjaan yang perlu diselesaikan untuk memenuhi kebutuhan proyek. Daftar pekerjaan tersebut diberi istilah *product backlog*. Di awal sprint, dilakukan evaluasi terhadap *product backlog* dan memindahkan pekerjaan yang mungkin diselesaikan pada masa 1 sprint ke depan ke dalam sprint *backlog*. Dari segi teknis, sprint *backlog* biasanya didampingi dengan berbagai artefak berbagai jenis diagram untuk merepresentasikan rencana dan hasil yang ingin dicapai. Misalnya, pada sebuah masa sprint dibutuhkan pengembangan fitur pengisian dokumen penunjang pendaftaran. Artefak yang dihasilkan berdasarkan fitur tersebut dapat direpresentasikan melalui *activity* diagram yang mendeskripsikan interaksi antara pengguna dengan sistem. Dengan mendasari struktur aplikasi dengan *activity* diagram, pengembang dapat meningkatkan modularisasi dan *readability* pada *source code* [19].

Pada masa sprint eksekusi kasus yang diangkat makalah ini, *backlog* yang ditentukan mencakup implementasi antar-muka isian data nilai rapor, formulir data prestasi, peringatan status pengembalian dana undur diri, dan formulir isian data rekening pengembalian.

B. *Tahap Perancangan*

Sebelum implementasi fitur dapat dilakukan, diperlukan rancangan tampilan dan alur interaksi aplikasi. Perancangan antarmuka aplikasi UIIAdmisi dilakukan menggunakan aplikasi Figma. Rancangan pada Figma menjadi panutan dan aturan yang harus pengembang *front-end* ikuti saat melakukan implementasi. Hasil rancangan antarmuka akan digunakan pada tahap pengujian untuk mengukur kesesuaian hasil implementasi dengan desain.

C. *Tahap Pengembangan*

Dalam siklus sprint, tahap pengembangan paling banyak memakan waktu yang dialokasikan. Sebagian besar porsi pengembangan UIIAdmisi berada pada tahap ini. Dari segi *front-end*, tahap pengembangan mencakup organisasi struktur aplikasi, penulisan kode, dan pengujian pertama implementasi.

D. *Tahap Pengujian*

Setiap pengembang selesai mengimplementasi suatu fitur, akan dilakukan pengujian secara manual. Pengujian yang dilakukan berupa pengetesan fungsionalitas tujuan, pencarian celah, pengukuran kesesuaian dengan rancangan, dan uji performa. Proses pengujian dilakukan secara terisolasi berdasarkan fitur yang dikembangkan.

E. *Tahap Peluncuran*

Setelah lolos pengujian, aplikasi akan dirilis ke publik pada alamat uii.admisi.ac.id. Proses perilisan ini dinamakan *deployment*. UIIAdmisi memanfaatkan fitur *Continuous Integration* pada Gitlab untuk menyajikan aplikasi ke pengguna. Proses ini berlangsung secara otomatis setiap pengembang melakukan perubahan pada *source code* di Gitlab.

IV. HASIL DAN PEMBAHASAN

Dalam siklus sprint, tahap pengembangan paling banyak memakan waktu yang dialokasikan. Sebagian besar porsi pengembangan UIIAdmisi berada pada tahap ini. Dari segi *front-end*, tahap pengembangan mencakup organisasi struktur aplikasi, penulisan kode, dan pengujian pertama implementasi. Pada tahap ini akan dijelaskan cara penggunaan *directive* pada beberapa halaman UIIAdmisi yang menjadi topik bahasan.

1) *Halaman Isi Data*: Universitas Islam Indonesia menyediakan berbagai pola seleksi yang berbeda. Setiap pola seleksi memiliki data tambahan masing-masing yang perlu diisi pendaftar, seperti nilai rapor dan dokumen penunjang. Halaman isi data memungkinkan pendaftar untuk mengisi data tersebut melalui sebuah formulir dinamis. Melihat banyaknya variasi pola seleksi, pembahasan akan dibatasi pada pola seleksi Penelusuran Siswa Berprestasi (PSB). Penerimaan melalui pola seleksi PSB mengharuskan pendaftar untuk mengumpulkan data tambahan berupa nilai rapor dan riwayat prestasi yang dilengkapi dengan bukti unggahan dokumen rapor dan sertifikat prestasi. Untuk formulir isian data nilai rapor, dibutuhkan antarmuka berupa kumpulan *input field* dalam bentuk matriks dua dimensi yang menunjukkan relasi antara label semester dan mata pelajaran. Formulir tersebut dapat diimplementasikan dengan menggunakan salah satu *structural directive* yaitu “ngFor”. *Directive* tersebut dapat ditambahkan pada *tag* HTML untuk menampilkan elemen tersebut berulang kali. Jumlah perulangan yang dilakukan terikat dengan ukuran sebuah *array* yang didefinisikan pada fail TypeScript. Mengingat formulir rapor yang bersifat dua dimensi, dibutuhkan 2 *array* berbeda; *array* semester dan *array* mata pelajaran. Ukuran *array* semester bersifat statis untuk merepresentasikan semester 1-5, sedangkan ukuran *array* mata pelajaran bergantung pada jurusan pendidikan terakhir pendaftar yang diambil dari *back-end*. Deklarasi kedua *array* tersebut dapat dilihat pada kutipan kode pada Gambar 5 dan Gambar 6.

```
initSemesterList() {
  //Mengisi array dengan 5 nilai kosong
  this.semesterList = Array.from(''.repeat(5))
}
```

Gambar 5. Deklarasi *array* semester

```
initSubjectList() {
  //Hit API daftar mata pelajaran
  this.psbSvc.list(PSBServiceType.PSB_HIGH_SCHOOL_SUBJECT, '', this.subjectListEndpoint)
    .subscribe(response => {
      //Memasukkan data luaran ke array mata pelajaran
      this.subjectList = response.data.normal;
    })
}
```

Gambar 6. Deklarasi *array* mata pelajaran

Pada fail HTML, kedua *array* tersebut digunakan pada *nested for loop* atau perulangan bertumpuk menggunakan "ngFor" seperti pada Gambar 7. Perhatikan penggunaan variabel i dan j untuk menyimpan indeks perulangan. Kedua indeks tersebut digunakan oleh komponen *input field* data nilai rapor untuk mengingat posisinya dalam matriks supaya interaksi perubahan data berjalan pada elemen yang benar.

```
<!-- Perulangan array mata pelajaran-->
<div class="row margin-left-0 margin-right-0" *ngFor="let subject of subjectList; index as i">
    <div class="entry-cell title-cell">
        <!-- Label nama mata pelajaran -->
        <label class="control-label control-required"> {{subject.nama_mapel}}</label>
    </div>
    <!-- Perulangan array semester -->
    <div *ngFor="let semester of semesterList; index as j" class="entry-cell">
        <!-- Komponen input field nilai rapor -->
        <uii-form-score-entry [action]="action" [placeholder]="placeholder[i][j]"
            [score]="scoreData[i][j]" (onFormChange)="onScoreChange(i,j,$event)">
        </uii-form-score-entry>
    </div>
</div>
```

Gambar 7. Perulangan bertumpuk formulir data nilai rapor

Selain data nilai rapor, terdapat formulir untuk mengisi data riwayat prestasi pendaftar. Berdasarkan rancangan, tidak ada batasan jumlah prestasi yang dapat ditambahkan oleh pendaftar. Dibutuhkan fungsionalitas tambah hapus riwayat prestasi. Fitur tersebut dapat diimplementasikan dengan cara "ngFor" yang sama, dengan tambahan fungsi manipulasi elemen *array* pengikat. Ketika pengguna menekan tombol tambah prestasi, *array* akan mendapat tambahan nilai baru berupa objek formulir kosong. Sebaliknya, tombol hapus prestasi akan menghilangkan data objek formulir *array* pada indeks yang dipilih. Kedua fungsi tersebut dapat dilihat pada Gambar 8.

```
achievementList = [];

addNewAchievement() {
  //Menambahkan objek dengan format sesuai formulir di akhir array
  this.achievementList = [...this.achievementList, {
    kd_kategori: '',
    kd_prestasi: '',
    nama_kegiatan: '',
    kd_peringkat: '',
    kd_tingkat: '',
  }];
  //Mengirim sinyal perubahan formulir
  this.onFormChange();
}

deleteAchievement($event) {
  //Menghilangkan data prestasi pada indeks
  this.achievementList = this.achievementList.filter((x, i) => i !== $event.index);
  //Menerapkan perubahan data prestasi ke komponen
  $event.component.patchValues(this.achievementList[$event.index]);
  //Mengirim sinyal perubahan formulir
  this.onFormChange();
}
```

Gambar 8. Fungsi tambah dan hapus data prestasi

Pada rancangan antarmuka, terdapat tambahan kondisi saat pendaftar belum menambahkan data prestasi. Dengan kosongnya data prestasi, tidak ada formulir yang ditunjukkan, melainkan sebuah tombol dengan gaya berbeda yang berfungsi untuk menambahkan data prestasi pertama. *Directive* yang dapat digunakan untuk mengakomodasi kondisi tersebut yaitu "ngIf". *Tag* HTML yang diberi "ngIf" akan ditampilkan hanya pada saat kondisi terpenuhi. Dengan begitu, perlu dilakukan pengecekan kondisi ukuran *array* prestasi sama dengan 0. Penerapan pengecekan kondisi tersebut dapat dilihat pada Gambar 9.

```
<div class="row margin-left-0 margin-right-0">
    <h5>Data prestasi</h5>
    <!-- Tampilkan tombol tambah prestasi saat ukuran array prestasi sama dengan 0 -->
    <button *ngIf="achievementList.length === 0" class="btn btn-success margin-top-10 margin-bottom-20"
        (click)="addNewAchievement()">
        Tambah prestasi
    </button>
    <!-- Tampilkan komponen formulir prestasi saat ukuran array prestasi lebih dari 0 -->
    <uii-psb-form-achievement-entry *ngFor="let achievement of achievementList; index as i" [action]="action"
        [index]="i" [isLastItem]="i===achievementList.length - 1" (onAddNewAchievement)="addNewAchievement()"
        (onDeleteAchievement)="deleteAchievement($event)" (onFormChanged)="onAchievementChange($event)"
        [achievementOptions]="achievementOptions" [achievement]="achievement">
    </uii-psb-form-achievement-entry>
</div>
```

Gambar 9. Pengecekan kondisi dengan "ngIf"

2) *Halaman Undur Diri*: Halaman undur diri memungkinkan pendaftar untuk mengajukan permohonan pengembalian dana pendaftaran. Pendaftar hanya akan mendapat pengembalian dana jika permohonan dilakukan sebelum melewati tenggat waktu yang sudah ditentukan. Rancangan antarmuka memerlukan sebuah kotak peringatan pada halaman pengajuan undur diri. Diperlukan dua gaya peringatan yang berbeda berdasarkan status kadaluwarsa pengajuan undur diri untuk dapat mengkomunikasikan peringatan dengan jelas. Saat status masih belum kadaluwarsa, kotak peringatan berwarna kuning muda dengan *icon* berwarna kuning tua, sedangkan saat status sudah melewati tenggat, kotak peringatan berwarna merah dengan *icon* merah tua. *Attribute directive* dapat digunakan untuk mengimplementasi fungsionalitas tersebut. Untuk mengatur warna kotak, digunakan "ng*Class*" untuk menambahkan *class* CSS pada elemen berdasarkan sebuah variabel *boolean* "isExpired" yang menyimpan kondisi kadaluwarsa. Jika kondisi tersebut terpenuhi, maka "ng*Class*" akan menambahkan *class* CSS bernama "expired-infobox" pada elemen kotak. Untuk pewarnaan *icon*, dibutuhkan nilai warna yang bersifat *inline* dalam bentuk kode *hex*. Oleh karena itu, *directive* yang lebih tepat digunakan yaitu "ng*Style*". Fungsionalitas *directive* ini serupa dengan "ng*Class*", hanya saja "ng*Style*" bukan menambahkan *class*, namun *style*. Dekorator "ng*Style*" ditulis menggunakan pengkondisian *ternary* dengan dua kemungkinan nilai yang dihasilkan. Penerapan kedua *directive* tersebut dapat dilihat pada Gambar 10 yang menunjukkan dokumen HTML kotak peringatan status kadaluwarsa undur diri.

```
<!-- Menambahkan kelas "expired-infobox" saat variabel isExpired = true -->
<div class="infobox" [ngClass]="{'expired-infobox': isExpired}">
    <!-- Menentukan warna icon berdasarkan variabel isExpired -->
    <i class="fa fa-info-circle" [ngStyle]="isExpired ? 'color: #FF9E00;' :'color: #F44444;'">
    </i>
    <div style="width:100%;" class="margin-left-10">
        <!-- Paragraf yang ditampilkan saat variabel isExpired = true -->
        <p *ngIf="isExpired">
            Pengajuan undur diri <b>melewati batas {{deadlineDate}},</b>
            berdasarkan ketentuan yang berlaku maka <b>TIDAK MENDAPAT PENGEMBALIAN DANA REGISTRASI.</b>
        </p>
        <!-- Paragraf yang ditampilkan saat variabel isExpired = false -->
        <p *ngIf="!isExpired">
            Pengajuan undur diri sebagai calon mahasiswa dengan pengembalian dana registrasi dapat dilakukan
            paling lambat <b>{{deadlineDate}}.</b>
        </p>
    </div>
</div>
```

Gambar 10. Penerapan *attribute directive*

Penerapan *directive* selanjutnya di halaman undur diri dapat dijumpai pada formulir detail rekening untuk pengembalian dana. Data detail rekening mencakup nama pemilik rekening, nomor rekening, dan nama bank. Isian

nomor rekening membutuhkan adanya batasan tipe data berupa angka. *Custom directive* dapat digunakan untuk memenuhi kebutuhan tersebut. Dibuat fail TypeScript baru yang dinamakan "allownumbersonly.*directive*.ts" untuk menampung *custom directive* yang dapat ditempelkan pada elemen *input* HTML untuk membatasi isian data pada elemen tersebut menjadi angka. Untuk mendeklarasikan *custom directive*, digunakan dekorator "*@Directive*" yang di dalamnya didefinisikan sebuah *selector* untuk menjadi alamat referensi *directive* tersebut agar dapat digunakan oleh komponen lain. Kemudian, logika pembatasan angka dimasukkan ke sebuah *class* yang diekspor dengan nama "AllowNumbersOnly*Directive*". Dalam *class* tersebut, digunakan dekorator "@HostListener" untuk menerima sinyal "onInputChange" dari DOM setiap kali terjadi perubahan isian data pada elemen *input*. Setiap sinyal tersebut diterima, dilakukan manipulasi terhadap data isian dengan menggunakan *regular expression* yang mendefinisikan formula pembatas tipe teks yang diperbolehkan. Fail "allownumbersonly.*directive*.ts" dapat dilihat pada Gambar 11.

```
//Import komponen yang dibutuhkan
import { Directive, ElementRef, HostListener } from '@angular/core';

//Deklarasi custom directive
@Directive({
    selector: '[AdmissionAllowNumbersOnly]'
})
export class AllowNumbersOnlyDirective {
    constructor(private el: ElementRef) { }
    // Menerima sinyal perubahan isian teks pada elemen input
    @HostListener('input', ['$event']) onInputChange(event) {
        // Mengambil elemen input
        let element = this.el.nativeElement;
        // Menyimpan teks yang diisi pada elemen input
        const initialValue = element.value;

        // Menghapus huruf pada teks dengan menggunakan formula regular expression
        element.value = initialValue.replace(/[^0-9]*/g, '');
        // Mengirim sinyal perubahan teks
        element.dispatchEvent(new Event('input'));
    }
}
```

Gambar 11. *Custom directive* pembatas tipe data isian

Custom directive yang sudah dibuat dapat ditempelkan pada elemen *input* pada fail HTML halaman undur diri dengan mencantumkan *selector* seperti pada Gambar 12.

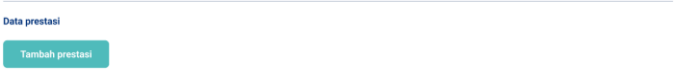
```
<label class="form-label">
    Nomor rekening
</label>
<input type="text" class="form-control"
    placeholder="Tulis nomor rekening" AdmissionAllowNumbersOnly/>
```

Gambar 12. Elemen *input* dengan *custom directive*

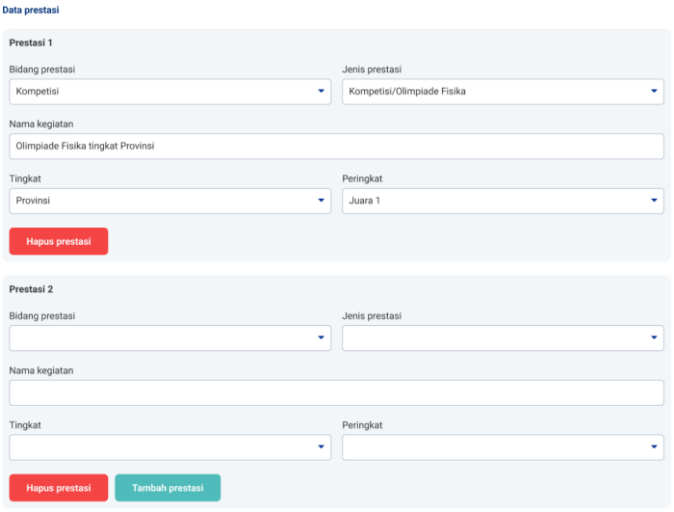
Hasil yang diperoleh setelah pengembangan selesai berupa aplikasi web UIIAdmisi yang dapat dibuka pada alamat admisi.uii.ac.id. Adapun hasil pengembangan halaman isi data dapat dilihat pada Gambar 13.

Gambar 13. Matriks isian data rapor

Dengan beberapa baris kode sederhana ditambah pemanfaatan fitur *directive* pada *framework* Angular, tercipta antarmuka yang cukup kompleks. Tanpa menggunakan *directive*, pengembang perlu menulis kode HTML yang sama secara berulang kali dengan jumlah perulangan yang terbatas dan tidak dinamis. Dengan menggunakan *framework* Angular, pengembang bahkan dapat memanipulasi jumlah perulangan berdasarkan *input* dari pengguna seperti pada hasil pengembangan fitur isi data prestasi yang ditunjukkan pada Gambar 14 dan Gambar 15.



Gambar 14. Tampilan saat data prestasi kosong



Gambar 15. Tampilan formulir data prestasi

Perhatikan kedua formulir data prestasi yang ditampilkan. Sekilas, tampak bahwa kedua formulir tersebut dibuat oleh pengembang secara manual. Kenyataannya, formulir tersebut merupakan sebuah komponen yang digunakan berulang kali. Hal ini merupakan kelebihan lain dari *framework* Angular yang memungkinkan penggunaan ulang komponen di mana pun. Jika dibutuhkan, komponen formulir tersebut dapat digunakan di seluruh bagian aplikasi.

Framework Angular memudahkan pembangunan antarmuka yang dinamis dengan menggunakan *attribute directive*, seperti pada hasil pengembangan kotak peringatan status undur diri pada Gambar 16 yang tampil saat tanggal belum melewati tenggat kadaluwarsa undur diri, dan Gambar 17 yang menunjukkan kotak peringatan yang tampil saat tenggat masa undur diri telah terlampaui.

Gambar 16. Kotak peringatan undur diri sebelum tenggat

Gambar 17. Kotak peringatan undur diri setelah tenggat

Terakhir, *framework* Angular menyediakan opsi kepada pengembang untuk membuat *custom directive* sesuai kebutuhan. Pada formulir data rekening UIIAdmisi, digunakan *custom directive* pembatas tipe data teks menjadi angka. Hasil penggunaan *custom directive* diperlihatkan pada Gambar 18.

Gambar 18. Formulir data rekening pengembalian

Pada elemen *input* nomor rekening tersebut, pengguna hanya diperbolehkan memasukkan angka, dan saat pengguna menekan tombol huruf tidak akan terjadi apa pun. Dengan pelaksanaan implementasi fitur tersebut pada seluruh elemen *input* yang sesuai, tingkat *user error* akan berkurang. Hal ini memenuhi kriteria fundamental penciptaan pengalaman pengguna yang baik yaitu *stability*, *readability*, dan *security* [20].

Seluruh fitur UIIAdmisi yang menjadi syarat pembukaan penerimaan mahasiswa baru berhasil dibangun tepat waktu. Adapun masa pengembangan UIIAdmisi dimulai pada bulan Agustus 2022 dengan masa tenggat peluncuran aplikasi seiring dengan pembukaan penerimaan mahasiswa baru pada bulan Desember 2022. Faktor utama yang memungkinkan terkejarnya target peluncuran aplikasi tersebut yaitu pemanfaatan *framework* Angular. Tidak hanya itu, pengalaman pengguna berbagai fitur UIIAdmisi terdorong dengan penggunaan *directive* untuk mewujudkan antarmuka yang dinamis, stabil, aman, dan jelas.

Setelah terjun dalam proses pengembangan aplikasi dengan menggunakan *framework* Angular, terdapat beberapa kelebihan yang dirasakan dibandingkan saat tidak menggunakan *framework* tersebut:

1) *Fleksibilitas tinggi*: Dengan menggunakan *custom directive*, pengembang dapat mengatur implementasi sesuai kebutuhan aplikasi. Tidak hanya itu, pengembang juga dapat dengan mudah mengakomodasi berbagai perubahan yang terjadi karena *custom directive* yang merupakan sebuah komponen yang berdiri sendiri dan mudah untuk dimodifikasi.

2) *Kerapian dan kecepatan pengembangan*: Angular menitikberatkan konsep penggunaan ulang komponen. Pembuatan komponen yang dapat digunakan tidak hanya sekali meminimalisir adanya duplikasi kode, sehingga *source code* terlihat rapi. Selain itu, penggunaan ulang komponen secara otomatis meningkatkan efisiensi pengembangan.

Sederhananya, Angular menyediakan berbagai fitur khususnya *directive* yang memberikan berbagai kemudahan pada proses pengembangan sebuah aplikasi berbasis *web*.

V. KESIMPULAN

Berdasarkan berbagai hasil dan pembahasan yang telah dikemukakan pada bagian sebelumnya, dapat disimpulkan bahwa telah berhasil dilakukan pemanfaatan *structural directive*, *attribute directive*, dan *custom directive* pada *framework* Angular untuk mengembangkan aplikasi penerimaan mahasiswa baru UIIAdmisi. Selain itu, pendekatan *scrum* yang merupakan salah satu model *agile* memungkinkan berjalannya proses pengembangan yang teratur sehingga target masa peluncuran aplikasi dapat tercapai.

Melihat lancarnya pengembangan UIIAdmisi, sudah sepantasnya digunakan *framework* Angular untuk proyek-proyek selanjutnya. Terdapat insentif untuk meningkatkan penggunaan fitur *directive* untuk memperkaya proses pengembangan seluruh fitur yang akan diimplementasikan. Besar dorongan terhadap Badan Sistem Informasi UII untuk mempertimbangkan pengadopsian metode dan teknologi yang dibahas pada makalah ini untuk memperbarui jajaran aplikasi yang sudah ada. Selebihnya, untuk mendorong tingkat kerapian dan konsistensi struktur aplikasi, perlu dibangun sebuah standar yang berisi berbagai komponen untuk digunakan kembali pada proyek-proyek selanjutnya.

REFERENSI

[1] A. Caputo, S. Pizzi, M. M. Pellegrini and M. Dabić, "Digitalization and business models: Where are we going? A science map of the field," *Journal of Business Research*, vol. 123, pp. 489-501, December 2021.

[2] D. Dinh and Z. Wang, "Modern front-end web development: how libraries and frameworks transform everything," *Theseus*, 2020.

[3] A. Aizpurua, S. Harper and M. Vigo, "Exploring the Relationship between Web Accessibility and User Experience," *International Journal of Human Computer Studies*, vol. 91, pp. 13-23, 2016.

[4] V. Kotaru, "Angular: Directives," in *Angular for Material Design*, Apress, 2019, pp. 95-108.

[5] Y. L. Antonucci, A. Fortune and M. Kirchmer, "An examination of associations between business process management capabilities and the benefits of digitalization: all capabilities are not equal," *Business Process Management Journal*, vol. 27, no. 1, pp. 124-144, 2021.

[6] S. M. Prasetiyo, M. I. P. Nugroho, R. L. Putri and O. Fauzi, "Pembahasan Mengenai Front-End Web Developer dalam Ruang Lingkup Web Development," *BULLET*, vol. 1, no. 6, pp. 1015-1020, December 2022.

[7] R. Tomar and S. Dangi, "Document Object Model," in *Javascript Syntax and Practices*, Chapman and Hall/CRC, 2021, pp. 145-178.

[8] N. F. Malik, A. Nadeem and M. A. Sindhu, "Achieving State Space Reduction in Generated Ajax Web Application State," *Intelligent Automation and Soft Computing*, vol. 33, no. 1, pp. 429-455, 2022.

[9] K. Bielak, B. Borek and B. Plechawska-Wójcik, "Web application performance analysis using Angular, React and Vue.js frameworks," *Journal of Computer Sciences Institute*, vol. 23, pp. 77-83, 2022.

[10] H. Singh, T. Srivastava and D. K. Shukla, "Angular Web Application," *YMER Digital*, vol. 21, no. 5, pp. 604-609, 2022.

[11] Angular, "Introduction to Angular concepts," 2020. [Online]. Available: https://angular.io/guide/architecture.

[12] M. Clow, "AngularJS vs. Angular (Old vs. New)," in *Angular 5 Projects*, Apress, 2018, pp. 15-25.

[13] M. Joshi, "Angular vs AngularJS," Browser Stack, [Online]. Available: https://www.browserstack.com/guide/angular-vs-angularjs.

[14] Angular, "Built-in directives," 2020. [Online]. Available: https://angular.io/guide/built-in-directives.

[15] S. H. Nova, A. P. Widodo and B. Warsito, "Analisis Metode Agile pada Pengembangan Sistem Informasi Berbasis Website: Systematic Literature Review," *Techno.Com*, vol. 21, no. 1, pp. 139-148, 2022.

[16] M. Alexander, "Agile vs. waterfall: Project methodologies compared," *Cio*, pp. 1-4, 2020.

[17] F. Fowler, "What Is Scrum?," in *Navigating Hybrid Scrum Environments*, Apress, 2019, pp. 3-8.

[18] D. Spišák and J. Lang, "Activity Diagram as an Orientation Catalyst within Source Code," *Acta Polytechnica Hungarica*, vol. 18, no. 3, pp. 127-146, 2021.

[19] A. Koshravi, T. J. Gandomani and H. Fahimian, "Introduction of Scrum in An Elite Team: A Case Study," *Journal of Software*, vol. 12, no. 4, pp. 173-179, 2017.

[20] E. Stull, UX Fundamentals for Non-UX Professionals, Apress, 2018.