

**Perbaikan Bug dan Pengembangan Zendesk API Tokopedia.**

**PT TOKOPEDIA**

**Wisma 77 Tower 2 Lantai 2**

**Jl. Letjen S. Parman Kav. 77 Jakarta Barat.**

**Periode: 18 Juli 2016 – 02 September 2016**

Oleh:

|  |  |
| --- | --- |
| Hanif Sudira | 5113100184 |
|  |  |

Pembimbing Jurusan

Hudan Studiawan, S.Kom., M.Kom.

Pembimbing Lapangan

Calina Tan, S.Kom.

JURUSAN TEKNIK INFORMATIKA

Fakultas Teknologi Informasi

Institut Teknologi Sepuluh Nopember

Surabaya 2016

KERJA PRAKTIK – KI141330

*[Halaman ini sengaja dikosongkan]*

**

**Perbaikan Bug dan Pengembangan Zendesk API Tokopedia.**

**PT TOKOPEDIA**

**Wisma 77 Tower 2 Lantai 2**

**Jl. Letjen S. Parman Kav. 77 Jakarta Barat.**

**Periode: 18 Juli 2016 – 02 September 2016**

Oleh:

|  |  |
| --- | --- |
| Hanif Sudira | 5113100184 |
|  |  |

Pembimbing Jurusan

Hudan Studiawan, S.Kom. M.Kom.

Pembimbing Lapangan

Calina Tan, S.Kom.

JURUSAN TEKNIK INFORMATIKA

Fakultas Teknologi Informasi

Institut Teknologi Sepuluh Nopember

Surabaya 2016

KERJA PRAKTIK – KI141330

*[Halaman ini sengaja dikosongkan]*

# LEMBAR PENGESAHAN

**KERJA PRAKTIK**

***Perbaikan Bug dan Pengembangan Zendesk API Tokopedia***

Oleh:

|  |  |
| --- | --- |
| **Hanif Sudira** | 5113100184 |
|  |  |

Disetujui oleh Pembimbing Kerja Praktik:

|  |  |
| --- | --- |
| 1. Hudan Studiawan, S.Kom. M.Kom. NIP. 198705112012121003 | ................................ (Pembimbing Jurusan) |
|  |  |
| 1. Calina, S. Kom. | ................................ (Pembimbing Lapangan) |

**SURABAYA**

**SEPTEMBER, 2016**

*[Halaman ini sengaja dikosongkan]*

**Perbaikan Bug dan Pengembangan Zendesk API Tokopedia**

**Nama Mahasiswa : Hanif Sudira**

**NRP : 5113100184**

**Jurusan : Teknik Informatika FTIf-ITS**

**Pembimbing Jurusan : Hanif Sudira, S.Kom., M.Kom.**

**Dosen Pembimbing II : Calina Tan, S.Kom.**

# Abstrak

Testing pada *software* merupakan hal yang wajib dilakukan guna menjamin kualitas *software*. Terlebih apabila *software* tersebut merupakan pesanan suatu klien, maka testing pada *software* mutlak dilakukan. Terdapat banyak jenis testing pada software. Salah satunya yaitu UAT (*User Acceptance Test*). UAT merupakan tes yang dilakukan guna menjaminnya *software* yang dibuat sesuai dengan kebutuhan user.

Terdapat *tools* untuk membuat UAT yang otomatis. Salah satunya yaitu Protractor. Saat ini belum ada IDE (*Integrated Development Environment*) yang khusus untuk Protractor. Maka dibuat juga IDE yang dikhususkan untuk memprogram Protractor. Adapun IDE tersebut dibuat menggunakan AngularJS dan MeteorJS.

Adapun ujicoba yang dilakukan terbagi menjadi dua bagian. Yaitu ujicoba pada program Protractor dan ujicoba pada pembuatan IDE menggunakan Meteor. Hasil dari ujicoba tersebut telah berhasil dilaksanakan sesuai dengan deskripsi proyek pada kerja praktik.

***Kata kunci: Protractor, AngularJS, MeteorJS, UAT (User Acceptance Test), IDE (Integrated Development Environment)***

*[Halaman ini sengaja dikosongkan]*

# KATA PENGANTAR

Puji dan syukur penulis hantarkan ke kehadirat Allah Subhanahu Wa Ta’alla atas rahmat dan karuniaNya, penulis dapat menyelesaikan rangkaian Kerja Praktek di PT. Tokopedia yang dimulai pada tanggal 18 Juli 2016 hingga 02 September 2016, serta dapat menyelesaikan laporan Kerja Praktek sesuai jadwal.

Kerja Praktek di Jurusan Teknik Informatika, Institut Teknologi Sepuluh Nopember Surabaya merupakan salah satu kegiatan wajib yang harus dilaksanakan oleh mahasiswa yang telah menempuh minimal 90 SKS. Penerapan kerja praktek ini sendiri pada hakikatnya adalah mengajarkan kepada kita untuk menghadapi situasi sebenarnya yang tidak beda jauh dengan kondisi disaat kita bekerja nantinya.

Pelaksanaan kerja praktek ini dapat dilaksanakan sampai selesai berkat bimbingan dan saran dari berbagai pihak. Pada kesempatan ini, kami ingin menyampaikan terima kasih kepada :

1. Allah Subhanahu Wa Ta’ala, karena tanpa ijin dari-Nya, penulis tidak akan mampu menyelesaikan Kerja Praktek ini.
2. Kedua orang tua penulis yang telah memberikan doa dan dukungannya.
3. Bapak Darlis Herumurti, S.Kom, M.Kom, Dr.Eng. selaku Ketua Jurusan Teknik Informatika ITS.
4. Dosen wali penulis, Bapak Royyana M Ijtihadie, S.Kom.,M.Kom.,Ph.D atas dukungan selama berada di Teknik Informatika.
5. Dosen pembimbing penulis, Bapak Hudan Studiawan, S.Kom., M.Kom. atas dukungan dan motivasinya saat sebelum sampai berakhirnya proses kerja praktek kami.
6. Bapak Radityo Anggoro, S.Kom, M.Sc selaku koordinator kerja praktek Jurusan Teknik Informatika ITS atas segala bimbingan dan arahannya selama ini yang sangat membantu kami untuk segera menyelesaikan kerja praktek ini.
7. Ibu Calina Tan,S.Kom sebagai pembimbing kerja praktik.
8. Bapak William Tanuwijaya dan Bapak Leontinus Alpha Edison dan seluruh nakama PT. Tokopedia untuk pengalaman, atmosfir, dan kesempatan bekerja sama yang sangat berharga.
9. Bapak dan Ibu dosen jurusan Teknik Informatika ITS yang telah mencurahkan semua ilmu yang sangat besar manfaatnya selama proses belajar.
10. Karyawan dan staf administrasi Teknik Informatika yang membantu dalam proses pengerjaan dan administrasi kerja praktek.
11. Rekan-rekan Teknik Informatika ITS angkatan 2013 atas segala dorongan semangat dan ilmu yang diberikan.
12. Teman-teman warga Teknik Informatika ITS.
13. Pihak-pihak lain yang tidak sempat penulis sebutkan, yang telah membantu dalam kelancaran pengerjaan kerja praktek kami.

Apabila terdapat kesalahan dan kekurangan dalam penulisan laporan kerja praktek ini, penulis mengharapkan kritik dan saran yang membangun sebagai perbaikan di masa akan datang dan kami meminta maaf yang sebesar-besarnya. Semoga laporan kerja praktek ini dapat memberikan manfaat dan dapat digunakan sebagaimana mestinya.

Surabaya, September 2016

Hanif Sudira

# DAFTAR ISI

[LEMBAR PENGESAHAN v](#_Toc437837585)

[Abstrak vii](#_Toc437837586)

[KATA PENGANTAR ix](#_Toc437837587)

[DAFTAR ISI xi](#_Toc437837588)

[DAFTAR GAMBAR xiii](#_Toc437837589)

[DAFTAR TABEL xv](#_Toc437837590)

[DAFTAR KODE SUMBER xvii](#_Toc437837591)

[1 BAB I PENDAHULUAN 19](#_Toc437837592)

[1.1. Latar Belakang 19](#_Toc437837593)

[1.1.1. Automated Testing using Protractor 19](#_Toc437837594)

[1.1.2. *Build Realtime Web Apps using Angular Meteor* 20](#_Toc437837595)

[1.2. Tujuan 20](#_Toc437837596)

[1.3. Manfaat 20](#_Toc437837597)

[1.4. Rumusan Permasalahan 20](#_Toc437837598)

[1.5. Lokasi dan Waktu Kerja Praktik 20](#_Toc437837599)

[1.6. Metodologi Kerja Praktik 21](#_Toc437837600)

[1.7. Sistematika Laporan 22](#_Toc437837601)

[2 BAB II PROFIL PERUSAHAAN 25](#_Toc437837602)

[2.1. Sejarah Perusahaan 25](#_Toc437837603)

[2.2. Visi dan Misi Perusahaan 26](#_Toc437837604)

[2.3. Struktur Organisasi 26](#_Toc437837605)

[2.4. Divisi Software Development Engineer 27](#_Toc437837606)

[3 BAB III TINJAUAN PUSTAKA 29](#_Toc437837607)

[3.1. UAT (User Acceptance Test) 29](#_Toc437837608)

[3.2. Angular JS 29](#_Toc437837609)

[3.3. Protractor JS 30](#_Toc437837610)

[3.4. Node JS 31](#_Toc437837611)

[3.5. Meteor JS 31](#_Toc437837612)

[3.6. Angular-Meteor 32](#_Toc437837613)

[3.7. MongoDB 32](#_Toc437837614)

[4 BAB IV ANALISIS DAN PERANCANGAN SISTEM 33](#_Toc437837615)

[4.1. Analisis Sistem 33](#_Toc437837616)

[4.1.1. Deskripsi Umum Sistem 33](#_Toc437837617)

[4.1.2. Usecase dan fitur sistem 34](#_Toc437837618)

[4.1.3. Diagram Aktifitas 36](#_Toc437837619)

[4.1.3. Perancangan Sistem 41](#_Toc437837620)

[4.1.4. Pengerjaan Sistem 41](#_Toc437837621)

[BAB V IMPLEMENTASI SISTEM 42](#_Toc437837622)

[4.2. Implementasi Antarmuka Pengguna 42](#_Toc437837623)

[4.2.1. Halaman Menulis Kode Program 42](#_Toc437837624)

[4.2.2. *See All Tests* 43](#_Toc437837625)

[4.2.3. *Submissions History* 44](#_Toc437837626)

[4.3. Implementasi Fungsi 45](#_Toc437837627)

[4.3.1. *Mengelola Akun* 45](#_Toc437837628)

[4.3.2. *Pembuatan Text Editor* 47](#_Toc437837629)

[4.3.3. *Organisasi File* 47](#_Toc437837630)

[4.3.4. *Mengkompilasi Kode Program Protractor* 48](#_Toc437837631)

[4.3.5. *Melihat Submissions History* 49](#_Toc437837632)

[5 BAB VI PENGUJIAN DAN EVALUASI 51](#_Toc437837633)

[5.1. Lingkungan Pengujian 51](#_Toc437837634)

[5.2. Skenario Pengujian 51](#_Toc437837635)

[5.3. Evaluasi Pengujian 52](#_Toc437837636)

[BAB VII KESIMPULAN DAN SARAN 55](#_Toc437837637)

[5.4. Kesimpulan 55](#_Toc437837638)

[5.5. Saran 55](#_Toc437837639)

[DAFTAR PUSTAKA 57](#_Toc437837640)

[LAMPIRAN 59](#_Toc437837641)

[Lampiran A. Kode program *Protractor* 59](#_Toc437837642)

[BIODATA PENULIS 67](#_Toc437837643)

# DAFTAR GAMBAR

[Gambar 1. Logo GDP Venture 25](#_Toc437836330)

[Gambar 2. Struktur Organisasi GDP Labs 26](#_Toc437836331)

[Gambar 3. Logo Angular JS 29](#_Toc437836332)

[Gambar 4. Logo Protractor JS 30](#_Toc437836333)

[Gambar 5. Logo Node JS 31](#_Toc437836334)

[Gambar 6. Logo Meteor JS 31](#_Toc437836335)

[Gambar 7. Use Case Diagram 34](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836336)

[Gambar 8. Diagram Aktifitas Menambah Akun 36](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836337)

[Gambar 9. Diagram Aktifitas Masuk ke Sistem 36](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836338)

[Gambar 10. Diagram Aktifitas Mengganti Password Akun 37](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836339)

[Gambar 11. Diagram Aktifitas Keluar dari Sistem 37](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836340)

[Gambar 12. Diagram Aktifitas Pembuatan *Text Editor* 38](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836341)

[Gambar 13. Diagram Aktifitas Menyimpan File 38](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836342)

[Gambar 14. Diagram Aktifitas Menghapus File 39](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836343)

[Gambar 15. Diagram Aktifitas Membuka File 39](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836344)

[Gambar 16. Diagram Aktifitas Mengkompilasi Kode Program 40](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836345)

[Gambar 17. Diagram Aktifitas Melihat Submissions History 40](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836346)

[Gambar 18 Arsitektur IDE-Protractor 41](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836347)

[Gambar 19. Sublime Text 3 41](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836348)

[Gambar 20. Menulis Kode Program 42](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836349)

[Gambar 21. Hasil Kompilasi Kode Program 43](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836350)

[Gambar 22. *See All Tests* 43](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836351)

[Gambar 23 Halaman indeks *change request* 43](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836352)

[Gambar 24. Submissions History 44](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836353)

[Gambar 25. Hasil Test yang Pernah Dibuat Sebelumnya 45](file:///D:\KULIAH\Kerja%20Praktik\Laporan%20KP.docx#_Toc437836354)

*[Halaman ini sengaja dikosongkan]*

# DAFTAR TABEL

[Tabel 1. Kebutuhan Fungsionalitas IDE-Protractor 35](#_Toc437835819)

[Tabel 2. Evaluasi Pengujian 53](#_Toc437835820)

*[Halaman ini sengaja dikosongkan]*

# DAFTAR KODE SUMBER

[Kode Sumber 1. Konfigurasi Protractor JS 59](#_Toc437836414)

[Kode Sumber 2. Spesifikasi Protractor JS 66](#_Toc437836415)

*[Halaman ini sengaja dikosongkan]*

# BAB I PENDAHULUAN

## Latar Belakang

Latar belakang ini dibagi menjadi dua bagian yakni Protractor-IDE : *Automated Testing using Protractor* dan *Build Realtime Web Apps using Angular Meteor*.

### Automated Testing using Protractor

Testing pada *software* merupakan hal yang wajib dilakukan guna menjamin kualitas *software*. Terlebih apabila *software* tersebut merupakan pesanan suatu klien, maka testing pada *software* mutlak dilakukan. Terdapat banyak jenis testing pada *software*. Salah satunya yaitu UAT (*User Acceptance Test*). UAT merupakan tes yang dilakukan guna menjaminnya *software* yang dibuat sesuai dengan kebutuhan user*.*

Terdapat *tools* untuk membuat UAT yang otomatis. Salah satunya yaitu Protractor. Adapun fitur yang terdapat pada Protractor diantaranya adalah sebagai berikut :

*Test Like a User*

Protractor merupakan salah satu *tools* untuk membuat UAT yang otomatis. Sehingga hasil tes seolah-olah *user* yang bertinteraksi dengan *software*.

*For AngularJS Apps*

Protractor mendukung aplikasi AngularJS. Dimana Protractor dapat mengenali elemen-elemen pada kode program AngularJS. Namun Protractor bisa juga digunakan untuk kode program selain AngularJS.

*Automatic Waiting*

Tidak perlu menambahkan fungsi *delay* apabila menggunakan Protractor. Karena Protractor akan menunggu otomatis sampai sebuah *software* selesai loading.

Adapun proyek yang dikerjakan pada kerja praktik ini yaitu membuat UAT pada aplikasi kalkulator menggunakan Protractor.

### *Build Realtime Web Apps using Angular Meteor*

MeteorJS merupakan salah satu *backend web* *framework* untuk bahasa pemrograman NodeJS. Sedangkan AngularJS merupakan salah *frontend web framework* untuk bahasa pemrograman JavaScript. Metode yang digunakan pada proyek kerja praktik ini yaitu penggabungan antara MeteorJS dan Angular JS yang selanjutnya akan saya sebut Angular-Meteor.

Adapun aplikasi yang dibuat yaitu pembuatan Protractor-IDE (*Integrated Development Environment*) berbasis web. Dimana sinkronisasi data pada aplikasi dibuat secara *realtime*.

## Tujuan

* Membuat UAT (*User Acceptance Test*) otomatis pada aplikasi kalkulator menggunakan Protractor.
* Membuat Protractor-IDE (*Integrated Development Environment*) menggunakan Angular-Meteor.

## Manfaat

Manfaat yang diperoleh antara lain :

* Mempermudah kegiatan UAT (*User Acceptance Test*).
* Mempermudah pengembangan program Protractor.

## Rumusan Permasalahan

* Bagaimana membuat UAT (*User Acceptance Test*) yang otomatis menggunakan Protractor?
* Bagaimana membuat Protractor-IDE (*Integrated Development Environment*) menggunakan Angular-Meteor?

## Lokasi dan Waktu Kerja Praktik

Lokasi kerja praktik berada di PT. Global Digital Prima (GDP) Venture. Dengan alamat Menara Citicon Lantai 17 Jl. Letjen S. Parman Kav. 72 Jakarta Barat.

Adapun tanggal kerja praktik dimulai pada tanggal 06 Juli 2015 hingga 04 September 2015 ( 7 minggu ). Masuk kerja dari hari senin hingga jum’at dari jam 08.00 – 17.00 ( durasi waktu 8 jam kerja ).

## Metodologi Kerja Praktik

1. Perumusan Masalah

Pada tahap perumusan masalah penulis menemukan titik pokok hal yang harus penulis kerjakan, untuk kerja praktik di PT. Global Digital Prima (GDP) Venture, yaitu “Bagaimana membuat UAT (*User Acceptance Test*) yang otomatis lalu menerapkannya dalam pembuatan IDE (*Integrated Development Environment*) menggunakan Angular-Meteor” sehingga pada tahap selanjutnya penulis harus menjawab pertanyaan tersebut.

1. Studi Literatur

Pada tahap studi literatur penulis melakukan pembelajaran terhadap UAT (*User Acceptance Test*) yang bernama *Protractor JS*. yang menggunakan bahasa pemrograman *Javascript*. Melakukan pembelajaran pembuatan aplikasi berbasis web menggunakan *Angular-Meteor* yang menggunakan Bahasa pemrograman *Javascript* dan *Node Js*. Hal ini dilakukan untuk menunjang pengetahuan dalam membangun pembuatan IDE (*Integrated Development Environment*) menggunakan *Angular-Meteor*.

1. Analisis Sistem

Pada tahap ini dilakukan analisis sistem, analisis dilakukan dengan cara mengobservasi proses UAT (*User Acceptance Test*) yang otomatis untuk diterapkan di aplikasi berbasis web menggunakan *Angular-Meteor*, kemudian merancang sistem yang akan dibangun berdasarkan hasil observasi.

1. Implementasi Sistem

Implementasi merupakan tahap membangun rancangan. Pada tahapan ini merealisasikan apa yang terdapat pada tahapan sebelumnya, sehingga sesuai dengan apa yang telah direncanakan.

1. Pengujian dan Evaluasi

Pada tahapan ini dilakukan uji coba pada aplikasi yang telah diimplementasikan. Tahapan ini bermaksud untuk mengevaluasi kesesuaian sistem dan aplikasi yang dibuat apakah dapat dilakukan dengan lancar atau tidak. Selain itu juga untuk mencari masalah yang mungkin timbul dan tidak lupa mengadakan perbaikan jika terdapat kesalahan

1. Kesimpulan dan Saran

Sistem infomasi yang dibangun yang bernama “Protractor-IDE : Automated Testing using Protractor & Build Realtime Web Apps using Angular Meteor” ini telah memenuhi syarat yang diinginkan, telah berjalan dengan baik dan lancar, dan dapat membantu proses otomatisasi pada UAT (*User Acceptance Test*).

## Sistematika Laporan

Laporan kerja praktik ini terbagi menjadi 7 bab dengan rincian sebagai berikut :

* **BAB 1 : PENDAHULUAN**

Berisi pendahuluan yang terdiri dari latar belakang, tujuan, manfaat, rumusan permasalahan, lokasi dan waktu kerja praktik, metodologi kerja praktik, dan sistematika penulisan.

* **BAB 2 : PROFIL PERUSAHAAN**

Berisi tentang gambaran umum dari tempat kerja praktik yaitu PT. Global Digital Prima (GDP) Venture, mulai dari sejarah perusahaan, visi dan misi perusahaan, struktur organisasi, dan divisi tempat kerja praktik dilakukan.

* **BAB 3 : TINJAUAN PUSTAKA**

Bab ini berisi dasar teori dari metode/teknologi yang digunakan dalam meyelesaikan proyek kerja praktik.

* **BAB 4 : ANALISIS SISTEM**

Pada bab ini dijelaskan mengenai desain antarmuka aplikasi serta yang digunakan dalam aplikasi.

* **BAB 5 : IMPLEMENTASI SISTEM**

Pada bab ini berisi uraian tahap-tahap yang dilakukan untuk proses implementasi aplikasi

* **BAB 6 : UJI COBA DAN EVALUASI**

Pada bab ini, dijelaskan mengenai hasil uji coba dan evaluasi dari perangkat lunak yang telah dikembangkan selama pelaksanaan kerja praktik.

* **BAB 7 : KESIMPULAN DAN SARAN**

Bab ini berisi kesimpulan dan saran dari proses pelaksanaan kerja praktik.

*[Halaman ini sengaja dikosongkan]*

# BAB II PROFIL PERUSAHAAN

## Sejarah Perusahaan



Gambar . Logo Tokopedia

Tokopedia merupakan salah satu mal online di Indonesia yang mengusung model bisnis *marketplace* dan mall online. Tokopedia memungkinkan setiap Individu, toko kecil untuk membuka dan mengelola toko online atau yang disebut *merchant* secara gratis.

Tokopedia telah berdiri kurang lebih 5 tahun sejak tahun 2009 tepatnya tokopedia pertama kali diluncurkan ke publik pada 17 Agustus 2009. GDP Venture tertarik untuk berinvestasi pada beberapa perusahaan yang menonjolkan sisi Indonesia seperti Kaskus dan MPI (Merah Putih Incubator). GDP Venture dipimpin generasi ketiga Grup Djarum, Martin Basuki Hartono, selaku CEO GDP Venture.

GDP Labs merupakan divisi Teknologi Informasi GDP Venture, yang dibentuk apa pada tahun 2012. GDP Labs bertujuan untuk membantu portfolio dan sister company GDP Venture dalam bidang Teknologi Informasi, serta bertindak sebagai inkubator startup yang dipandang potensial. Kantor GDP Labs terletak di Menara Citicon Lantai 17, Jl. Letjen S. Parman Kav. 72, Jakarta Barat. GDP Labs dipimpin oleh On Lee, selaku CTO GDP Labs.

## Visi dan Misi Perusahaan

Visi:

“*Membangun Indonesia yang Lebih Baik Lewat Internet*”.

Misi:

1. “*Value effective collaboration, humility, and embracing change and failure in our pursuit of success*”.

2. “*Focus also on fostering a constant learning environment*”.

## Struktur Organisasi



**Gambar 2**. **Struktur Organisasi GDP Labs**

Di GDP Labs terdapat beberapa unit kerja yang terbagi menjadi 4, yaitu :

1. Software Development Engineer, berhubungan dengan pemrograman dan pengembangan aplikasi, baik desktop, web maupun mobile.

2. Graphic Designer, berhubungan dengan perancangan grafis.

3. Product Manager, berhubungan dengan definisi produk, pembagian kerja dan pengawasan pelaksanaan pengembangan aplikasi.

4. System Engineer, berhubungan dengan infrastruktur, konfigurasi dan instalasi perangkat lunak atau kebutuhan-kebutuhan lain terkait efisiensi pelaksanaan kerja Software Development Engineer.

## Divisi Software Development Engineer

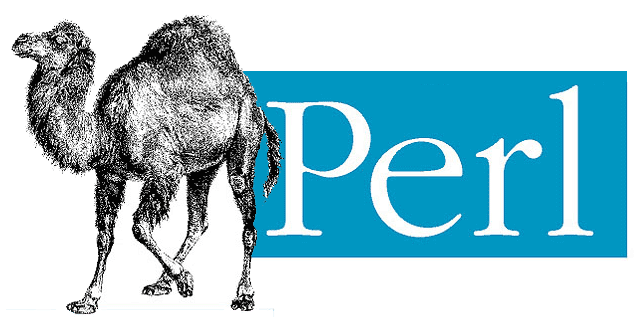
Divisi Software Development Engineer merupakan divisi yang menangani pemrograman dan pengembangan aplikasi baik itu desktop, web maupun mobile.

*[Halaman ini sengaja dikosongkan]*

# BAB III TINJAUAN PUSTAKA

Pada bab ini, dijelaskan mengenai dasar teori yang digunakan selama proses perbaikan bug dan pengembanga Zendesk API.

## Perl Programming Language

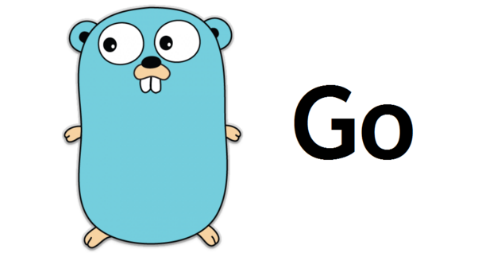


Gambar 2. Logo Perl Programming Language

*Perl* adalah singakatan dari Practical Extraction and Report Language. Bahasa pemograman ini diciptakan oleh Larry Wall pada tahun 1986 yang awalnya digunakan sebagai perangkat lunak yang digunakan untuk menulis program di lingkungan UNIX.

*Perl* merupakan Bahasa pemograman yang menggabungkan kemampuan Bahasa C, utilitas awk , sed , grep dam fitur shell oleh karena itu *Perl* tergolong sebagai interpreter walaupun bukan interpreter murni. *Perl* membaca kode sumber, kemudian akan melakukan kompilasi ke memori baru kemudian dijalankan.

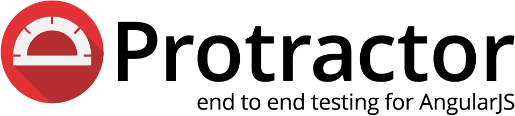
## Go Programming Language



Gambar . Logo Go Programming Language

*Go Language* atau yang biasa disebut *Golang.* Bahasa pemograman open source ini dibuat di google pada tahun 2007 oleh Robert Griesemer, Rob Pike dan Ken Thompson. *Golang* adalah Bahasa pemrograman yang mudah, sederhana, handal, efisien.

## Protractor JS



Gambar . Logo Protractor JS

*Protractor JS* merupakan kerangka kerja UAT (*User Acceptance Test*) otomatis untuk aplikasi yang menggunakan *Angular JS*. *Protractor JS* berjalan pada *web browser* sehingga dapat membuat UAT sesuai apa yang diinginkan [4].

*Protractor JS* memiliki beberapa fitur, diantaranya yaitu :

* Dapat melakukan tes layaknya *user*

*Protractor* dibangun dengan menggunakan lapisan atas *WebDriver.js*. Yaitu menggunakan *event* yang dapat beinteraksi dengan elemen-elemen pada suatu aplikasi. Sehingga dapat melakukan UAT layaknya *user* yang melakukan itu.

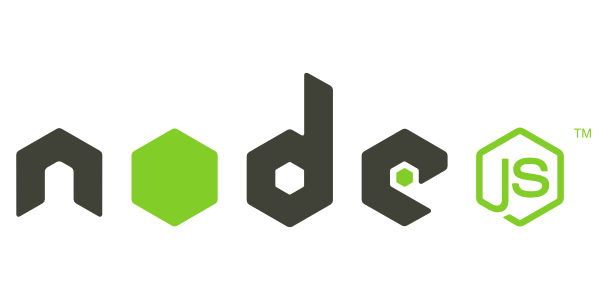
* Untuk aplikasi *Angular JS*

*Protractor* mendukung untuk menjangkau elemen-elemen pada Angular JS tanpa perlu melakukan pengaturan terlebih dahulu.

* Secara otomatis menunggu

Kita tidak perlu menambahkan *script* pada kode *Angular JS* untuk menunggu suatu *event*. *Protractor JS* secara otomatis dapat mengeksekusi langkah selanjutnya pada suatu tes sampai aplikasi menyelesaikan tugas yang tertunda.

## Node JS



Gambar . Logo Node JS

*Node JS* merupakan bahasa pemrograman yang dibangun menggunakan *Javascript*. *Node JS* dapat berjalan di berbagai sistem operasi, diantaranya *OS* *X*, *Microsoft* *Windows*, *Linux*, *FreeBSD*, *NonStop*, *IBM AIX*, *IBM System z*, dan *IBM I* [5].

*Node JS* dapat digunakan untuk membuat server web dan aplikasi jaringan. Beberapa kerangka kerja Node JS dapat memudahkan pembuatan aplikasi, diantaranya yaitu *Express JS*, *Socket IO*, *Meteor JS*, dll.

## Meteor JS



Gambar . Logo Meteor JS

*Meteor JS* merupakan suatu kerangka kerja yang menggunakan bahasa pemrograman *Node JS*. Meteor JS mendukung dalam pembuatan pembuatan aplikasi antar *platform* (Web, Android, IOS) dengan cepat [6].

Ada beberapa fitur yang merupakan keunggulan dari *Meteor JS*, diantaranya yaitu :

* *Javascript* Universal

Kode program yang sama dapan berjalan dari baik dari klien maupun server. Kode program yang sama juga, dapat berjalan pada browser dan perangkat *mobile* yang berbeda.

* *Mobile* dan Web

Mudah mengintegrasikan aplikasi dalam berbagai platform. Baik membangun aplikasi web maupun *mobile*.

* Kode program yang sederhana

Membuat program 10 baris di Meteor JS bisa mencapai 1000 baris di kerangka kerja yang lainnya. Sehingga hanya sedikit program di Meteor JS mampu membuat aplikasi yang luar biasa.

## Angular-Meteor

*Angular-Meteor* merupakan kombinasi dari kerangka kerja terbaik, yaitu *Angular JS* dan *Meteor JS*. *Angular-Meteor* merupakan suatu paket yang tersedia di NPM Node JS [7]. Sehingga dapat mudah untuk menggunakan *Angular-Meteor* ini.

## MongoDB

*MongoDB* pertama kali dikembangkan pada bulan Oktober 2007. *MongoDB* merupakan sistem basis data berorientasi dokumen lintas platform yang dikategorikan sebagai basis data “NoSQL”. *MongoDB* menghindari sistem basis data relasional yang berbasis table. Selain itu *MongoDB* mendukung JSON sehingga dapat membuat integrasi data dalam beberapa jenis aplikasi menjadi lebih mudah dan cepat. Sehingga saat ini MongoDB dinobatkan sebagai sistem basis data “NoSQL” yang paling popular [8].

# BAB IV ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini akan dijelaskan mengenai analisis dan perancangan sistem pengembangan Protractor-IDE.

## Analisis Sistem

### Deskripsi Umum Sistem

Sistem yang dibuat oleh penulis bernama Protractor-IDE. Protractor-IDE merupakan aplikasi berbasis web yang dikembangkan dengan bahasa pemograman *Node JS* dengan menggunakan *framework Meteor JS dan Angular JS*. Aplikasi ini bertujuan untuk memudahkan pengguna dalam membantu proses kompilasi program dan mengorganisasi file yang menggunakan Protractor.

Fitur pertama IDE-Protractor yaitu proses pengelolaan akun aplikasi. Diantaranya yaitu pengguna aplikasi dapat dengan mudah membuat akun, mengedit kata sandi, masuk kedalam aplikasi, dan dapat keluar aplikasi.

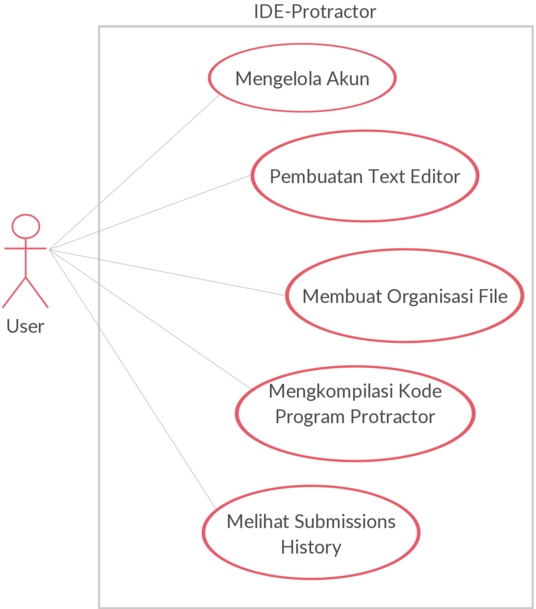
Fitur kedua IDE-Protractor adalah adanya text-editor program Protractor, yaitu dapat menulis kode program konfigurasi maupun kode program spesifikasi. Selain itu adanya koreksi apabila kode program yang ditulis salah. Sehingga pengguna dapat dengan mudah mengoreksi kode program yang ditulis dalam aplikasi ini.

Fitur ketiga IDE-Protractor adalah organisasi file. Sehingga pengguna dapat dengan mudah mengorganisasi file Protractor*.* Diantaranya yaitu dapat menyimpan file, menghapus file, serta membuka file yang sebelumnya pernah disimpan.

Fitur keempat IDE-Protractor adalah kompilasi kode program Protractor. Yaitu menjalankan program Protractor yang pernah ditulis oleh pengguna lalu menampilkan hasilnya. Sehingga pengguna tidak perlu repot untuk menjalankan aplikasi lewat *command promt* pada sistem operasi *windows* atau *terminal* pada sistem operasi *linux*.

Fitur kelima IDE-Protractor adalah melihat *submissions history*. Yaitu melihat hasil program yang pernah dijalankan sebelumnya. Pada fitur ini terdapat indikasi program yang lolos tes dan tidak lolos tes. Sehingga pengguna dapat dengan mudah melihat tes mana saja yang pernah lolos dan tes mana saja yang pernah tidak lolos uji.

### Usecase dan fitur sistem

**Hanya terdapat 1 peran dalam Protractor-IDE, yaitupengguna*.* Dalam Protractor-IDE peran ini dapat memiliki semua akses terhadap fitur yang telah disebutkan sebelumnya. Diantaranya yaitu mengelola akun, menggunakan *text-editor*, melakukan organisasi file, mengkompilasi program, dan melihat submissions history.

Gambar . Use Case Diagram

|  |  |  |
| --- | --- | --- |
| **No Kebutuhan** | **Kebutuhan** | **Deskripsi Kebutuhan** |
| F-001 | Mengelola Akun | Pengguna aplikasi dapat dengan membuat akun, mengedit kata sandi, masuk kedalam aplikasi, dan dapat keluar aplikasi. |
| F-002 | Pembuatan *Text Editor* | Pengguna dapat menulis kode program konfigurasi maupun kode program spesifikasi. Selain itu adanya koreksi apabila kode program yang ditulis salah. |
| F-003 | Membuat Organisasi File | Pengguna dapat menyimpan file, menghapus file, serta membuka file yang sebelumnya pernah disimpan. |
| F-004 | Mengkompilasi Kode Program *Protractor* | Menjalankan program *Protractor* yang pernah ditulis oleh pengguna lalu menampilkan hasilnya. |
| F-005 | Melihat *Submissions History* | Melihat hasil program yang pernah dijalankan sebelumnya. Pada fitur ini terdapat indikasi program yang lolos tes dan tidak lolos tes. |

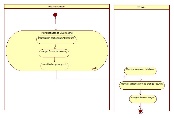
Tabel . Kebutuhan Fungsionalitas IDE-Protractor

### Diagram Aktifitas

#### Mengelola Akun

#### Menambah Akun

Gambar . Diagram Aktifitas Menambah Akun



#### Masuk (*sign in*) ke Sistem

Gambar . Diagram Aktifitas Masuk ke Sistem

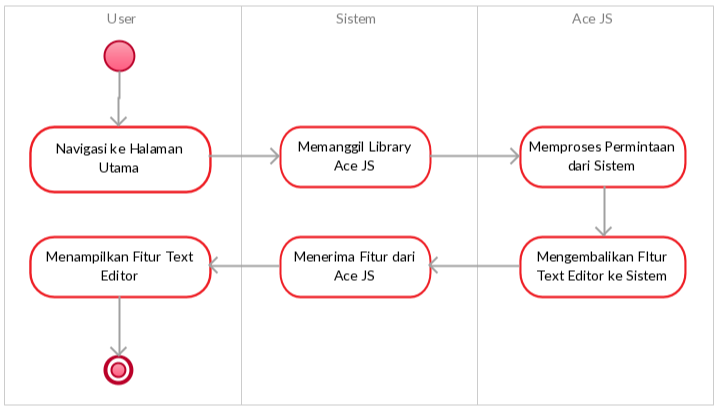
#### Mengganti Password Akun

Gambar . Diagram Aktifitas Mengganti Password Akun

#### Keluar (sign out) Dari Sistem

Gambar . Diagram Aktifitas Keluar dari Sistem

#### Pembuatan *Text Editor*



Gambar . Diagram Aktifitas Pembuatan *Text Editor*

#### Membuat Organisasi File

#### Menyimpan File

Gambar . Diagram Aktifitas Menyimpan File

#### Menghapus File

Gambar . Diagram Aktifitas Menghapus File

#### Membuka File

Gambar . Diagram Aktifitas Membuka File

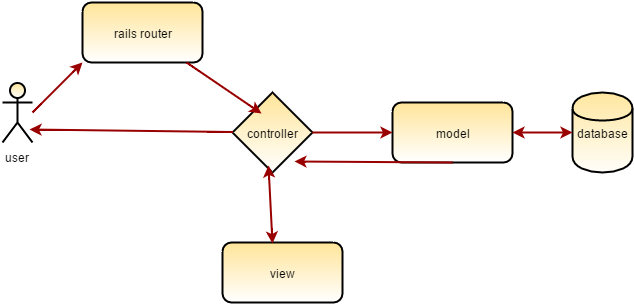
#### Mengkompilasi Kode Program

Gambar . Diagram Aktifitas Mengkompilasi Kode Program

#### Melihat *Submissions History*

Gambar . Diagram Aktifitas Melihat Submissions History

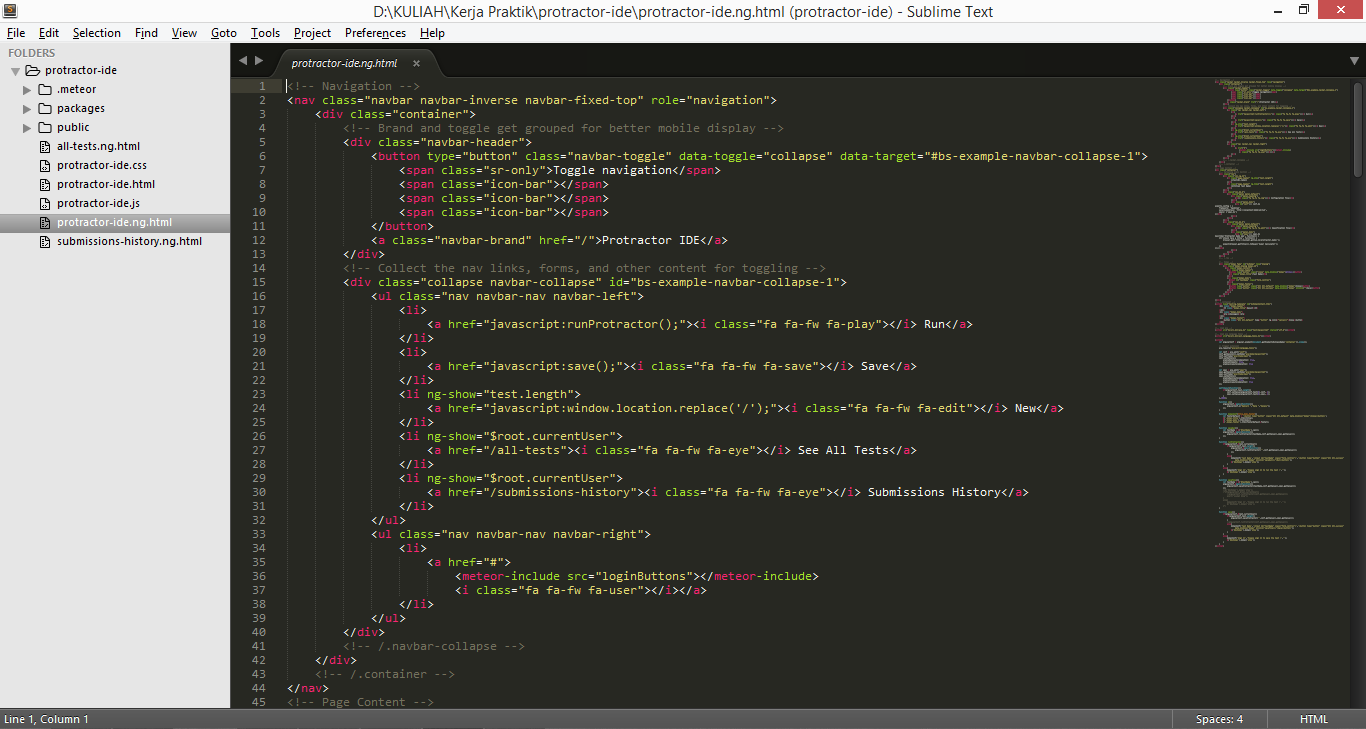
### Perancangan Sistem

 Sistem yang dibangun berbasis web, dengan arsitektur MVC (*Model View Controller*).

Meteor JS

Gambar Arsitektur IDE-Protractor

### Pengerjaan Sistem

 Pengerjaan sistem dilakukan menggunakan *text-editor* Sublime Text 3.

Gambar . Sublime Text 3

# BAB V IMPLEMENTASI SISTEM

Bab ini membahas tentang implementasi dari perancangan sistem.

## Implementasi Antarmuka Pengguna

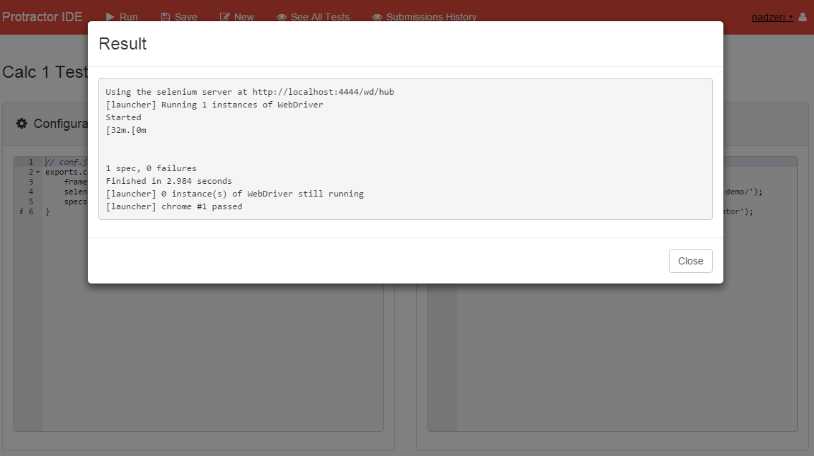
Implementasi antarmuka pengguna untuk Protractor-IDE secara garis besar terdapat tiga yaitu, menulis kode program*, See All Tests,* dan *Submissions History*.

### Halaman Menulis Kode Program

Gambar . Menulis Kode Program

Pada laman menulis kode program dibagi menjadi dua bagian. Yaitu menulis kode program untuk file konfigurasi dan menulis kode program untuk file spesifikasi. File konfigurasi berfungsi untuk menulis konfigurasi pada Protractor. Sedangkan file spesifikasi berfungsi untuk menulis kode testing pada aplikasi *Angular JS*.

Selain itu pada halaman ini terdapat fitur *syntax highlighting* pada *text-editor*. Sehingga apabila ada tulisan yang disisipkan selain kode program Protractor, maka terdapat pemberitahuan *error*. Selain itu juga terdapat fitur *autocomplete* sehingga memudahkan pengguna dalam menulis kode program Protractor.

Apabila kode program telah selesai dibuat, *user* dapat menekan tombol “Run” di bagian navigasi. Tujuannya yaitu menjalankan kode program yang telah dibuat. Setelah itu komputer akan menjalankan *browser* web yang bertujuan untuk menjalankan kode program Protractor tersebut. Setelah semua program dijalankan aplikasi ini akan menampilkan hasil dari program yang telah dijalankan, seperti yang terlihat pada gambar 21.

Gambar . Hasil Kompilasi Kode Program

### *See All Tests*

Gambar . *See All Tests*

Gambar Halaman indeks *change request*

Halaman ini bertujuan untuk menampilkan file tes yang pernah dibuat sebelumnya. Adapun yang ditampilkan pada laman ini yaitu ID file tes, nama file, dibuat pada, terakhir *update* pada, lalu pilihan untuk menghapus file.

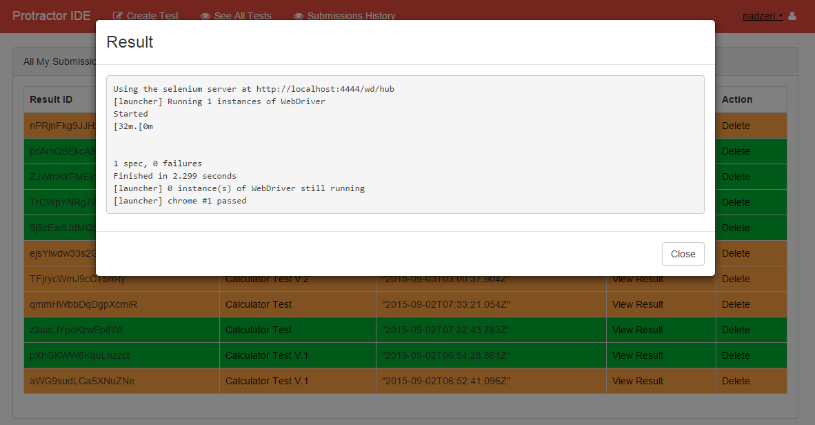
Adapun daftar file yang ada di tabel tersebut disortir berdasarkan yang terakhir di-*update*. Sehingga dapat memudahkan pengguna untuk mencari file yang terakhir di-*update*. Apabila ingin membuka file yang pernah dibuat, klik nama file yang bersangkutan. Setelah itu file yang dimaksud akan ditampilkan seperti yang terlihat di gambar 20.

### *Submissions History*

Gambar . Submissions History

Pada laman menulis *submissions history* terdapat dua kategori yang berbeda. Yaitu baris tabel yang berwarna oranye dan baris tabel yang berwarna hijau. Warna oranye maksudnya tes tersebut gagal sedangkan warna hijau maksudnya tes tersebut berhasil. Adapun maksud dari hal tersebut yaitu untuk memudahkan *user* dalam melihat *submissions history* mana yang sukses dan *submissions history* mana yang gagal.

Selain itu pada halaman ini terdapat beberapa atribut tabel yang ditampilkan. Diantaranya yaitu *Result ID*, Nama test, dibuat pada, lihat hasil, dan pilihan aksi hapus. Adapun daftar file yang ada di tabel tersebut disortir berdasarkan yang terakhir dikompilasi. Sehingga dapat memudahkan pengguna untuk melihat daftar test terakhir yang dikompilasi tersebut.

Selain itu apabila ingin melihat menghapus tes yang pernah dikompilasi sebelumnya, klik *delete* sesuai test yang ingin dihapus. Apabila ingin melihat hasil tes yang pernah dikompilasi sebelumnya, klik *view result* sesuai test yang ingin dilihat. Setelah itu hasil yang dimaksud akan ditampilkan seperti yang terlihat di gambar 25.

Gambar . Hasil Test yang Pernah Dibuat Sebelumnya

## Implementasi Fungsi

### *Mengelola Akun*

#### *Fungsi Menambah Akun*

Meteor.users.insert({

username: username,

password: password

});

Fungsi pada kerangka kerja *Meteor JS* untuk menambah akun pengguna.

#### *Mengedit Kata Sandi*

Meteor.users.update(

{'id':id},

{$set{'password': password}}

);

Fungsi pada kerangka kerja *Meteor JS* untuk mengedit password pengguna.

#### *Masuk Kedalam Aplikasi*

Template.login.events({

'submit form': function(event){

event.preventDefault();

var username = $('[name=username]').val();

var password = $('[name=password]').val();

Meteor.loginWithPassword(username, password);

}

});

Fungsi pada kerangka kerja *Meteor JS* untuk masuk kedalam aplikasi.

#### *Keluar Aplikasi*

Template.navigation.events({

'click .logout: function(event){

event.preventDefault();

Meteor.logout();

}

});

Fungsi pada kerangka kerja *Meteor JS* untuk masuk kedalam aplikasi.

### *Pembuatan Text Editor*

#### *Text-Editor File Konfigurasi*

var conf = ace.edit("conf");

conf.getSession().setMode("ace/mode/javascript");

conf.setTheme("ace/theme/dawn");

conf.setOptions({

enableBasicAutocompletion: true,

enableSnippets: true,

enableLiveAutocompletion: true

});

Fungsi pada kerangka kerja *Angular JS* untuk membuat *text-editor* file konfigurasi *Protractor* menggunakan *Ace JS*.

#### *Text-Editor File Spesifikasi*

var spec = ace.edit("spec");

spec.getSession().setMode("ace/mode/javascript");

spec.setTheme("ace/theme/dawn");

spec.setOptions({

enableBasicAutocompletion: true,

enableSnippets: true,

enableLiveAutocompletion: true

});

Fungsi pada kerangka kerja *Angular JS* untuk membuat *text-editor* file konfigurasi *Protractor* menggunakan *Ace JS*.

### *Organisasi File*

#### *Menyimpan File*

'saveFile' : function(testName,conf,spec){

if (! Meteor.userId()) {

throw new Meteor.Error('not-authorized');

}

var testId = Tests.insert({

name: testName,

conf: conf,

spec: spec,

createdAt: new Date(),

lastUpdate: new Date(),

status: "ok",

userId: Meteor.userId()

},

function(err,msg){

Meteor.call("saveFile",msg,conf,spec);

});

return (testId);

},

Fungsi pada kerangka kerja *Meteor JS* untuk menyimpan menyimpan file.

#### *Membuka File*

$scope.test = $meteor.collection(function() {

return Tests.find(

{\_id:$stateParams.testId}

);

});

Fungsi pada kerangka kerja *Meteor JS* untuk membuka file yang telah disimpan. Lalu menampilkannya *text-editor* file konfigurasi *Protractor* menggunakan *Ace JS*.

#### *Menghapus File*

'deleteTest' : function(testId){

return Tests.update(testId, {

$set: {status: "remove"}

});

},

Fungsi pada kerangka kerja *Meteor JS* untuk menghapus file yang telah disimpan.

### *Mengkompilasi Kode Program Protractor*

'runProtractor': function(testId){

if (! Meteor.userId()) {

throw new Meteor.Error('not-authorized');

}

var commmand = "protractor \"D:/KULIAH/Kerja Praktik/protractor-ide/public/test-file/" + Meteor.userId() + "/" +testId + "/conf.js\"";

return Meteor.call("executeCommand",commmand);

}

Fungsi pada kerangka kerja *Meteor JS* untuk menjalankan kode program Protractor JS.

### *Melihat Submissions History*

$scope.results = $meteor.collection(function() {

return Results.find({});

});

Fungsi pada kerangka kerja *Meteor JS* untuk mengambil seluruh data test.

*[Halaman ini sengaja dikosongkan]*

# BAB VI PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada aplikasi yang dikembangkan. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsionalitas sistem dan pengujian kegunaan sistem atau pengujian usabilitas. Pengujian fungsionalitas mengacu pada ketercapaian setiap implementasi Pengujian kegunaan sistem dilakukan dengan mengetahui tanggapan dari pengguna terhadap kegunaan sistem. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

## Lingkungan Pengujian

Lingkungan pengujian sistem pada pengerjaan kerja praktik ini dilakukan pada lingkungan dan alat kakas sebagai berikut:

*Processor* : Intel® Core™ i5-4200U CPU

(1.6 GHz 2.3 GHz)

*Memory* : 4.00 GB

Jenis *Device* : Personal Computer

Sistem Operasi : Microsoft Windows 8.1 Industry Pro

*Browser* : Google Chrome

## Skenario Pengujian

Skenario pengujian berdasarkan scenario yang telah dideskripsikan pada tahap testing, dapat dilihat di lampiran kode sumber testing. Secara umum scenario pengujian sebagai berikut:

1. Mengelola Akun Aplikasi

Pengguna aplikasi dapat dengan mudah membuat akun, mengedit kata sandi, masuk kedalam aplikasi, dan dapat keluar aplikasi.

1. Pembuatan *Text-Editor*

Pengguna dapat menulis kode program konfigurasi maupun kode program spesifikasi. Selain itu adanya koreksi apabila kode program yang ditulis salah. Sehingga pengguna dapat dengan mudah mengoreksi kode program yang ditulis dalam aplikasi ini.

1. Membuat Organisasi File

Pengguna dapat dengan mudah mengorganisasi file Protractor*.* Diantaranya yaitu dapat menyimpan file, menghapus file, serta membuka file yang sebelumnya pernah disimpan*.*

1. Mengkompilasi Kode Program *Protractor*

Pengguna dapat menjalankan program Protractor yang pernah ditulis oleh pengguna lalu menampilkan hasilnya. Sehingga pengguna tidak perlu repot untuk menjalankan aplikasi lewat *command promt* pada sistem operasi *windows* atau *terminal* pada sistem operasi *linux.*

1. Melihat *Submissions History*

Pengguna dapat melihat hasil program yang pernah dijalankan sebelumnya. Pada fitur ini terdapat indikasi program yang lolos tes dan tidak lolos tes. Sehingga pengguna dapat dengan mudah melihat tes mana saja yang pernah lolos dan tes mana saja yang pernah tidak lolos uji.

## Evaluasi Pengujian

|  |  |  |  |
| --- | --- | --- | --- |
| **No Kebutuhan** | **Kebutuhan** | **Deskripsi Kebutuhan** | **Status** |
| F-001 | Mengelola Akun | Pengguna aplikasi dapat dengan membuat akun, mengedit kata sandi, masuk kedalam aplikasi, dan dapat keluar aplikasi. | Berhasil. |
| F-002 | Pembuatan *Text Editor* | Pengguna dapat menulis kode program konfigurasi maupun kode program spesifikasi. Selain itu adanya koreksi apabila kode program yang ditulis salah. | Berhasil. |
| F-003 | Membuat Organisasi File | Pengguna dapat menyimpan file, menghapus file, serta membuka file yang sebelumnya pernah disimpan. | Berhasil. |
| F-004 | Mengkompilasi Kode Program *Protractor* | Menjalankan program *Protractor* yang pernah ditulis oleh pengguna lalu menampilkan hasilnya. | Berhasil. |
| F-005 | Melihat *Submissions History* | Melihat hasil program yang pernah dijalankan sebelumnya. Pada fitur ini terdapat indikasi program yang lolos tes dan tidak lolos tes. | Berhasil. |

Tabel . Evaluasi Pengujian

Berdasarkan hasil pengujian baik melaui *script* maupun secara langsung semua fungsionalitas dapat berjalan dengan lancar sesuai dengan perencanaan.

*[Halaman ini sengaja dikosongkan]*

# BAB VII KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan kerja praktik serta saran-saran tentang pengembangan yang dapat dilakukan terhadap kerja praktik ini di masa yang akan datang.

## Kesimpulan

Kesimpulannya, semua kebutuhan fungsional yang ada pada IDE-Protractor telah terpenuhi sepertimengelola akun, pembuatan *text-editor*, membuat organisasi file, mengkompilasi kode program *Protractor*, dan melihat *submissions history*. IDE-Protractor dapat memudahkan pengguna dalam membantu proses kompilasi program dan mengorganisasi kode program yang menggunakan Protractor.

## Saran

Saran untuk pengembangan serta perbaikan aplikasi pada kerja praktik kali ini :

Pembuatan *user interface* yang lebih *friendly.*

Penambahan cakupan *testing.*

Penambahan *case testing.*

Pembuatan aplikasi secara *online* sehingga dapat dipakai oleh pengguna secara luas.

Pembuatan dokumentasi cara penggunaan aplikasi ini.

*[Halaman ini sengaja dikosongkan]*

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | Cimperman, Rob. “*UAT Defined: A Guide to Practical User Acceptance Testing*”. Pearson Education, pp. Chapter 2, 2006. |
| [2] | Goethem, Brian Hambling, Pauline Van, “*User acceptance testing : a step-by-step guide*,” BCS Learning & Development Limited, 2013. |
| [3] | “*What is Angular ?,*” [Online]. Available: https://docs.angularjs.org/guide/introduction. [Accessed 13 Desember 2015]. |
| [4] | “*Protractor, end to end testing for Angular JS ?,*” [Online]. Available: http://angular.github.io/protractor/#/. [Accessed 13 Desember 2015]. |
| [5] | Pedro Teixeira, “*Professional Node.js: Building JavaScript Based Scalable Software*,” John Wiley & Sons Inc, 2012. |
| [6] | “*Full stack JavaScript for amazing apps: Why Meteor?,*” [Online]. Available: https://www.meteor.com/why-meteor/features. [Accessed 13 Desember 2015]. |
| [7] | “*Angular-Meteor Manifesto,*” [Online]. Available: http://www.angular-meteor.com/manifesto. [Accessed 13 Desember 2015]. |
| [8] | “*DB-Engines Ranking,*” [Online]. Available: http://db-engines.com/en/ranking. [Accessed 13 Desember 2015]. |

*[Halaman ini sengaja dikosongkan]*

# LAMPIRAN

1. Kode program *Protractor*

// protractor configuration for user acceptance test (UAT)

exports.config = {

seleniumAddress: 'http://127.0.0.1:4444/wd/hub',

capabilities: { 'browserName': 'chrome'},

specs: [ 'spec-angular.js' ],

onPrepare: function() {

browser.driver.manage().window().maximize();

global.isAngularSite = function(flag) {

browser.ignoreSynchronization = !flag;

};

}

}

Kode Sumber . Konfigurasi Protractor JS

var Calculator = function(){

this.OPERATOR = {'+':0,

'/':1,

'%':2,

'\*':3,

'-':4};

this.first = element(by.model("first"));

this.operator = element.all(by.options("value for (key, value) in operators"));

this.second = element(by.model("second"));

this.execute = element(by.id("gobutton"));

this.result = element(by.binding("latest"));

this.history = element.all(by.repeater("result in memory"));

this.get = function(){

browser.get('http://juliemr.github.io/protractor-demo');

};

this.setFirst = function(data){

this.first.sendKeys(data);

};

this.setOperator = function(data){

this.operator.get(this.OPERATOR[data]).click();

};

this.setSecond = function(data){

this.second.sendKeys(data);

};

this.doExecute = function(){

this.execute.click();

};

this.getResult = function(){

return this.result.getText();

};

};

describe("Functionality Calculator Test Cases", function() {

var calculator = new Calculator();

beforeEach(function() {

isAngularSite(true); // isAngularSite(true), if it's an Angular app!

});

it("1. Check the addition of two positive integer numbers", function() {

calculator.get();

calculator.setFirst('13');

calculator.setOperator('+');

calculator.setSecond('19');

calculator.doExecute();

expect(calculator.getResult()).toEqual('32');

});

it("2. Check the addition of two positive decimal numbers", function() {

calculator.get();

clickalculator.setFirst('13.3');

calculator.setOperator('+');

calculator.setSecond('19.4');

calculator.doExecute();

expect(calculator.getResult()).toEqual('32.7');

});

it("3. Check the addition of two negative integer numbers", function() {

calculator.get();

calculator.setFirst('-18');

calculator.setOperator('+');

calculator.setSecond('-1');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-19');

});

it("4. Check the addition of two negative decimal numbers", function() {

calculator.get();

calculator.setFirst('-18.4');

calculator.setOperator('+');

calculator.setSecond('-1.6');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-20');

});

it("5. Check the addition of one positive integer and one negative decimal number", function() {

calculator.get();

calculator.setFirst('18');

calculator.setOperator('+');

calculator.setSecond('-8.3');

calculator.doExecute();

expect(calculator.getResult()).toEqual('9.7');

});

it("6. Check the addition of one positive decimal and one negative integer number", function() {

calculator.get();

calculator.setFirst('18.3');

calculator.setOperator('+');

calculator.setSecond('-8');

calculator.doExecute();

expect(calculator.getResult()).toEqual('10.3');

});

it("7. Check the subtraction of two positive integer numbers", function() {

calculator.get();

calculator.setFirst('18');

calculator.setOperator('-');

calculator.setSecond('21');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-3');

});

it("8. Check the subtraction of two positive decimal numbers", function() {

calculator.get();

calculator.setFirst('18.7');

calculator.setOperator('-');

calculator.setSecond('21.4');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-2.7');

});

it("9. Check the subtraction of two negative integer numbers", function() {

calculator.get();

calculator.setFirst('-18');

calculator.setOperator('-');

calculator.setSecond('-12');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-6');

});

it("10. Check the subtraction of two negative decimal numbers", function() {

calculator.get();

calculator.setFirst('-18.5');

calculator.setOperator('-');

calculator.setSecond('-12.3');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-6.2');

});

it("11. Check the subtraction of one negative integer and one positive decimal number", function() {

calculator.get();

calculator.setFirst('-18');

calculator.setOperator('-');

calculator.setSecond('12.7');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-30.7');

});

it("12. Check the multiplication of two positive integer numbers", function() {

calculator.get();

calculator.setFirst('18');

calculator.setOperator('\*');

calculator.setSecond('5');

calculator.doExecute();

expect(calculator.getResult()).toEqual('90');

});

it("13. Check the multiplication of two negative integer numbers", function() {

calculator.get();

calculator.setFirst('-18');

calculator.setOperator('\*');

calculator.setSecond('-5');

calculator.doExecute();

expect(calculator.getResult()).toEqual('90');

});

it("14.Check the multiplication of one negative decimal and one positive integer number", function() {

calculator.get();

calculator.setFirst('-18.9');

calculator.setOperator('\*');

calculator.setSecond('5');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-94.5');

});

it("15. Check the division of two positive integer numbers", function() {

calculator.get();

calculator.setFirst('18');

calculator.setOperator('/');

calculator.setSecond('5');

calculator.doExecute();

expect(calculator.getResult()).toEqual('3.6');

});

it("16. Check the division of two negative integer numbers", function() {

calculator.get();

calculator.setFirst('-18');

calculator.setOperator('/');

calculator.setSecond('-5');

calculator.doExecute();

expect(calculator.getResult()).toEqual('3.6');

});

it("17. Check the division of one positive decimal number and one integer negative number", function() {

calculator.get();

calculator.setFirst('20.5');

calculator.setOperator('/');

calculator.setSecond('-5');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-4.1');

});

it("18. Check the division of a number by zero.", function() {

calculator.get();

calculator.setFirst('20');

calculator.setOperator('/');

calculator.setSecond('0');

calculator.doExecute();

expect(calculator.getResult()).toEqual('Infinity');

});

it("19. Check the division of a positive integer number by negative decimal number", function() {

calculator.get();

calculator.setFirst('20');

calculator.setOperator('/');

calculator.setSecond('-2.5');

calculator.doExecute();

expect(calculator.getResult()).toEqual('-8');

});

it("20. Check the division of zero by any number", function() {

calculator.get();

calculator.setFirst('0');

calculator.setOperator('/');

calculator.setSecond('-5.1');

calculator.doExecute();

expect(calculator.getResult()).toEqual('0');

});

});

Kode Sumber . Spesifikasi Protractor JS

# BIODATA PENULIS



Hanif Sudira, lahir pada tanggal 22 April 1995 di Padang. Penulis menempuh pendidikan mulai dari SDN 29 Padang (2001 – 2007), SMPN 12 Padang (2007 – 2010), SMAN 10 Padang (2010 – 2013). Saat ini penulis sedang menempuh pendidikan perguruan tinggi di Institut Teknologi Sepuluh Nopember Surabaya di jurusan Teknik Informatika Fakultas Teknologi Informasi angkatan tahun 2013.

Penulis memliki bidang minta di Laboratorium Komputasi Berbasis Jaringan (KBJ) dengan fokus studi Forensik Digital. Penulisan dapat dihubungi melalu email : **sudirahanif@gmail.com**.