**ANALISIS PERBANDINGAN FRAMEWORK LARAVEL DENGAN FRAMEWORK CODEIGNITER PADA APLIKASI DIAGNOSA PENYAKIT UNTUK ANAK MENGGUNAKAN   
METODE FORWARD CHAINING**

**SKRIPSI**

Karya Tulis sebagai syarat memperoleh  
Gelar Sarjana Komputer dari Fakultas Teknologi Informasi  
Universitas Bale Bandung

Disusun oleh:

GALIH REXY HAKIKI  
NPM. C1A160009



PROGRAM STRATA 1

PROGRAM STUDI TEKNIK INFOMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS BALE BANDUNG

BANDUNG

2020

**LEMBAR PERSETUJUAN PEMBIMBING**

**ANALISIS PERBANDINGAN FRAMEWORK LARAVEL DENGAN FRAMEWORK CODEIGNITER PADA APLIKASI DIAGNOSA PENYAKIT UNTUK ANAK MENGGUNAKAN   
METODE FORWARD CHAINING**

Disusun oleh:

GALIH REXY HAKIKI

NPM. C1A160009

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

**SARJANA KOMPUTER**

Pada

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS BALE BANDUNG**

Baleendah, Juli 2020

Disetujui oleh

|  |  |
| --- | --- |
| Pembimbing 1 | Pembimbing 2 |
|  |  |
| Yaya Suharya, S.Kom., M.T.  NIK. 0407047706 | Nurul Imamah, S.T., M.T.  NIK. 04104808121 |

**LEMBAR PENGESAHAN PENGUJI**

**ANALISIS PERBANDINGAN FRAMEWORK LARAVEL DENGAN FRAMEWORK CODEIGNITER PADA APLIKASI DIAGNOSA PENYAKIT UNTUK ANAK MENGGUNAKAN   
METODE FORWARD CHAINING**

Disusun oleh:

GALIH REXY HAKIKI

C1A160009

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

**SARJANA KOMPUTER**

Pada

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS BALE BANDUNG**

Baleendah, Juli 2020

Disetujui oleh

|  |  |
| --- | --- |
| Penguji 1 | Penguji 2 |
|  |  |
| Zen Munawar, S.Kom., M.Kom  NIDN. 0422037002 | Denny Rusdianto, S.T., S.Kom.  NIDN. 04104808094 |

**LEMBAR PERSETUJUAN PROGRAM STUDI**

**ANALISIS PERBANDINGAN FRAMEWORK LARAVEL DENGAN FRAMEWORK CODEIGNITER PADA APLIKASI DIAGNOSA PENYAKIT UNTUK ANAK MENGGUNAKAN   
METODE FORWARD CHAINING**

Disusun oleh:

GALIH REXY HAKIKI

C1A160009

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

**SARJANA KOMPUTER**

Pada

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS BALE BANDUNG**

Baleendah, Juli 2020

Disetujui oleh

|  |  |
| --- | --- |
| Mengetahui, Dekan, | Mengesahkan, Ketua Program Studi |
|  |  |
| Yudi Herdiana, S.T., M.T.  NIK. 04104808008 | Yaya Suharya, S.Kom., M.T.  NIK. 0407047706 |

**LEMBAR PERNYATAAN KEASLIAN**

Yang bertanda tangan di bawah ini :

Nama : GALIH REXY HAKIKI

NPM : C1A160009

Judul Skripsi : ANALISIS PERBANDINGAN FRAMEWORK LARAVEL DENGAN FRAMEWORK CODEIGNITER PADA APLIKASI DIAGNOSA PENYAKIT UNTUK ANAK MENGGUNAKAN METODE FORWARD CHAINING

Menyatakan dengan sebenarnya bahwa penulisan laporan skripsi ini berdasarkan hasil penelitian, pemikiran dan pemaparan asli dari penyusun sendiri, baik untuk naskah laporan maupun kegiatan pemrograman yang tercantum sebagai bagian dari laporan skripsi ini, jika terdapat karya orang lain maka penyusun akan mencantumkan sumber secara jelas dan apabila ada karya pihak lain yang ternyata memiliki kemiripan dengan karya penyusun yang telah penyusun buat ini, maka hal ini adalah di luar pengetahuan penyusun dan terjadi tanpa kesengajaan.

Dengan demikian pernyataan ini penyusun buat dengan sesungguhnya dan apabila di kemudian hari terdapat penyimpangan dan ketidak benaran dalam pernyataan ini, maka penyusun bersedia menerima sanksi akademik yang sesuai dengan norma yang berlaku di perguruan tinggi ini.

|  |
| --- |
| Bandung, 27 Juli 2020 |
| Yang Membuat Pernyataan, |
|  |
| GALIH REXY HAKIKI |
| NPM. C1A160009 |

# ABSTRACT

*The development of an increasingly modern era has created a means of delivering information quickly and in real time. The website is currently experiencing a significant increase from time to time. With the development of the website created a framewok technology. Where in making a website can be more structured and shorter. Programming language. PHP is a popular language and has several frameworks. Among them are the Laravel framework and the Codeigniter framework which are the most widely used framework for the PHP programming language.*

*To compare the two frameworks, the writer will make a disease diagnosis application for children under 5 years using the forward chaining method which will be used to compare the two frameworks in terms of performance (request per second, time per request and execute time), the size of the data generated (resourse ), and how to access the database. Both the Laravel framework and the Codeigniter framework use the MVC concept. M for Model is used as a system for database access, V for View is used to display views on the website, and C for Controller is used as a regulator in the use of Models and View.*

*The results showed that in terms of performance (requests per second, time per request and execute time) as a whole, CodeIgniter has a superior value. In terms of size (resource) the Codeigniter Framework has a smaller size while the Laravel Framework has a larger size because there is already a package that will be very helpful, And in terms of how to access the database, both the CodeIgniter Framework and the Laravel Framework have the same application. Both are able to simplify the operation of CRUD With this research it is expected that web developers can choose the technology in accordance with what is desired.*

***Keywords:*** *Frameworks, Laravel, Codeigniter, Expert System, Forward Chaining.*

# ABSTRAK

*Perkembangan zaman yang semakin modern telah menciptakan sarana penyampaian informasi secara cepat dan real time. Website pada saat ini terus mengalami peningkatan yang signifikan dari waktu ke waktu. Dengan berkembangnya website tercipta sebuah teknologi framewok. Dimana dalam pembuatan website pun dapat lebih terstruktur dan lebih singkat. Bahasa pemrogramman. PHP merupakan salah satu bahasa yang populer dan memiliki beberapa framework. Diantaranya ada framework Laravel dan framewok Codeigniter yang merupakan framework paling banyak digunakan untuk bahasa pemrogramman PHP.*

*Untuk membandingkan kedua framework tersebut penulis akan membuat aplikasi diagnosa penyakit untuk anak dibawah 5 tahun menggunakan metode forward chaining yang akan digunakan untuk membandingkan kedua framework tersebut dari segi performa (request per second, time per request dan execute time), ukuran data yang dihasilkan (resourse), dan cara akses database. Baik framework Laravel maupun framework Codeigniter menggunakan konsep MVC. M untuk Model digunakan sebagai sistem untuk pengaksesan database, V untuk View digunakan untuk menampilkan tampilan pada website, dan C untuk Controller digunakan sebagai pengatur dalam penggunaan Model dan View.*

*Hasil penelitian menunjukkan bahwa dari segi performa (request per second, time per request dan execute time) secara keseluruhan, CodeIgniter memiliki nilai yang lebih unggul. Dari segi ukuran (resource)Framework Codeigniter memiliki ukuran yang lebih kecil sedangkan Framework Laravel memiliki ukuran yang lebih besar karena sudah terdapat package yang akan sangat membantu, Dan dari segi cara akses database, baik Framework CodeIgniter maupun Framework Laravel memiliki penerapan yang sama. Keduanya mampu mempermudah dalam operasi CRUD Dengan adanya penelitian ini diharapkan pengembang web dapat memilih teknologi sesuai dengan yang diinginkan.*

***Kata kunci*:** *Framework, Laravel, Codeigniter, Sistem Pakar, Forward Chaining.*

# KATA PENGANTAR

Alhamdulillahirobbilalamin, puji dan syukur penyusun panjatkan kehadirat Allah SWT atas segala nikmat dan karunia Nya. Atas izin-Nya, melalui berbagai macam proses akhirnya penulis dapat menyelesaikan proposal skripsi ini dengan lancar. Tanpa kuasa-Nya.

Tantangan dan hambatan selama pengerjaan laporan skripsi dapat diselesaikan berkat bantuan dari beberapa pihak yang terus memberikan dukungan dan masukan. Oleh karena itu, penyusun mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Bapak Yudi Herdiana, S.T., M.T., selaku Dekan Fakultas Teknologi Informasi Universitas Bale Bandung.
2. Bapak Yaya Suharya, S.Kom., M.T., selaku Ketua Program Studi Teknik Informatika dan selaku Pembimbing 1.
3. Bu Nurul Imamah, S.T., M.T., selaku Pembimbing 2
4. Bapak Mohammad Ridwan, S.T., M.Kom., yang turut membantu pembimbing 2 dalam proses bimbingan.
5. Bapak Zen Munawar, S.Kom., M.Kom., selaku Penguji 1.
6. Bapak Denny Rusdianto, S.T., S.Kom., selaku Penguji 2
7. Kedua orang tua yang telah memberi berbagai macam bantuan baik secara dorongan doa,motivasi, moral dan materi.
8. Teman seangkatan yang telah melewati kebersamaan baik senang maupun susah.
9. Semua pihak yang telah membantu dan memberikan dukungan kepada penyusun untuk menyelesaikan laporan ini.

Dalam proses penulisan skripsi ini penulis telah berusaha semaksimal mungkin untuk menghasilkan yang terbaik, Penulis sadar bahwa proposal ini masih jauh dari sempurna. Untuk itu penulis mengharapkan adanya kritik dan saran dari semua pihak. Semoga penelitian ini bermanfaat bagi para seluruh pembaca dan dapat dikembangkan untuk kemajuan ilmu pengetahuan nantinya.

|  |
| --- |
| Bandung, Juli 2020  Galih Rexy Hakiki |

# DAFTAR ISI

[ABSTRACT v](#_Toc53762474)

[ABSTRAK vi](#_Toc53762475)

[KATA PENGANTAR vii](#_Toc53762476)

[DAFTAR ISI ix](#_Toc53762477)

[DAFTAR TABLE xii](#_Toc53762478)

[DAFTAR GAMBAR xiii](#_Toc53762479)

[BAB I PENDAHULUAN 1](#_Toc53762480)

[1.1. Latar Belakang 1](#_Toc53762481)

[1.2. Rumusan Masalah 2](#_Toc53762482)

[1.3. Pembatasan Masalah 2](#_Toc53762483)

[1.4. Tujuan 3](#_Toc53762484)

[1.5. Metodologi Penelitian 4](#_Toc53762485)

[1.6. Metode Pengujian 6](#_Toc53762486)

[1.7. Sistematika Penulisan 7](#_Toc53762487)

[BAB II TINJAUAN PUSTAKA 8](#_Toc53762488)

[2.1. Tinjauan Penelitian Sebelumnya 8](#_Toc53762489)

[2.2. Dasar Teori 9](#_Toc53762490)

[2.2.1. HTML 9](#_Toc53762491)

[2.2.2. CSS 10](#_Toc53762492)

[2.2.3. PHP 11](#_Toc53762493)

[2.2.4. MySql 12](#_Toc53762494)

[2.2.5. Web Browser 13](#_Toc53762495)

[2.2.6. XAMPP 14](#_Toc53762496)

[2.2.7. Framework 14](#_Toc53762497)

[2.2.8. Framework Laravel 16](#_Toc53762498)

[2.2.9. Framework Codeigniter 17](#_Toc53762499)

[2.2.10. Sublime Text 18](#_Toc53762500)

[2.2.11. Bootstrap 4.1.0 20](#_Toc53762501)

[2.2.12. Forward Chaining 21](#_Toc53762502)

[2.2.13. Waterfall 22](#_Toc53762503)

[2.2.14. Sistem Pakar 25](#_Toc53762504)

[2.2.15. Request Per Second (RPS) 30](#_Toc53762505)

[2.2.16. Execute time 31](#_Toc53762506)

[BAB III METODOLOGI PENELITIAN 32](#_Toc53762507)

[3.1. Kerangka Pikir 32](#_Toc53762508)

[3.2. Deskripsi Teori 33](#_Toc53762509)

[3.2.1. Observasi Awal 33](#_Toc53762510)

[3.2.2. Identifikasi Masalah 33](#_Toc53762511)

[3.2.3. Metodologi Pengumpulan Data 33](#_Toc53762512)

[3.2.4. Pengembangan Sistem 34](#_Toc53762513)

[3.2.5. Pengujian Framework 35](#_Toc53762514)

[3.2.6. Pembuatan Laporan 36](#_Toc53762515)

[BAB IV ANALISIS DAN PERANCANGAN 37](#_Toc53762516)

[4.1. Analisis 37](#_Toc53762517)

[4.1.1. Analisis Masalah 37](#_Toc53762518)

[4.1.2. Analisis Kebutuhan 38](#_Toc53762519)

[4.1.3. Analisis Pengguna 39](#_Toc53762520)

[4.1.4. User Interface 40](#_Toc53762521)

[4.1.5. Fitur-Fitur 45](#_Toc53762522)

[4.1.6. Analisis Data 45](#_Toc53762523)

[4.2. Perancangan Aplikasi 48](#_Toc53762524)

[4.2.1. Unified Modeling Language (UML) 48](#_Toc53762525)

[4.2.1.1. Use Case Diagram 48](#_Toc53762526)

[4.2.1.2. Activity Diagram 49](#_Toc53762527)

[4.2.1.3. Sequence Diagram 53](#_Toc53762528)

[4.2.2. ERD 56](#_Toc53762529)

[4.2.3. Struktur Tabel 60](#_Toc53762530)

[4.3. Perancangan Pengujian 62](#_Toc53762531)

[4.3.1. Perancangan Analisis Performa 62](#_Toc53762532)

[4.3.2. Perancangan Analisis Ukuran 63](#_Toc53762533)

[4.3.3. Perancangan Analisis Cara Akses Database 63](#_Toc53762534)

[BAB V IMPLEMENTASI DAN PENGUJIAN 64](#_Toc53762535)

[5.1. Implementasi 64](#_Toc53762536)

[5.1.1. Implementasi User Interface 64](#_Toc53762537)

[5.1.2. Implementasi Metode Forward Chaining 71](#_Toc53762538)

[5.2. Pengujian 73](#_Toc53762539)

[5.2.1. Pengujian Segi Performa 73](#_Toc53762540)

[5.2.2. Pengujian Segi Ukuran 82](#_Toc53762541)

[5.2.3. Pengujian Cara Akses Database 84](#_Toc53762542)

[BAB VI KESIMPULAN DAN SARAN 89](#_Toc53762543)

[DAFTAR PUSTAKA 91](#_Toc53762544)

[LAMPIRAN 92](#_Toc53762545)

# DAFTAR TABLE

[Tabel 4. 1 Spesifikasi Hardware 39](#_Toc46719734)

[Tabel 4. 2 Data keluhan 45](#_Toc46719735)

[Tabel 4. 3 Data Penyakit 46](#_Toc46719736)

[Tabel 4. 4 Data gejala 48](#_Toc46719737)

[Tabel 4. 5 Data aturan diagnosa 48](#_Toc46719738)

[Tabel 4. 6 Simbol dan komponen Entitas Relationship Diagram 57](#_Toc46719739)

[Tabel 4. 7 ERD 59](#_Toc46719740)

[Tabel 4. 8 User 60](#_Toc46719741)

[Tabel 4. 9 Anak 60](#_Toc46719742)

[Tabel 4. 10 Diagnosa 60](#_Toc46719743)

[Tabel 4. 11 Detail Diagnosa 61](#_Toc46719744)

[Tabel 4. 12 Gejala 61](#_Toc46719745)

[Tabel 4. 13 Gejala Return 61](#_Toc46719746)

[Tabel 4. 14 Keluhan 61](#_Toc46719747)

[Tabel 4. 15 Penyakit 61](#_Toc46719748)

[Tabel 5. 1 Request per second Laravel 75](#_Toc46719749)

[Tabel 5. 2 Request per second Codeigniter 75](#_Toc46719750)

[Tabel 5. 3 Pengujian time per second Laravel 77](#_Toc46719751)

[Tabel 5. 4 Pengujian time per second Codeigniter 77](#_Toc46719752)

[Tabel 5. 5 Execute time proses registrasi 79](#_Toc46719753)

[Tabel 5. 6 Execute time proses login 80](#_Toc46719754)

[Tabel 5. 7 Execute time proses tambah data anak 80](#_Toc46719755)

[Tabel 5. 8 Execute time proses diagnosa 81](#_Toc46719756)

[Tabel 5. 9 Execute time proses data diagnosa 81](#_Toc46719757)

[Tabel 5. 10 Rata-rata Execute time 82](#_Toc46719758)

[Tabel 5. 11 Direktori Laravel 83](#_Toc46719759)

[Tabel 5. 12 Direktori Codeigniter 84](#_Toc46719760)

# DAFTAR GAMBAR

[Gambar 2. 1 Struktur SDLC 23](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46711121)

[Gambar 2. 2 Struktur Paralel Development 24](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46711122)

[Gambar 2. 3 Expert System 26](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46711123)

[Gambar 2. 4 Problem & Knowledge Domain 27](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46711124)

[Gambar 3. 1 Kerangka Pikir 32](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46711130)

[Gambar 4. 1 Referensi Desain 40](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719775)

[Gambar 4. 2 Rancangan Register 41](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719776)

[Gambar 4. 3 Rancangan Login 41](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719777)

[Gambar 4. 4 Referensi Desain 3 42](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719778)

[Gambar 4. 5 Referensi Desain 2 42](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719779)

[Gambar 4. 6 Perancangan Halaman Home 43](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719780)

[Gambar 4. 7 Perancangan Halaman Profil 43](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719781)

[Gambar 4. 8 Perancangan Halaman Diagnosa 44](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719782)

[Gambar 4. 9 Perancangan Halaman Data Diagnosa 44](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719783)

[Gambar 4. 10 Use case user 49](#_Toc46719784)

[Gambar 4. 11 Activity diagram tambah data anak 50](#_Toc46719785)

[Gambar 4. 12 Activity diagram proses diagnosa 51](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719786)

[Gambar 4. 13 Activity diagram lihat hasil diagnosa 52](#_Toc46719787)

[Gambar 4. 14 Sequence diagram tambah data anak 53](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719788)

[Gambar 4. 15 Sequence diagram proses diagnosa 54](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719789)

[Gambar 4. 16 Sequence diagram lihat hasil diagnosa 55](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719790)

[Gambar 5. 1 Halaman Login 65](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719791)

[Gambar 5. 2 Halaman Home 65](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719792)

[Gambar 5. 3 Halaman Profil 66](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719793)

[Gambar 5. 4 Tambah Data Profil 66](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719794)

[Gambar 5. 5 Data Anak 67](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719795)

[Gambar 5. 6 Halaman Diagnosa 67](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719796)

[Gambar 5. 7 Pilih data anak 68](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719797)

[Gambar 5. 8 Pilih Keluhan 68](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719798)

[Gambar 5. 9 Pilih Gejala 1 69](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719799)

[Gambar 5. 10 Pilih Gejala 2 69](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719800)

[Gambar 5. 11 Data Diagnosa 70](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719801)

[Gambar 5. 12 Detail Diagnosa 70](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719802)

[Gambar 5. 13 Forward Chaining Penyakit 1 71](#_Toc46719803)

[Gambar 5. 14 Forward Chaining Penyakit 2 72](#_Toc46719804)

[Gambar 5. 15 Forward Chaining Penyakit 3 72](#_Toc46719805)

[Gambar 5. 16 Perintah untuk server Laravel 73](#_Toc46719806)

[Gambar 5. 17 Perintah untuk server Codeigniter 74](#_Toc46719807)

[Gambar 5. 18 Tampilan hasil tool ab 74](#_Toc46719808)

[Gambar 5. 19 Perintah time per request Laravel 76](#_Toc46719809)

[Gambar 5. 20 Perintah time per request Codeigniter 76](#_Toc46719810)

[Gambar 5. 21Penerapan pengujian execute time 78](file:///D:\Perkuliahan\Semester%208\SKRIPSI\Word%20SKRIPSI\print\UPS1%20-%20revisi%20-%202%20(Repaired).docx#_Toc46719811)

[Gambar 5. 22 Model Diagnosa 85](#_Toc46719812)

[Gambar 5. 23 Controller Diagnosa 85](#_Toc46719813)

[Gambar 5. 24 Create Laravel 86](#_Toc46719814)

[Gambar 5. 25 Model Diagnosa Codeigniter 87](#_Toc46719815)

[Gambar 5. 26 Controller Diagnosa Codeigniter 87](#_Toc46719816)

[Gambar 5. 27 Create Codeigniter 88](#_Toc46719817)

# BAB I PENDAHULUAN

## Latar Belakang

Dengan perkembangan zaman yang semakin modern, penyebaran informasi menggunakan media yang lebih efektif dan efisien. Awalnya semua berita, info, event dan lain lain hanya dapat diperoleh dari buku, koran, radio dan televisi. Namun sejak munculnya internet, semua lapisan masyarakat dapat memperoleh informasi yang cepat, akurat dan relevan dalam sebuah halaman web yang dapat diakses secara real time dari gadget maupun komputer. Perusahaan atau instansi saat ini dalam memberikan berbagai macam informasi menggunakan media perantara yaitu website. Pembuatan sebuah website mengalami banyak perkembangan. Perkembangan tersebut terbukti dengan banyaknya pilihan framework yang dapat digunakan untuk membangun sebuah website. Framework adalah sebuah kerangka kerja dari bahasa pemrograman dasar yang telah dikembangkan dan dipermudah penggunaannya agar suatu website dapat diselesaikan dalam waktu yang relatif singkat. Framework yang digunakan untuk membangun sebuah website yang sedang populer saat ini seperti CodeIgniter dan Laravel.

CodeIgniter merupakan framework yang diklaim memiliki eksekusi tercepat dibandingkan dengan framework lainnya. CodeIgniter bersifat open source dan menggunakan model berbasis MVC (Model View Controller), yang 2 merupakan konsep modern framework yang digunakan saat ini. Framework CodeIgniter memiliki beberapa kelebihan diantaranya gratis, berukuran kecil dan cepat, dokumentasi, menggunakan konsep MVC, portability (Raelda Rispadina Sitio, 2013).

Laravel adalah salah satu dari sekian banyak framework PHP yang tersedia. Laravel dibuat oleh Taylor Otwell sejak tahun 2011. Framework ini mengaku “clean and classy”, dengan kode yang lebih singkat, mudah dimengerti, dan

ekspressif. Kelebihan Laravel adalah expressif, simpel, tersedia composer, open source, kompatibel dengan PHP 5.3 keatas, dokumentasi yang lengkap, prinsip model view controller (Asli Khatul Khasanah, 2015).

Aplikasi diagnosa untuk anak merupakan aplikasi sistem pakar untuk mendiagnosa seorang anak terkena suatu penyakit. Aplikasi ini ditujukan untuk anak 5 tahun kebawah karena pada umur tersebut rawan bagi anak terkena penyakit. Aplikasi juga dibuat semata hanya untuk menganalisis perbandingan framework. Penulis akan membandingkan Framework CodeIgniter dan Framework Laravel dari segi performa (*request per second, time per request, dan execute time*), segi ukuran, dan cara akses database.

## Rumusan Masalah

Dari latar belakang yang telah dijabarkan diatas, maka dapat dirumuskan suatu rumusan masalah adalah sebagai berikut:

1. Bagaimana mendapatkan hasil perbandingan Framework Laravel dan Framework Codeigniter dari segi kecepatan/*performa*?
2. Bagaimana mendapatkan hasil perbandingan Framework Laravel dan Framework Codeigniter dari segi ukuran/*resource*?
3. Bagaimana mendapatkan hasil perbandingan Framework Laravel dan Framework Codeigniter dari segi cara akses *database*?
4. Bagaimana pembuatan aplikasi diagnosa penyakit pada anak dengan menggunakan Framework Laravel dan Framework Codeigniter?

## Pembatasan Masalah

Untuk menjelaskan permasalahan agar pembahasan ini tidak terlalu jauh dari kajian masalah yang penulis paparkan. maka penulis membatasi masalah pada hal-hal berikut ini:

1. Aplikasi yang digunakan untuk penelitian ini adalah aplikasi diagnosa penyakit pada anak umur 5 ke bawah.
2. Perbandingan framework Laravel dan framework Codeigniter dari segi performa, ukuran dan jumlah penggunaan memory.
3. Performa diambil dari pengukuran *request per second, time per second* dan *execute time.*
4. *Request per second dan time per request* diambil dengan *tool* Apache Benchmark (ab).
5. *Execute time* diambil dengan menggunakan fungsi yang sudah tersedia pada PHP. Pengujian dilakukan dengan menambahkan fungsi pada awal skrip kode dan akhir skrip kode. Maka akan didapat waktu *execute time* dari tiap fitur yang tersedia.
6. Cara akses *database* dilakukan dengan membandingkan dan menganalisis akses *database* untuk tata cara pengaksesan tabel dalam operasi *Create, Read, Update, Delete* (CRUD). Cara pengaksesan *database* di Laravel menggunakan Eloquent ORM dan cara pengaksesan *database* di CodeIgniter menggunakan Query Builder.
7. Aplikasi yang dibuat hanya berfokus menganalisis perbandingan framework dan tidak akan di*hosting* pada *server online*.

## Tujuan

Adapun tujuan yang ingin dicapai dalam penelitian ini, adalah:

1. Menghasilkan analisis dari perbandingan Framework Laravel dan Framework Codeigniter dari segi performa.
2. Menghasilkan analisis dari perbandingan Framework Laravel dan Framework Codeigniter dari segi ukuran.
3. Menghasilkan analisis dari perbandingan Framework Laravel dan Framework Codeigniter dari segi cara akses database.

## Metodologi Penelitian

* 1. Kerangka Kerja Penelitian

Untuk membantu dalam menyelesaikan penyusunan penelitian, maka perlu adanya susunan kerangka kerja yang jelas. Kerangka kerja ini merupakan langkah-langkah yang akan dilakukan dalam penyelesaian masalah yang akan dibahas, dengan tahapan-tahapan sebagai berikut:

1. Observasi Awal

Pada observasi awal penulis melakukan pengamatan melalui internet dan menentukan masalah yang selanjutnya ditentukan tujuan serta pemecahan masalahnya.

1. Pengumpulan Data

Pada tahap ini dilakukan pengumpulan data dengan metode studi pustaka untuk mendapatkan data yang dibutuhkan penulis

1. Pengembangan Sistem

Pada tahap ini dilakukan proses pengembangan sistem dengan menggunakan metode SDLC (*System Development Life Cycle*) dengan metode *Waterfall* untuk pembuatan aplikasinya.

1. Pengujian

Pada tahap ini dilakukan pengujian perbandingan antara framework Laravel dan framework codeigniter.

1. Pembuatan Laporan

Pada tahap ini dilakukan proses pembuatan laporan yang disusun berdasarkan hasil penelitian.

* 1. Metode Pengumpulan Data

Dalam melakukan penelitian ini, penulis melakukan beberapa metode pengumpulan data dan referensi yang dibutuhkan dalam penelitian ini. Adapun beberapa cara tersebut adalah sebagai berikut:

* 1. Wawancara

Pengumpulan data dengan mengadakan wawancara atau tanya jawab baik secara *offline* atau *online* kepada beberapa orang yang telah menggunakan framework Laravel dan Codeigniter.

* 1. Studi Pustaka

Mengumpulkan data dengan cara membaca referensi dari buku-buku dan jurnal-jurnal yang berhubungan dengan topik penelitian.

* 1. Metode Pengembangan

Metode yang digunakan penulis dalam pengembangan aplikasi adalah metode Waterfall yang memiliki tahapan seperti berikut:

1. *Analysis*

Pada tahapan ini dilakukan analisis kebutuhan aplikasi dan perancangan aplikasi. Analisis dan perancangan dilakukan untuk menentukan struktur database yang akan digunakan untuk penyimpanan data pada aplikasi.

1. *Desain*

Pada tahapan ini dilakukan pengumpulan data untuk kebutuhan aplikasi. Data yang dikumpulkan adalah referensi desain aplikasi, gambar ilustrasi, skema warna, pembuatan logo dan jenis font.

1. *Coding*

Pada tahapan ini dilakukan penulisan skrip kode untuk pembuatan 2 aplikasi dengan menggunakan Laravel dan Codeigniter.

1. *Testing*

Pada tahapan ini dilakukan pengujian apakah aplikasi berjalan dengan baik sesuai dengan fungsi yang telah ditentukan. Jika masih terdapat fungsi yang belum sesuai maka akan dilakukan perbaikan.

1. *Maintenance*

Pada tahapan ini dilakukan maintenance jika dibutuhkan. Dan aplikasi sudah lulus tes untuk masuk ke proses selanjutnya yaitu proses pengujian untuk membandingkan *framework* Laravel dan Codeigniter.

## Metode Pengujian

1. Pengujian Performa

Pengujian performa dilakukan dengan membandingkan Framework Laravel dengan Framework Codeigniter dari beberapa faktor antara lain:

* Pengujian Request Per Second (RPS)
* Pengujian Time Per Request
* Pengujian Execute Time (Waktu Eksekusi)

1. Pengujian Ukuran

Analisis ukuran dilakukan dengan membandingkan dan menganalisis struktur direktori dan besarnya total file yang terdapat pada direktori.

1. Pengujian Cara Akses Database

Cara akses database dilakukan dengan membandingkan dan menganalisis akses database untuk tata cara pengaksesan tabel dalam operasi Create, Read, Update, Delete (CRUD). Cara pengaksesan database di Laravel menggunakan Eloquent ORM dan cara pengaksesan database di CodeIgniter menggunakan Query Builder.

## Sistematika Penulisan

Gambaran mengenai keseluruhan skripsi dan pembahasannya dapat di jelaskan dalam sistematika penulisan sebagai berikut :

BAB I : PENDAHULUAN

Bab ini menjelaskan latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian, dan sistematika penulisan.

BAB II : TINJUAN PUSTAKA

Bab ini menjelaskan tentang landasan teori dan dasar teori pendukung dalam penelitian. Studi pustaka ini bersumber dari jurnal, tesis, buku teks, dan website.

BAB III : METODOLOGI PENELITIAN

Bab ini menjelaskan tentang gambaran kerangka pikir dan definisi dari gambaran kerangka pikir tersebut.

BAB IV : ANALISIS DAN PERANCANGAN

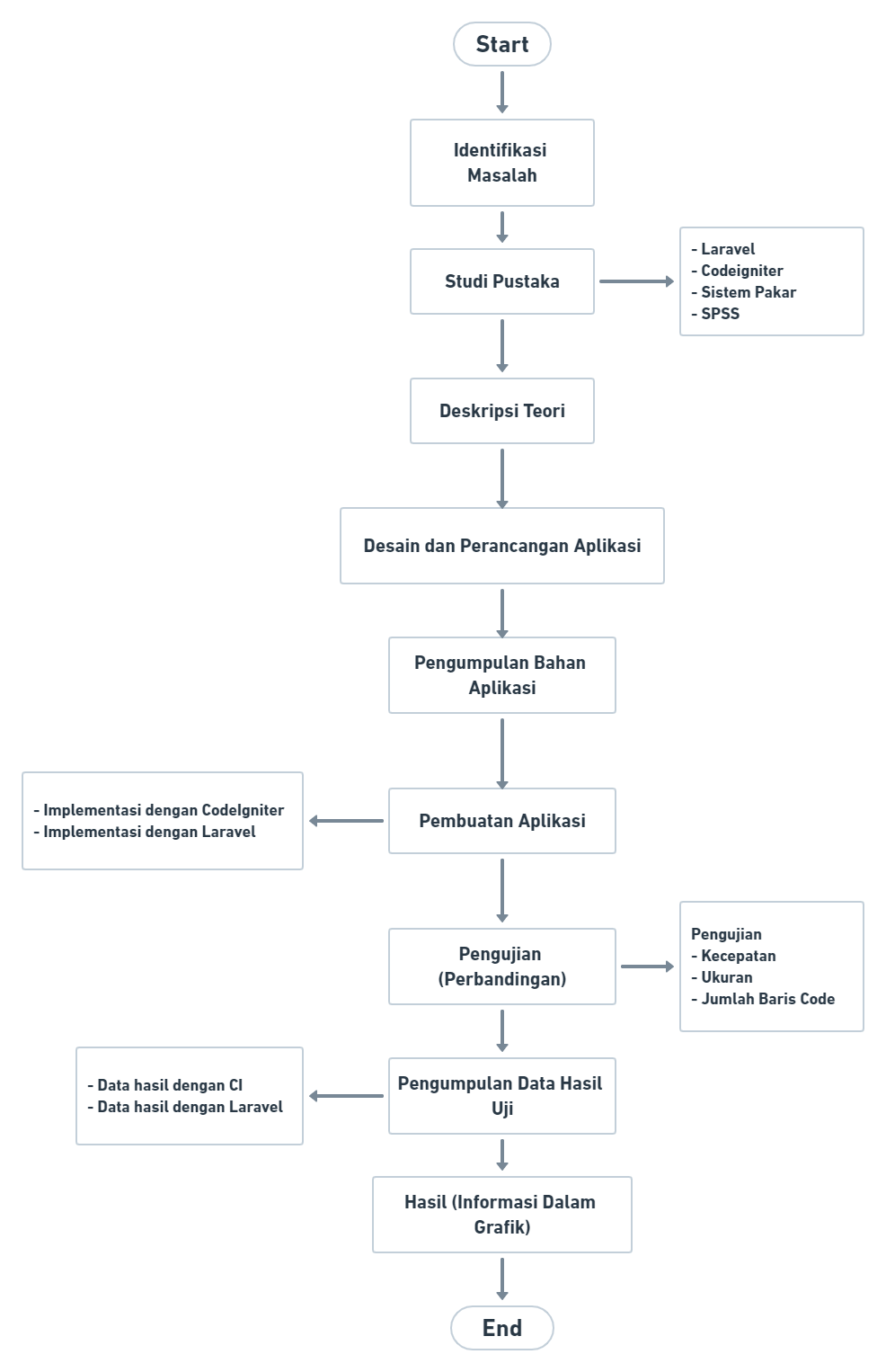
Bab ini menjelaskan tentang analisis dan perancangan dari aplikasi yang akan dibuat

BAB V : IMPLEMENTASI DAN PENGUJIAN

Bab ini menjelaskan tentang implementasi dan pengujian dari aplikasi yang sudah dibuat.

BAB VI : KESIMPULAN DAN SARAN

Bab ini berisi penarikan kesimpulam penelitian dan pengembangan yang dilakukan serta saran untuk mengembangkan aplikasi selanjutnya.



# BAB II TINJAUAN PUSTAKA

## Tinjauan Penelitian Sebelumnya

Untuk pelaksanaan penelitian, peserta skripsi menggunakan pengetahuan yang diperoleh selama masa perkuliahan sebagai landasan teori pengembangan. Pengetahuan dan teori yang digunakan antara lain:

1. Penelitian oleh Muhammad Nur Hamid (2019), yang berjudul “Analisis Perbandingan Framework Codeigniter dan Framework Laravel (Studi Kasus Inventaris HMJ TI STMIK AKAKOM YOGRAKARTA)”. Pada penelitian ini ditemukan masalah yang terjadi yakni pengembang web bingung Ketika harus memilih framewok yang akan mereka pakai untuk pengembangan webnya. Pada penelitian ini dibandingkan framework Laravel dan Codeigniter dari segi performa, cara akses database, dan implementasi fitur AJAX. Metode dari penelitian ini adalah bahan/data, peralatan, prosedur dan pengumpulan data, analisis dan rancangan sistem dan rancangan output analisis. Hasil dari penelitian ini adalah bahwa dari segi performa secara keseluruhan, CodeIgniter memiliki nilai yang lebih unggul. Dari segi cara akses database, Laravel memiliki pilihan yang lebih banyak. Dan segi implementasi fitur AJAX, baik Framework CodeIgniter maupun Framework Laravel memiliki penerapan yang sama karena sama-sama ditempatkan di sisi klien.

1. Penelitian oleh Ruli Erinton, Ridha Muldina Negara, Danu Dwi Sanjoyo (2017). Yang berjudul “Analisis Performasi Framework Codeigniter dan Laravel Menggunakan Web Server Apache”. Pada penelitian ini dilakukan pengujian terhadap 2 framework yaitu Laravel dan Codeiginiter dari segi performance yaitu pengujian load test dan pengujian stress test. Hasil dari penelitian ini adalah aplikasi web yang menggunakan framework Codeigniter lebih baik dari sisi performasinya dibandingkan dengan aplikasi web yang menggunakan framework Laravel. Nilai time pada Codeigniter 150,5 ms lebih rendah dibandingkan nilai time pada Laravel 254,5 ms. Nilai error testing didapat pada Laracel 79,7. Pada parameter QoS nilai throughput tertinggi 6,227 Mbps, packet loss 0%, retransmission terendah 1, delay terendah 91,46 dengan klasifikasi sangat baik berdasarkan standar ITU-T.

1. Penelitian oleh Bagus Fery Yanto, Indah Werdiningsih, Endah Purwanti (2017). Yang berjudul “*Aplikasi Sistem Pakar Diagnosa Penyakit Pada Anak Bawah Lima Tahun Menggunakan Metode Forward Chaining”*. Penelitian ini bertujuan untuk membuat sebuah aplikasi sistem pakar diagnosa penyakit pada Balita berbasis mobile. Penelitian ini terdiri dari tiga tahap. Tahap pertama adalah pengumpulan data dan informasi dari Manajemen Terpadu Balita Sakit (MTBS) dan wawancara dengan Bidan. Dari pengumpulan data dan informasi tersebut ditemukan fakta penyakit, keluhan, gejala dan saran penanganan. Tahap kedua adalah pembuatan rule dengan 18 penyakit. Tahap ketiga adalah implementasi aplikasi sistem pakar berbasis mobile dengan fitur diagnosa penyakit, riwayat diagnosa dan kumpulan penyakit. Aplikasi sistem pakar yang dibuat dapat mendiagnosa penyakit dan memberikan saran penanganan. Hasil evaluasi dari 50 data uji coba menghasilkan tingkat akurasi sebesar 82%, dimana 41 hasil diagnosa yang benar dan 9 diagnosa yang salah.

## Dasar Teori

* 1. HTML

HTML merupakan singkatan dari Hypertext Markup Language yaitu Bahasa standar web yang dikelola penggunanya oleh W3C (World Wide Web Consortium) berupa tag-tag yang Menyusun setiap elemen dari website. HTML berperan sebagai penyusun struktur halaman website yang menempatkan setiap elemen website sesuai layout yang diinginkan. Yang bisa dilakukan dengan HTML yaitu:

* Mengatur tampilan dari halaman web dan isinya.
* Membuat tabel dalam halaman web.
* Mempublikasikan halaman web secara online.
* Membuat form yang bisa digunakan untuk menangani registrasi dan transaksi via web.
* Menambahkan objek-objek seperti citra, audio, video, animasi, java applet dalam halaman web.
* Menampilkan area gambar (canvas) dibrowser.
  1. CSS

*Cascading Style Sheet* (CSS) dapat menyimpat format dan menggunakannya kapan pun dan di mana pun kita inginkan. Seperti *Formatting* dan *Style* dalam membuat dokumen office, maka *style sheets* juga sangat dinamis. Memang menggunakan *style sheet* bukan suatu keharusan dalam membuat web, namun ketika website dengan halaman yang sangat banyak kedepannya akan sulit untuk debugging, perbaikan dan perawatannya.

*Style sheet* dapat membuat efek-efek tertentu untuk konten web. Misalnya bagian header tabel pada web selalu ber-*font* Verdana, ukuran 16 dan berwarna hitam. Maka dapat didefinisikan *style* tersebut untuk tag <th>. Contoh lain misalnya tulisan pada suatu web bertipe Verdana dan ukutan 12, maka definisikan *style* tersebut pada tag <p>.

*Cascading Style Sheet* (CSS) sudah didukung oleh hamper semua web browser karena CSS telah distandarkan oleh World Wide Web Consortium (W3C). Jadi ini pilihan tepat bagi anda untuk memformat halaman web agar terlihat cantik di manapun web tersebut dibuka. Ada 4 cara memasang kode CSS ke dalam kode HTML / halaman web, yaitu:

* *Inline style sheet* (Memeasukkan kode CSS langsung pada tag HTML)
* *Internal Style Sheet* (*Embed* atau memasang kode CSS ke dalam bagian <head>)
* Me-link ke *external* CSS
* *Import* CSS file
  1. PHP

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs Personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari web.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis 2.0 ini, *interpreter* PHP sudah diimplementasikan dalam program C. Di dalam rilisan ini juga ikut disertakan modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Kemudian pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang *interpreter* PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada juni 1998, perusahaan tersebut merilis *interpreter* baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP diubah menjadi akronim berulang **PHP: *Hypertext Preprocessing.***

Pada pertengahan tahun 1999, Zend merilis *interpreter* PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari *interpreter* PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan Bahasa pemrograman kea rah paradigma berorientasi objek.

PHP memiliki kelebihan yaitu:

* PHP berbasis *Server Side scripting*
* *Command Line Scripting* pada PHP
* *PHP dapat membuat aplikasi desktop*
* *Digunakan untuk berbagai macam platform OS*
* *Mendukung berbagai macam web server*
* *Object Oriented Programming atau Procedural*
* *Output file PHP pada XHTML, HTML & XML*
* *Mendukung banyak RDMS (Database)*
* *Mendukung banyak komunikasi*
* *Pengolahan teks yang sangat baik*
  1. MySql

MySQL adalah salah satu aplikasi DBMS yang sudah sangat banyak digunakan oleh para pemrogram aplikasi web. Contoh DBMS lainnya adalah PostgreSQL, (*freeware*), SQL, Server, MS Access dari Microsoft, DB2 dari IBM, Oracle dari Oracle Corp, Dbase, FoxPro, dsb.

Kelebihan dari MySQL adalah gratis, handal, selalu di-update dan banyak forum yang memfasilitasi para pengguna jika memiliki kendala. MySQL juga menjadi DBMS yang sering dibundling dengan web server sehingga proses instalasinya lebih mudah.

* 1. Web Browser

Web browser secara umum adalah suatu perangkat lunak atau software yang digunakan untuk mencari informasi atau mengakses situs-situs yang ada di internet. Perangkat ini akan lebih memudahkan pengguna dalam mengakses data atau mencari referensi yang dibutuhkan. Ada berbagai macam perangkat web browser yang kini digunakan seperti Mozilla Firefox, Google Chrome, Opera, Safari, Internet Explorer dan lain sebagainya.

Awalnya, web browser berorientasi pada teks dan belum dapat menampilkan gambar. Namun, web browser sekarang tidak hanya menampilkan gambar dan teks saja, tetapi juga memutar file multimedia seperti video dan suara. Web browser juga dapat mengirim dan menerima email, mengelola HTML, sebagai input dan menjadikan halaman web sebagai hasil output yang informative.

Tujuan utama dari web browser adalah untuk membawa sumber informasi kepada pengguna. Proses ini dimulai ketika pengguna memasukan sebuah Uniform Resource Identifier (URI), misalnya, https://thidiweb.com ke dalam browser. Sumber yang telah diambil web browser akan ditampilkan. HTML ditampilkan ke mesin tata letak browser, dan akan diubah dari markup ke dokumen interaktif. Selain dari HTML, browser umumnya bisa menampilkan setiap jenis konten yang menjadi bagian dari suatu halaman web.

* 1. XAMPP

Kata XAMPP sendiri berasal dari:

* X yang berarti *cross platform* karena XAMPP bisa dijalankan di Windows, Linux, Mac dsb
* A yang berarti Apache sebagai web server-nya
* M yang berarti MySQL sebagai Database Management System (DBMS)-nya
* PP yang berarti PHP dan Perl sebagai Bahasa yang didukungnya.

Web server adalah tempat menyimpan aplikasi web kemudian mengaksesnya melalui Internet. Setiap perubahan, kecil maupun besar, Dibutuhkannya web server ini adalah karena untuk *server side script* seperti PHP, pemeriksaan baru akan tampil jika menggunakan web server. Itulah bedanya dengan *client side script* seperti HTML, CSS dan Javascript yang cukup dengan browser script dapat diketahui apakah sudah sesuai dengan keinginan atau belum

Web server Apache sudah sangat popular di Internet sejak April 1996. Kelebihan lainnya adalah The Apache Software Foundation sangat tinggi komitmenya untuk terus mengembangkan web server Apache sehingga keterjaminannya untuk senantiasa kompatibel dengan teknologi web terkini sangat tinggi. Dan yang terakhir, begitu banyak forum-forum untuk bertanya jika terdapat masalah dalam penggunaan web server Apache.

* 1. Framework

Framework atau kerangka kerja pengembangan aplikasi adalah suatu standar yang harus diikuti untuk melakukan pengembangan aplikasi oleh pemrogram. Standar ini mengatur banyak hak, mulai dari nama file, direktori, dan cara memprogramnya. Framework memberikan kerangka program, kumpulan library dan fungsi yang bisa langsung digunakan, serta aturan untuk menggunakannya.

Pengembanng aplikasi diminta untuk mengikuti aturan yang telah ditetapkan oleh framework, agar bisa menghasilkan sebuah aplikasi atau modul dengan cepat dengan menggunakan standar, File apa saja yang harus disiapkan dan isinya harus seerti apa adalah aturan yang telah ditetapkan oleh framework.

Framework pengembang aplikasi merupakan salah satu solusi untuk menjawab tuntutan agar bisa dengan cepat menyelesaikan pembuatan atau pengembangan aplikasi masa kini. Mengapa? Karena sekarang pengembang aplikasi dituntut untuk bisa segera memberikan hasil dari aplikasi yang diinginkan oleh pengguna atau pemberi pekerjaan.

Beberapa keuntungan yang didapat dalam penggunaan framework adalah:

1. Menghemat waktu pengembangan

Dengan *libraty* yang telah disediakan oleh framework maka tidak perlu lagi memikirkan hal-hal dasar atau hal-hal umum yang sudah dibuat pada *library*. Jika hanya fokus ke proses bisnis yang akan dikerjakan.

1. Penggunaan ulang program/kode

Dengan menggunakan framework maka pekerjaan akan memiliki struktur yang baku, sehingga dapat menggunakan program/kode itu kembali untuk pekerjaan lainnya

1. Bantuan komunitas

Pada umumnya setiao framework dimana komunitas inilah yang siap membantu jika ada permasalahan, selain itu juga bisa berbagi ilmu sehingga dapat meningkatkan kemampuan perograman.

1. Kumpulan program terbaik

Sebuah framework merupakan kumpulan program terbaik yang sudah teruji sehingga dapat meningkatkan kualitas program/kode yang dibuat.

* 1. Framework Laravel

Laravel adalah sebuah framework PHP yang dirilis dibawah lisensi MIT, dibangun dengan konsep MVC (model view controller). Laravel adalah pengembangan website berbasis MVP yang ditulis dalam PHP yang dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan, dan untuk meningkatkan pengalaman bekerja dengan aplikasi dengan menyediakan sintaks yang ekspresif, jelas dan menghemat waktu.

MVC adalah sebuah pendekatan perangkat lunak yang memisahkan aplikasi logika dari presentasi. MVC memisahkan aplikasi berdasarkan komponen- komponen aplikasi, seperti : manipulasi data, controller, dan user interface.

1. Model, Model mewakili struktur data. Biasanya model berisi fungsi-fungsi yang membantu seseorang dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.
2. View, View adalah bagian yang mengatur tampilan ke pengguna. Bisa dikatakan berupa halaman web.
3. Controller, Controller merupakan bagian yang menjembatani model dan view.

Beberapa fitur yang terdapat di Laravel :

* Bundles, yaitu sebuah fitur dengan sistem pengemasan modular dan tersedia beragam di aplikasi.
* Eloquent ORM, merupakan penerapan PHP lanjutan menyediakan metode internal dari pola “active record” yang menagatasi masalah pada hubungan objek database.
* Application Logic, merupakan bagian dari aplikasi, menggunakan controller atau bagian Route.
* Reverse Routing, mendefinisikan relasi atau hubungan antara Link dan Route.
* Restful controllers, memisahkan logika dalam melayani HTTP GET and POST.
* Class Auto Loading, menyediakan loading otomatis untuk class PHP.
* View Composer, adalah kode unit logikal yang dapat dieksekusi ketika view sedang loading.
* IoC Container, memungkin obyek baru dihasilkan dengan pembalikan controller.
* Migration, menyediakan sistem kontrol untuk skema database.
* Unit Testing, banyak tes untuk mendeteksi dan mencegah regresi.
* Automatic Pagination, menyederhanakan tugas dari penerapan halaman.
  1. Framework Codeigniter

Codeigniter (CI) adalah framework pengembangan aplikasi (Application Development Framework) dengan menggunakan PHP, suatu kerangka pembuatan program dengan menggunakan PHP. Pengembang dapat langsung menghasilkan program dengan cepat, dengan mengikuti kerangka kerja untuk membuat yang telah disiapkan oleh framework CI ini. Dengan menggunakan framework, kita tidak perlu membuat program daru awal, tetapi kita sudah diberikan library fungsi-fungsi yang sudah diorganisasi untuk dapat membuat suatu program dengan cepat. Kita hanya perlu memanggil fungsi-fungsi yang sudah ada untuk memproses data, kemudian memanggil fungsi untuk menamplkannya.

Codeigniter (CI) adalah framework pengembangan aplikasi (Application Development Framework) dengan menggunakan PHP, suatu kerangka untuk bekerja atau membuat program dengan menggunakan PHP yang lebih sistematis, Pemrogram tidak perlu membuat program dari awal (From scratch), karena CI menyediakan sekumpulan library dari fungsi yang banyak, yang diperlukan untuk menyelesaikan pekerjaan yang umum, dengan menggunakan antarmuka dan struktur logika yang sederhana untuk mengakses librarinya, Pemrogram dapat memfokuskan dari pada kode yang harus dibuat untuk menyelesaikan suatu pekerjaan.

* 1. Sublime Text

Sublime Text adalah aplikasi editor untuk kode dan teks  yang dapat berjalan diberbagai platform operating system dengan menggunakan teknologi Phyton API. Terciptanya aplikasi ini terinspirasi dari aplikasi Vim, Aplikasi ini sangatlah fleksibel dan powerfull. Fungsionalitas dari aplikasi ini dapat dikembangkan dengan menggunakan sublime-packages.  Sublime  Text  bukanlah  aplikasi  opensource  dan juga aplikasi yang dapat digunakan dan didapatkan secara gratis,  akan tetapi  beberapa  fitur  pengembangan  fungsionalitas  (packages)  dari  aplikasi  ini merupakan hasil dari temuan dan  mendapat dukungan  penuh dari  komunitas serta memiliki linsensi aplikasi gratis.

Sublime  Text  mendukung  berbagai  bahasa  pemrograman  dan  mampu menyajikan fitur  syntax  highlight  hampir di semua bahasa pemrogramman yang didukung  ataupun  dikembangkan  oleh  komunitas  seperti;  C,  C++,  C#, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, MATLAB, OCaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textile and XML. Biasanya bagi bahasa pemrograman yang didukung ataupun belum terdukung secara default dapat lebih dimaksimalkan atau didukung dengan menggunakan add-ons yang bisa didownload sesuai kebutuhan user.

Berikut beberapa fitur yang diunggulkan dari aplikasi Sublime Text:

* + Goto Anything

Fitur yang sangat membantu dalam membuka file ataupun menjelajahi isi dari file hanya dengan beberapa keystrokes.

* + Multiple Selections

Fitur ini memungkinkan user untuk mengubah secara interaktif banyak baris sekaligus, mengubah nama variabel dengan mudah, dan memanipulasi file lebih cepat dari sebelumnya.

* + Command Pallete

Dengan hanya beberapa keystorkes, user dapat dengan cepat mencari fungsi yang diinginkan, tanpa harus menavigasi melalu menu.

* + Distraction Free Mode

Bila user memerlukan fokus penuh pada aplikasi ini, fitur ini dapat membantu user dengan memberikan tampilan layar penuh.

* + Split Editing

Dapatkan hasil yang maksimal dari monitor layar lebar dengan dukungan editing perpecahan. Mengedit sisi file dengan sisi, atau mengedit dua lokasi di satu file. Anda dapat mengedit dengan banyak baris dan kolom yang user inginkan.

* + Instant Project Switch

Menangkap semua file yang dimasukkan kedalam project pada aplikasi ini. Terintegrasi dengan fitur Goto Anything untuk menjelajahi semua file yang ada ataupun untuk beralih ke file dalam project lainnya dengan cepat.

* + Plugin API

Dilengkapi dengan plugin API berbasis Python sehingga membuat aplikasi ini tangguh.

* + Customize Anything

Aplikasi ini memberikan  user  fleksibilitas  dalam  hal  pengaturan fungsional dalam aplkasi ini.

* + Cross Platform

Aplikasi ini dapat berjalan hampir disemua  operating system  modern seperti Windows, OS X, dan Linux based operating system.

* 1. Bootstrap 4.1.0

Bootstrap adalah library (pustaka / kumpulan fungsi-fungsi) dari Framework CSS yang dibuat khusus untuk bagian pengembangan fontend dari suatu website. Didalam library tersebut terdapat berbagai jenis file yang diantaranya HTML, CSS, dan Javascript. Hampir semua developer website menggunakan framework bootstrap agar memudahkan dan mempercepat pembuatan website. Karena semuanya sudah ada dalam frameworknya sehingga para develop / pengembang hanya tinggal membuat / menyisipkan class nya yang ingin dipakai seperti membuat tombol, grid navigasi dan lain sebagainya.

Bootstrap telah menyediakan kompulan aturan dan komponen [class interface](http://onestringlab.com/penjelasan-class-interface-pada-java/) dasar sebagai modal dalam pembuatan web yang telah dirancang sangat baik untuk memberikan tampilan yang sangat menarik, bersih, ringan dan memudahkan bagi penggunanya. Dan penggunaan bootstrap ini pengguna juga diberikan keleluasan salama pengembangan website, anda bisa merubah dan menambah class sesuai dengan keinginan.

Bootstrap awalnya dibuat dan dikembangkan oleh pekerja / programmer Twitter, yaitu Mark Octo dan Jacob Thornton sejak tahun 2011. Saat itu memang para programmer di Twitter menggunakan berbagai macam tools dan library yang dikuasai dan disukai untuk melakukan pekerjaannya, sehingga tidak ada standarisasi dalam penamaan suatu class. Akibatnya sulit untuk dikelola, maka dari itu keduanya membuat suatu tools ataupun framework yang digunakan bersama dilingkukan internal twitter.

Sejak diluncurkan pada bulan agustus 2011, bootstrap telah berevolusi dari proyek yang hanya basis css menjadi sebuah framework yang lebih lengkap yang juga berisi javascript plugin, icon, Forms, dan button.

Bootstrap sendiri sudah kompatibel dengan versi terbaru dari beberapa browser seperti google chrome, firefox, internet explorer, dan safari browser. Meskipun beberapa browser ini tidak didukung pada semua platform.

Beberapa alasan mengapa saat ini cukup banyak pengembang yang menggunakan Bootstrap dalam membuat front-end website, yaitu karena beberapa kelebihan yang dimiliki oleh Bootstrap itu sendiri yang antara lain:

* Dapat mempercepat waktu proses pembuatan front-end website
* Tampilan bootstrap yang sudah cukup terlihat modern.
* Tampilan Bootsrap sudah responsive,sehingga mendukung segala jenis resolusi, baik itu PC, tablet, dan juga smartphone.
* Website menjadi Sangat ringan ketika diakses, karena bootsrap dibuat dengan sangat terstruktur.
  1. Forward Chaining

Forward Chaining merupakan suatu penalaran yang dimulai dari fakta untuk mendapatkan kesimpulan (conclusion) dari fakta tersebut. Forward chaining bisa dikatakan sebagai strategi inference yang bermula dari sejumlah fakta yang diketahui. Pencarian dilakukan dengan menggunakan rules yang premisnya cocok dengan fakta yang diketahui tersebut untuk memperoleh fakta baru dan melanjutkan proses hingga goal dicapai atau hingga sudah tidak ada rules lagi yang premisnya cocok dengan fakta yang diketahui maupun fakta yang diperoleh.

Forward chaining bisa disebut juga runut maju atau pencarian yang dimotori data (data driven search). Jadi pencarian dimulai dari premis-premis atau informasi masukan (if) dahulu kemudian menuju konklusi atau derived information (then). Forward Chaining berarti menggunakan himpunan aturan kondisi-aksi. Dalam metode ini, data digunakan untuk menentukan aturan mana yang akan dijalankan atau dengan menambahkan data ke memori kerja untuk diproses agar ditemukan suatu hasil.

Forward Chaining digunakan jika :

1. Banyak aturan berbeda yang dapat memberikan kesimpulan yang sama.
2. Banyak cara untuk mendapatkan sedikit konklusi.
3. Benar-benar sudah mendapatkan berbagai fakta, dan ingin mendapatkan konklusi dari fakta-fakta tersebut.

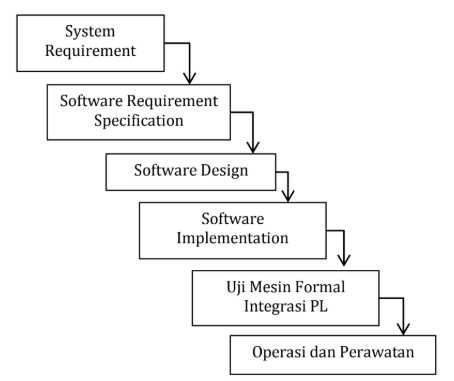
Adapun tipe sistem yang dapat menggunakan teknik pelacakan forward chaining, yakni :

1. Sistem yang direpresentasikan dengan satu atau beberapa kondisi.
2. Untuk setiap kondisi, sistem mencari rule-rule dalam knowledge base untuk rule-rule yang berkorespondensi dengan kondisi dalam bagian if.
3. Setiap rule dapat menghasilkan kondisi baru dari konklusi yang diminta pada bagian then. Kondisi baru ini dapat ditambahkan ke kondisi lain yang sudah ada.
4. Setiap kondisi yang ditambahkan ke sistem akan diproses. Jika ditemui suatu kondisi, sistem akan kembali ke langkah 2 dan mencari rule-rule dalam knowledge base Jika tidak ada konklusi baru, sesi ini berakhir.

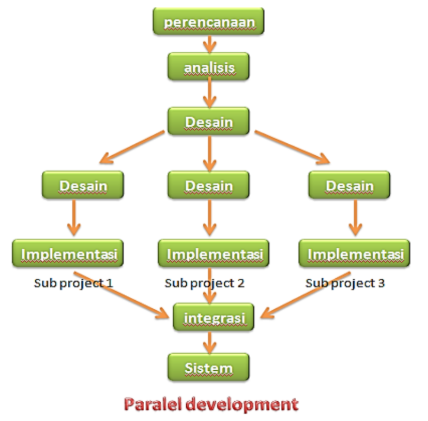
Jika klausa premis sesuai dengan situasi (bernilai true), maka proses akan meng-assert konklusi. Forward chaining juga digunakan jika suatu aplikasi menghasilkan tree yang lebar dan tidak dalam. Pada metode forward chaining, ada 2 cara yang dapat dilakukan untuk melakukan pencarian, yaitu :

1. Dengan memasukkan semua data yang tersedia ke dalam sistem pakar pada satu kesempatan dalam sesi konsultasi. Cara ini banyak berguna pada sistem pakar yang termasuk dalam proses terautomatisasi dan menerima data langsung dari komputer yang menyimpan data base, atau dari satu set sensor.
2. Dengan hanya memberikan elemen spesifik dari data yang diperoleh selama sesi konsultasi kepada sistem pakar. Cara ini mengurangi jumlah data yang diminta, sehingga data yang diminta hanyalah data-data yang benar-benar dibutuhkan oleh sistem pakar dalam mengambil kesimpulan.
   1. Waterfall

Waterfall merupakan model yang membangun perangkat lunak berdasarkan Daur Hidup Perangkat Lunak (SDLC), yaitu model yang mempunyai struktur yang dimulai dari perencanaan, analisis, desain dan implementasi, hingga tahap pengembangan dalam waterfall mempunyai struktur model pengembangan yang disebut dengan linier dan sequential.



Gambar 2. 1 Struktur SDLC

Dalam Waterfall ada proses-proses yang dapat berjalan bersamaan pada waktu tertentu yang sering disebut dengan Parallelism.

Gambar 2. 2 Struktur Paralel Development

Waterfall merupakan salah satu metode dalam SDLC yang mempunyai ciri khas pengerjaan yaitu setiap fase dalam waterfall harus diselessaikan terlebih dahulu sebelum melanjutkan ke fase selanjutnya. Artinya fokus terhadap masing-masing fase dapat dilakukan maksimal karena jarang adanya pengerjaan yang sifatnya parallel walaupun dapat saja terjadi paralelisme dalam waterfall.

Tahapan dari metode waterfall adalah sebagai berikut:

1. *Requirement Analysis*

Seluruh kebutuhan *software* harus bisa didapatkan dalam fase ini, termasuk di dalamnya kegunaan *software* yang diharapkan pengguna dan batasan *software.* Informasiini biasanya daapat diperoleh melalui wawancara, survey atau diskusi. Informasi tersebut dianalisis untuk mendapatkan dokumentasi kebutuhan pengguna untuk digunakan pada tahap selanjutnya.

1. *System Design*

Tahap ini dilakukan sebelum melakukan *coding.* Tahap ini bertujuan untuk memberikan gambaran apa yang seharusnya dikerjakan dan bagaimana tampilannya. Tahap ini membantu dalam memspesifikasikan kebutuhan *hardware* dan sistem serta mendefinisikan arsitektur sistem secara keseluruhan.

1. *Implementation*

Dalam tahapan ini dilakukan pemrograman. Pembuatan software dipecah menjadi modul-modul kecil yang nantinya akan digabungkan dalam tahap berikutnya. Selain itu dalam tahap ini juga dilakukan pemeriksaan terhadap modul yang dibuat, apakah sudah memenuhi fungsi yang diinginkan atau belum.

1. *Integration & Testing*

Di tahap ini dilakukan penggabungan modul-modul yang sudah dibuat dan dilakukan pengujian ini dilakukan untuk mengetahui apakah software yang dibuat telah sesuai dengan desainnya dan masih terdapat kesalahan atau tidak.

1. *Operation & Maintenance*

Ini merupakan tahap terakhir dalam model waterfall. Software yang sudah dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

* 1. Sistem Pakar

Sistem pakar sistem computer yang ditujukan untuk meniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*decision making*) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selaknya seorang pakar untuk memecahkan masalah.

Pakar atau ahli (*expert*) didefinisikan sebagai yang memiliki pengetahuan atau keahlian khusus yang tidak dimiliki oleh kebanyakan orang. Seorang pakar dapat memecahkan masalah yang tidak mampu dipecahkan kebanyakan orang. Dengan kata lain, dapat memecahkan suatu masalah dengan lebih efisien namun bukan berarti lebih murah. Pengetahuan yang dimuat ke dalam sistem pakar dapat berasal dari seorang pakar atau pun pengetahuan yang berasal dari buku, jurnal, majalah, dan dokumentasi yang dipublikasikan lainnya, serta orang yang memiliki pengetahuan meskipun bukan ahli. Istilah sistem pakar (*expert system*), sering disinonimkan dengan sistem berbasis pengetahuan (*knowledge-based system*) atau sistem pakar berbasis pengetahuan (*knowledge-based expert system*).

Gambar di bawah ini mengilustrasikan konsep dasar sistem pakar berbasis pengetahuan (*knowledge-based expert system*). *User* memberikan informasi atau fakta kepada sistem dan menerima respon berupa saran ahli (*advice/expertise*). Secara internal, sistem terdiri dari dua komponen utama yaitu pengetahuan (*knowledge-based*), berisi pengetahuan yang akan digunakan oleh komponen lainnya yaitu mesin inferensi (*inference engine*) untuk menghasilkan kesimpulan sebagai respon terhadap kueri yang dilakukan *user.*

Gambar 2. 3 Expert System

User

Facts

Expertice

Knowledge-Based

Inference-Engine

Expert System

Pengetahuan yang dimiliki pakar bersifat spesifik dalam satu area masalah (*problem domain*). Area masalah merupakan satu wilayah masalah yang spesifik seperti kedokteran atau pengobatan (*medicine*), keuangan (*finance*), rekayasa (*engineering*) dan lainnya. Pengetahuan pakar untuk memecahkan masalah yang spesifik tersebut dikenal sebagai area pengetahuan (*knowledge domain*). Gambar berikut ini menggambarkan hubungan antara area masalah (*problem domain*) dengan area pengetahuan (*knowledge domain*). Area pengetahuan seluruhnya berada dalam area masalah. Bagian yang berada di luar area pengetahuan menyatakan pengetahuan mengenai masalah uang tidak dimiliki oleh sistem. Sebuah sistem pakar umumnya tidak memiliki pengetahuan lain di luar area pengetahuannya kecuali jika diprogram dan dimuat ke dalam sistem. Misalkan sebuah sistem pakar yang memuat pengetahuan mengenai penyakit infeksi mungkin tidak memiliki pengetahuan lain dalam area masalah kedokteran. Dalam area pengetahuan yang dimiliki, sistem pakar melakukan inferensi atau membuat kesimpulan dengan cara yang sama seperti seorang pakar menarik kesimpulan.

Problem Domain

Knowledge  
Domain

Gambar 2. 4 Problem & Knowledge Domain

Sistem pakar memiliki beberapa kelebihan seperti:

* + Meningkatkan ketersediaan (*increased availability*). Kepakaran atau keahlian menjadi tersedia dalam sistem computer. Dapat dikatakan bahwa sistem pakar merupakan produksi kepakaran secara massal.
  + Mengurangi biaya (*reduced cost*). Biaya yang diperlukan untuk menyediakan keahlian per satu orang *user* menjadi berkurang.
  + Mengurangi bahaya (*reduced danger*). Sistem pakar dan pengetahuan yang terdapat di dalamnya bersifat lebih permanen dibandingkan manusia yang dapat merasa lelah, bosan, dan pengetahuannya hilang saat sang pakar meninggal dunia.
  + Keahlian multipel (*multiple expertise*). Pengetahuan dari beberapa pakar dapat dimuat ke dalam sistem dan bekerja secara simultan dan kontinyu menyelesaikan suatu masalah setiap saat. Tingkat keahlian atau pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan satu orang pakar.
  + Meningkatkan kehandalan (*increased reliability*). Sistem pakar meningkatkan kepercayaan dengan memberikan hasil yang benar sebagai alternative pendapat dari seorang pakar atau sebagai penengah jika terjadi konflik antara beberapa pakar. Namun hal tersebut tidak berlaku jika sistem dibuat oleh salah seorang pakar, sehingga akan selalu sama dengan pendapat pakar tersebut kecuali jika sang pakar melakukan kesalahan yang mungkin terjadi pada saat tertekan atau stress.
  + Penjelasan (explanation). Sistem pakar dapat menjelaskan detail proses penalaran (*reasoning*) yang dilakukan hingga mencapai suatu kesimpulan. Seorang pakar mungkin saja terlalu Lelah, tidak bersedia atau tidak mampu melakukannya bahwa kesimpulan yang dihasilkan adalah benar.
  + Respon yang cepat (*fast response*). Respon yang cepat atau *real time* diperlukan pada beberapa aplikasi. Meskipun bergantung pada hardware dan software yang digunakan, namun sistem pakar relative memberikan respon yang lebih cepat dibandingkan seprang pakar.
  + Stabil, tidak emosional, dan memberikan respon yang lengkap setiap saat (*steady, unemotional, and complete response at all* times). Karakteristik ini diperlukan pada situasi *real-time* dan keadaan darurat (*emergency*) Ketika seorang pakar mungkin tidak berada pada kondisi puncak disebabkan oleh stres atau kelelahan.
  + Pembimbing pintar (*intelligent tutor*). Sistem pakar dapat berperan sebagai *intelligent tutor* dengan memberikan kesempatan pada *user* untuk menjalankan contoh program dan menjelaskan proses *reasoning* yang dilakukan.
  + Basis data cerdas (*intelligent database*). Sistem pakar dapat digunakan untuk mengakses basis data secara cerdas.

Sistem pakar tidak lepas dari elemen manusia yang terkait di dalamnya. Personil yang terkait dengan sistem pakar ada 4, yaitu:

1. Pakar (*expert*)

Pakar adalah seorang individu yang memiliki pengetahuan khusus, pemahaman, pengalaman, dan metode-metode yang digunakan untuk memecahkan persoalan dalam bidang tertentu. Seorang pakar memiliki kemampuan kepakaran seperti dapat mengenali dan merumuskan suatu masalah, menyelesaikan masalah dengan cepat dan tepat, menjelaskan solusi dari suatu masalah, restrukturisasi pengetahuan, belajar dari pengalaman, dan memahami batas kemampuan. Selain itu, pakar juga memiliki kemampuan untuk mengaplikasikan pengetahunya dan memberikan saran serta pemecahan masalah pada domain tertentu. Ini merupakan pekerjaan pakar, memberikan pengetahuan tentang bagaimana seseorang melaksanakan tugas untuk menyelesaikan masalah. Penyelesaian masalah ini didukung atau bahkan secara ekstrim akan dilaksanakan oleh sistem berbasis pengetahuan (sistem pakar). Seorang pakar mengetahui fakta-fakta mana yang penting, sebab akibat, fenomena-fenomena yang terkait dengan fakta, memahami arti hubungan antar fakta, juga hubungan sebab akibat, dan hubungan dengan fenomena-fenomena yang terkait serta mampu menginterpretasikan akibat-akibat yang terjadi karena sesuatu sebab terjadi.

1. Pembangun pengetahuan (*knowledge engineer*)

Pembangun pengetahuan memiliki tugas utama menejermahkan dan merepresentasikan pengetahuan yang diperoleh dari pakar, baik berupa pengalaman pakar dalam menyelesaikan malasah maupun sumber terdokumentasi lainnya ke dalam bentuk yang bisa diterima oleh sistem pakar. Dalam hal ini pembangun pengetahuan (*knowledge engineer*) menginterpretasikan dan merepresentasikan pengetahuan yang diperoleh dalam bentuk jawaban-jawaban atas pertanyaan-pertanyaan yang diajukan pada pakar atau pemahaman, penggambaran analogis, sistematis, konseptual yang diperoleh dari membaca beberapa dokumen cetak seperti *text book,* jurnal, makalah, dan sebagainya, kurangnya pengalaman *knowledge engineer* merupakan kesulitan utama dalam mengkontruksi sistem pakar. Untuk mengatasi hal tersebut, perancang sistem pakar menggunakan *tools* komersial. (Seperti pada editor-editor khusus maupun *logic debuggers*) dan usahanya akan dipusatkan pada pembangunan mesin inferensi.

1. Pembangun sistem (*system engineer*)

Pembangun sistem adalah orang yang bertugas untuk merancang antarmuka pemakai sistem pakar, merancang pengetahuan ke dalam bentuk yang sesuai dan dapat diterima oleh sistem pakar dan mengimplementasikannya ke dalam mesin inferensi. Selain hal tersebut, pembangun sistem juga bertanggung jawab apabila sistem pakar akan diintegrasikan dengan sistem komputerisasi lain. Alat pembangun (*tools builder*) dapat dipakai untuk menyajikan atau membangun *tool* yang spesifik. Penjual (*vendor*) dapat memberikan *tool* dan saran, staf pendukung dapat memberikan saran dan bantuan secara teknis dalam proses pembangunan sistem pakar.

1. Pengguna (*user*)

Pengguna adalah orang yang akan menggunakan sistem pakar. Pengguna mungkin tidak terbiasa dengan computer dan mungkin pada domain masalah. Bagaimanapun juga, banyak solusi permasalahan menjadi lebih baik dan kemungkinan lebih murah dan keputusan yang cepat bila menggunakan sistem pakar. Pakar dan pembangun sistem harus mengantisipasi kebutuhan-kebutuhan pengguna dan membuat batasan-batasan Ketika mendesain sistem pakar.

* 1. Request Per Second (RPS)

Requests Per Second, RPS, atau r / s adalah ukuran skalabilitas yang mencirikan throughput yang ditangani oleh suatu sistem.

Menggunakan RPS ketika menentukan ukuran dan skala pengujian beban adalah umum ketika menguji API, dengan cara yang sama bahwa jumlah pengunjung dan pengguna virtual (*virtual user)* bersamaan adalah konsep dan metrik yang paling umum ketika berbicara tentang situs web atau aplikasi.

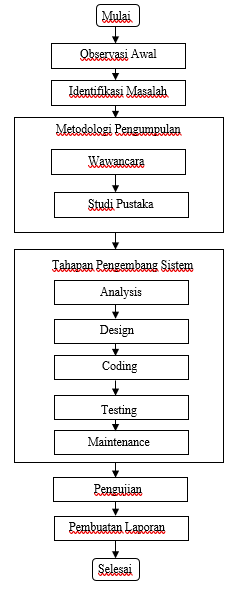
Dalam versi LoadImpact saat ini, ukuran tes beban ditentukan oleh VU. Mengubah skala atau ukuran tes beban dari VU ke RPS tergantung pada sejumlah faktor seperti waktu respons, kompleksitas pemrosesan skrip, dan faktor konkurensi VU. Saat menguji titik akhir API tunggal, biasanya lebih logis untuk berpikir dalam hal permintaan per detik daripada pengguna secara bersamaan. Untuk alasan ini, kami telah mengembangkan skrip sampel yang dapat digunakan untuk memaksimalkan efisiensi Pengguna Virtual saat menguji titik akhir API. Kami memanfaatkan kemampuan Pengguna Virtual untuk dapat membuka banyak koneksi secara paralel.

* 1. Execute time

Execute time adalah waktu yang diperlukan untuk mengeksekusi skrip kode. Dengan mengetahui berapa waktu yang dibutuhkan maka developer dapat dapat mengetahui skrip code yang ditulisnya cepat atau lambat. Sehingga dapat dibandingkan dan diganti dengan skrip yang lain yang lebih cepat dalam proses eksekusinya.

# BAB III METODOLOGI PENELITIAN

1. Kerangka Pikir

Dalam penelitian ini, penulis melakukan tahapan sesuai dengan rencana kegiatan yang telah dibuat meliputi metodologi pengumpulan data, pembuatan aplikasi dan pengujian framework.

Gambar 3. 1 Kerangka Pikir

1. Deskripsi Teori
   1. Observasi Awal

Observasi awal penulis melakukan pengamatan melalui internet. Penulis mengamati terkait perbedaan suatu Framework dengan Framework yang lainnya.

* 1. Identifikasi Masalah

Penulis melakukan identifikasi masalah mengenai perbandingan Framework dengan Framework lainnya.

* 1. Metodologi Pengumpulan Data

Penelitian ini memerlukan data dan informasi untuk mendukung kebenaran dalam pembahasannya. Metodologi pengumpulan yang dilakukan adalah sebagai berikut:

Wawancara

Pengumpulan data terkait dengan penelitian ini salah satunya wawancara secara online. Data yang didapatkan adalah masalah terkait pemilihan framework sebagai bahan belajar atau memulai membuat program.

Studi Pustaka

Dalam tahap ini penulis mengumpulkan dan membaca referensi yang berasal dari buku-buku dan jurnal-jurnal yang berhubungan dengan topik penelitian.

* 1. Pengembangan Sistem

Dalam pengembangan sistem penulis membuat 2 aplikasi. Pertama dengan menggunakan Framework Laravel dan kedua menggunakan Framework Codeigniter. Penulis juga menggunakan metode Forward Chaining untuk metode Sistem Pendukung Keputusannya.

1. *Analysis*

Pada tahap ini penulis melakukan analisis sistem dan analisis kebutuhan. Berikut tahapan analisis yang dilakukan.

1. Analisis Kebutuhan
2. Kebutuhan *software*:

* XAMPP
* Sublime Text
* Command Prompt / Git
* Web Browser
* Apache Becnhmark (ab)
* Framework Laravel
* Framework Codeigniter

1. Kebutuhan Fungsional:
   * + - * Data awal untuk kebutuhan database yang didapat dari jurnal yang telah dicantumkan
2. Kebutuhan Hardware

Laptop Lenovo ideapad 320

* Processor: AMD A9-9420 RADEON R5, 5 COMPUTE CORES 2C+3G 3,00GHz
* RAM: 4 GB
* Tipe sistem: 64-bit Operating System, x64 based processor

1. *Desain*

Perancangan dibuat dengan *Unified Modeling Language* (UML) agar dengan mudah dalam proses pengembangan dan visualisasinya.

1. *Coding*

Aplikasi dibuat dengan menggunakan Framework Laravel dan Framewok Codeigniter.

1. *Testing*

Pengujian terhadap aplikasi yang dibuat dengan cara mengecek apakah semua fungsi telah berjalan sesuai dengan yang telah ditentukan.

1. *Maintenance*

Apabila aplikasi tersebut sudah layak maka aplikasi tersebut siap untuk digunakan untuk tahap berikutnya yaitu pengujian Framework.

* 1. Pengujian Framework

1. Pengujian Performa

Pengujian performa dilakukan dengan membandingkan Framework Laravel dengan Framework Codeigniter dari beberapa faktor antara lain:

* Pengujian Request Per Second (RPS)
* Pengujian Time Per Request
* Pengujian Execute Time (Waktu Eksekusi)

1. Pengujian Ukuran

Analisis ukuran dilakukan dengan membandingkan dan menganalisis struktur direktori dan besarnya total file yang terdapat pada direktori.

1. Pengujian Cara Akses Database

Cara akses database dilakukan dengan membandingkan dan menganalisis akses database untuk tata cara pengaksesan tabel dalam operasi Create, Read, Update, Delete (CRUD). Cara pengaksesan database di Laravel menggunakan Eloquent ORM dan cara pengaksesan database di CodeIgniter menggunakan Query Builder.

* 1. Pembuatan Laporan

Pada tahap ini merupakan tahap terakhir dalam melakukan penelitian yang terdiri dari 6 bab. Berikut adalah sistematika penulisan :

BAB I : PENDAHULUAN

Bab ini menjelaskan latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian, dan sistematika penulisan.

BAB II : TINJUAN PUSTAKA

Bab ini menjelaskan tentang landasan teori dan dasar teori pendukung dalam penelitian. Studi pustaka ini bersumber dari jurnal, tesis, buku teks, dan website.

BAB III : METODOLOGI PENELITIAN

Bab ini menjelaskan tentang gambaran kerangka pikir dan definisi dari gambaran kerangka pikir tersebut.

BAB IV : ANALISIS DAN PERANCANGAN

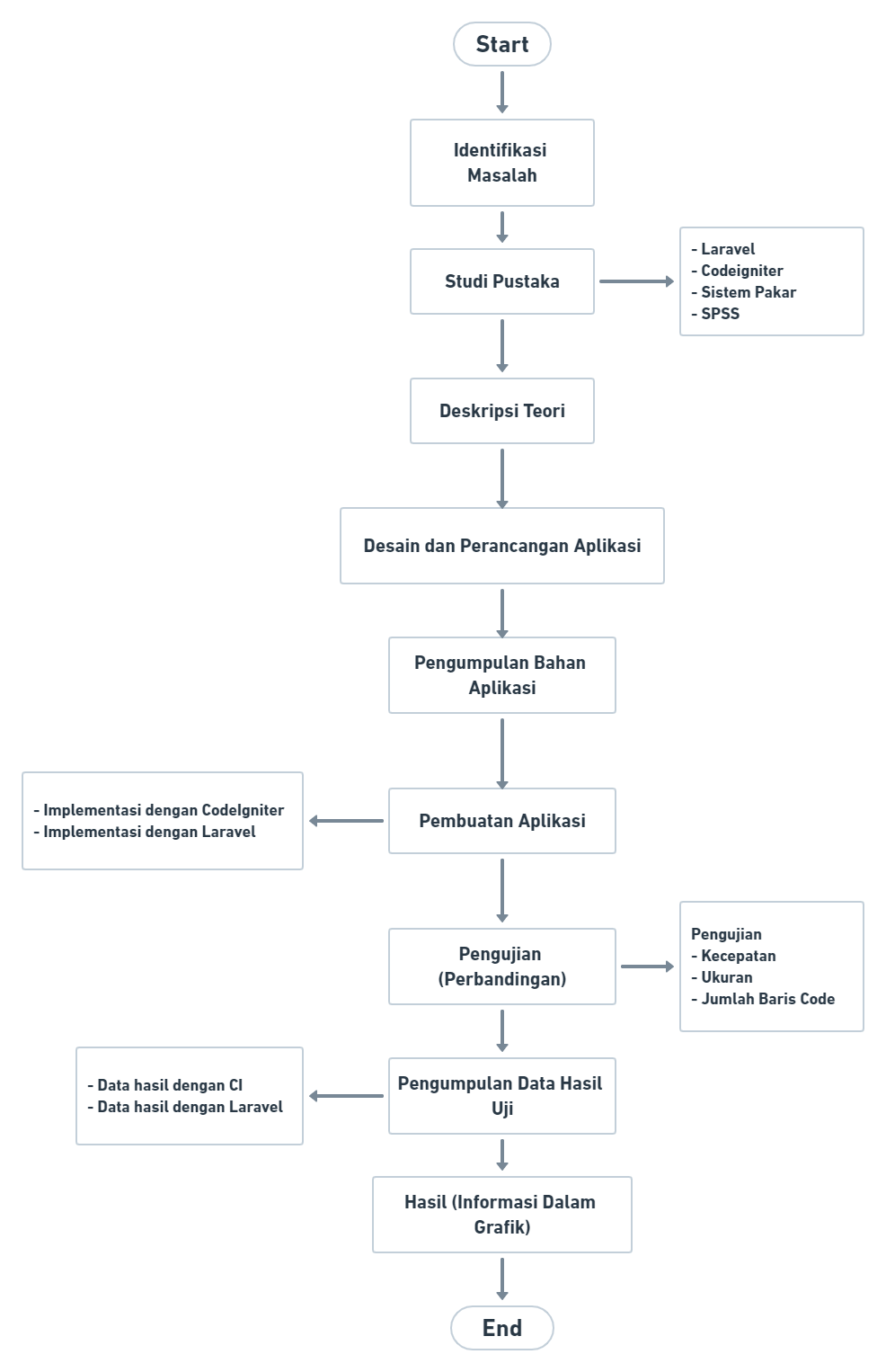
Bab ini menjelaskan tentang analisis dan perancangan dari aplikasi yang akan dibuat

BAB V : IMPLEMENTASI DAN PENGUJIAN

Bab ini menjelaskan tentang implementasi dan pengujian dari aplikasi yang sudah dibuat.

BAB VI : KESIMPULAN DAN SARAN

Bab ini berisi penarikan kesimpulam penelitian dan pengembangan yang dilakukan serta saran untuk mengembangkan aplikasi selanjutnya.



# BAB IV ANALISIS DAN PERANCANGAN

1. Analisis
   1. Analisis Masalah
2. Bagaimana mendapatkan hasil perbandingan Framework Laravel dan Framework Codeigniter dari segi kecepatan/*performa*?

* Performa diambil dari pengukuran *request per second, time per second* dan *execute time.*
* *Request per second dan time per request* diambil dengan *tool* Apache Benchmark (ab).
* *Execute time* diambil dengan menggunakan fungsi yang sudah tersedia pada PHP. Pengujian dilakukan dengan menambahkan fungsi pada awal skrip kode dan akhir skrip kode. Maka akan didapat waktu *execute time* dari tiap fitur yang tersedia.

1. Bagaimana mendapatkan hasil perbandingan Framework Laravel dan Framework Codeigniter dari segi ukuran/*resource*?

* Analisis ukuran dilakukan dengan membandingkan dan menganalisis struktur direktori dan besarnya total file yang terdapat pada direktori. Kebanyakan framework PHP yang menganut pola MVC (Model-View-Controller) menggunakan skema direktori dengan nama "Model", "View" dan "Controller" yang seluruhnya dikumpulkan kedalam sebuah direktori utama yang bernama "src" atau "app" / "application" seperti Codeigniter. Sedikit berbeda dengan Codeigniter, direktori dengan nama “View” pada Laravel justru diletakkan di luar direktori “app”. Direktori “View” digunakan untuk menyimpan file-file yang berhubungan dengan tampilan aplikasi .Lalu pada direktori “app” baik pada Framework Laravel atau Framewok Codeigniter terdapat direktori “Model” yang digunakan untuk menyimpan class PHP yang berhubungan dengan model database. Kemudian direktori “Controller” digunakan untuk menyimpan class PHP yang berhubungan dengan *application logic.*

1. Bagaimana mendapatkan hasil perbandingan Framework Laravel dan Framework Codeigniter dari segi cara akses database?

* Cara akses *database* dilakukan dengan membandingkan dan menganalisis akses *database* untuk tata cara pengaksesan tabel dalam operasi *Create, Read, Update, Delete* (CRUD). Cara pengaksesan *database* di Laravel menggunakan Eloquent ORM, Query Builder, dan Raw Query. Cara pengaksesan *database* di CodeIgniter menggunakan Query Builder dan Query Basics.

1. Bagaimana pembuatan aplikasi diagnosa dengan menggunakan Framework Laravel dan Framework Codeigniter?
   1. Analisis Kebutuhan

Ada beberapa *software* yang digunakan penulis untuk membuat skripsi diantaranya :

* XAMPP
* Sublime Text
* Command Prompt / Git
* Web Browser
* Apache Becnhmark (ab)
* Framework Laravel
* Framework Codeigniter

Kebutuhan data(*database*/*files*) yang digunakan penulis untuk membuat skripsi ini adalah sebagai berikut :

* + - * + Data awal untuk kebutuhan database yang didapat dari jurnal yang telah dicantumkan

Kebutukan *hardware* (Laptop) yang digunakan penulis untuk membuat skripsi ini adalah sebagai berikut :

|  |  |
| --- | --- |
| Processor | AMD A9-9420 RADEON R5, 5 COMPUTE CORES 2C+3G 3,00GHz |
| RAM | 4,00 GB |
| System Type | 64-bit Operating System, x64 based processor |

Tabel 4. 1 Spesifikasi Hardware

* 1. Analisis Pengguna

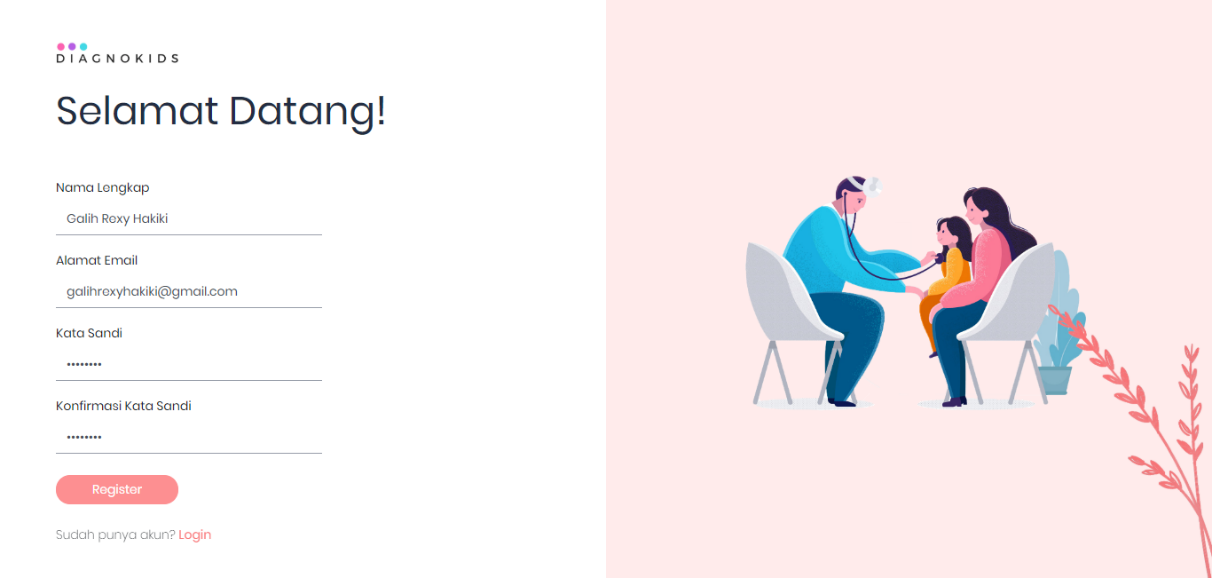
Target pengguna dari aplikasi ini merupakan para orang tua yang memiliki anak yang berusia kurang dari 5 tahun. Untuk membuat para orang tua merasa nyaman dan meyakinkan mereka bahwa aplikasi ini benar untuk diagnosa anak maka tampilan dibuat penuh warna dan terdapat gambar ilustrasi yang berkaitan dengan kedokteran.

* 1. User Interface

User Interface pada aplikasi dirancang dengan memberikan pewarnaan yang menarik dan menambahkan beberapa illustrasi. User interface juga dibuat sederhana supaya pengguna mudah dalam mengoprasikannya. Dalam merancang. *User Interface* penulis mengumpulkan referensi desain website pada sebuah web yang bernama dribbble[.](https://dribbble.com/) Berikut merupakan beberapa referensi desain website yang penulis gunakan untuk pembuatan aplikasi diagnosa penyakit pada anak ini.

Gambar 4. 1 Referensi Desain

Referensi desain ini merupakan desain untuk halaman login dan halaman register. Maka dari itu penulis juga jadikan referensi ini sebagai referensi dalam pembuatan halaman registrasi dan halaman login.

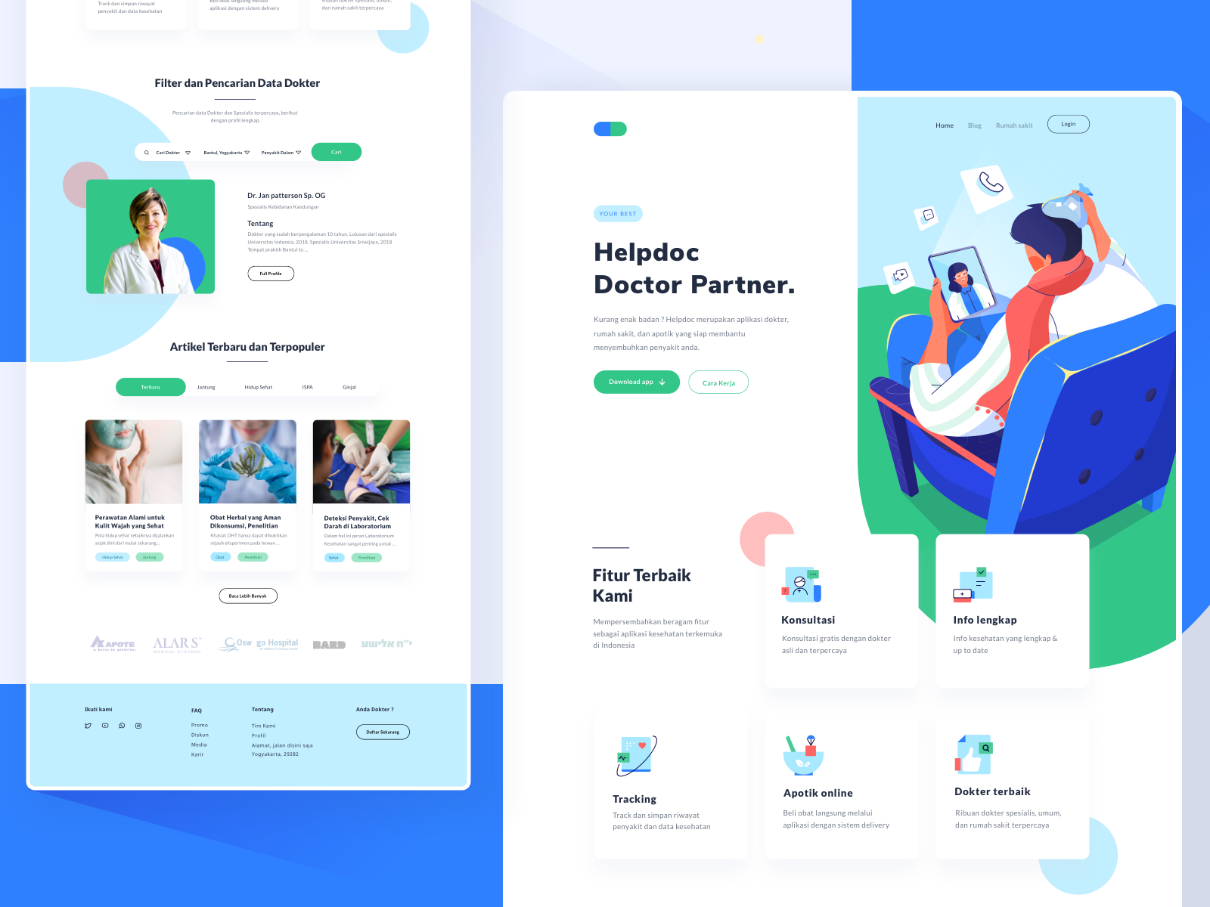
Gambar di atas merupakan hasil dari perancangan yang dibuat dengan menggunakan referensi desain website. Lalu ada referensi desain website yang penulis gunakan sebagai referensi dalam pemilihan skema warna dan tata letak untuk desain halaman home, halaman profil, halaman diagnosa dan halaman data diagnosa.

Gambar 4. 2 Rancangan Register

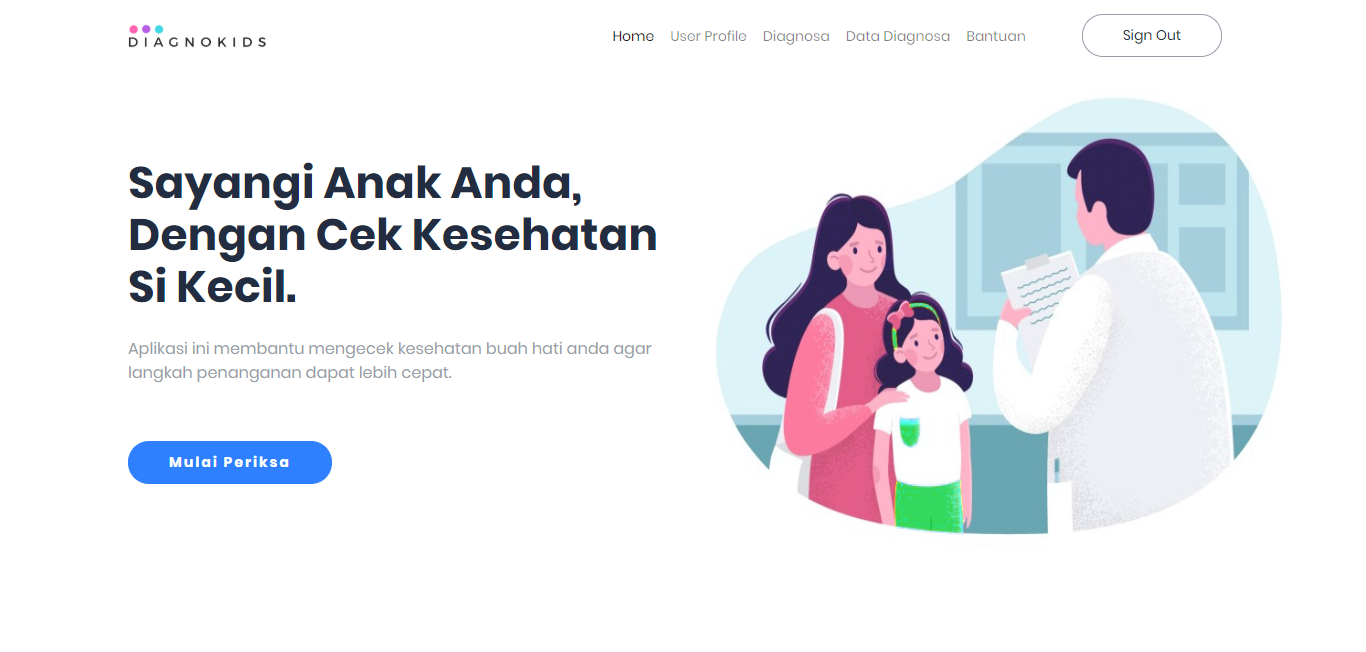
Gambar 4. 3 Rancangan Login



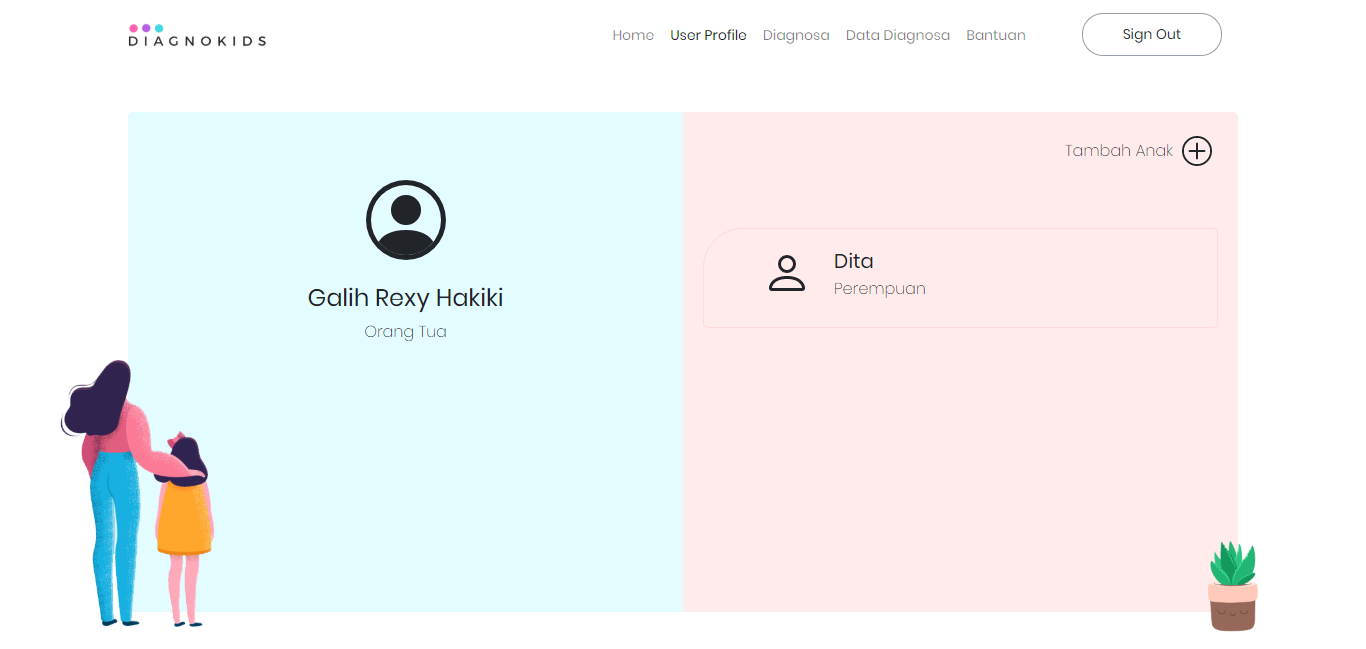
Gambar 4. 4 Referensi Desain 3



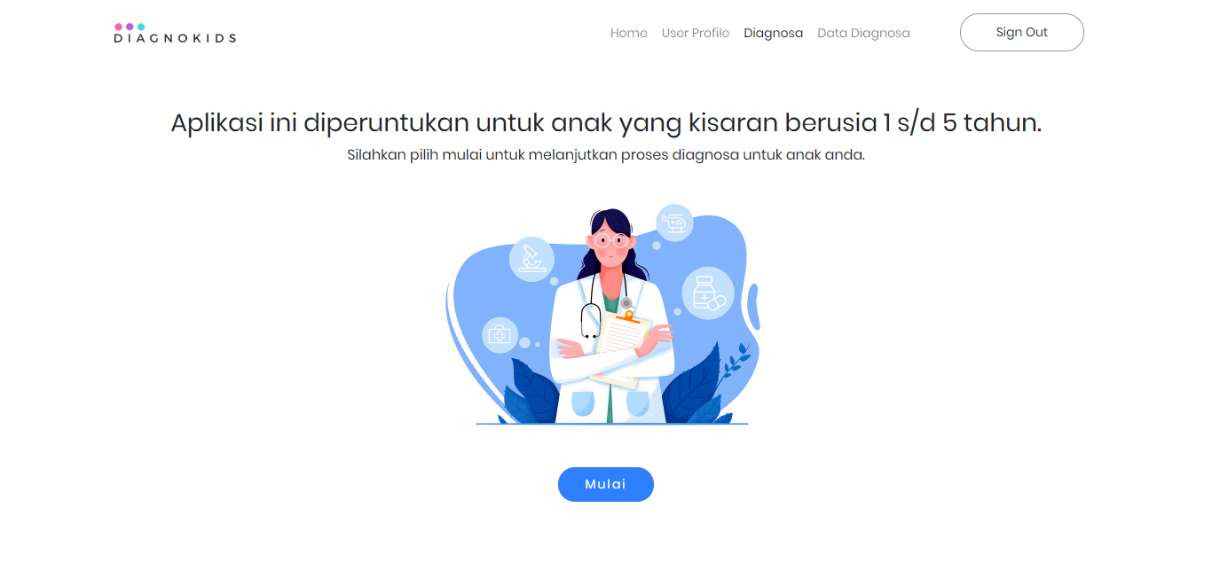
Gambar 4. 5 Referensi Desain 2

Kedua referensi desain website inilah yang penulis gunakan sebagai referensi dalam pemilihan skema warna dan tata letak untuk desain halaman home, halaman profil, halaman diagnosa dan halaman data diagnosa. Dan berikut merupakan hasil perancangannya.

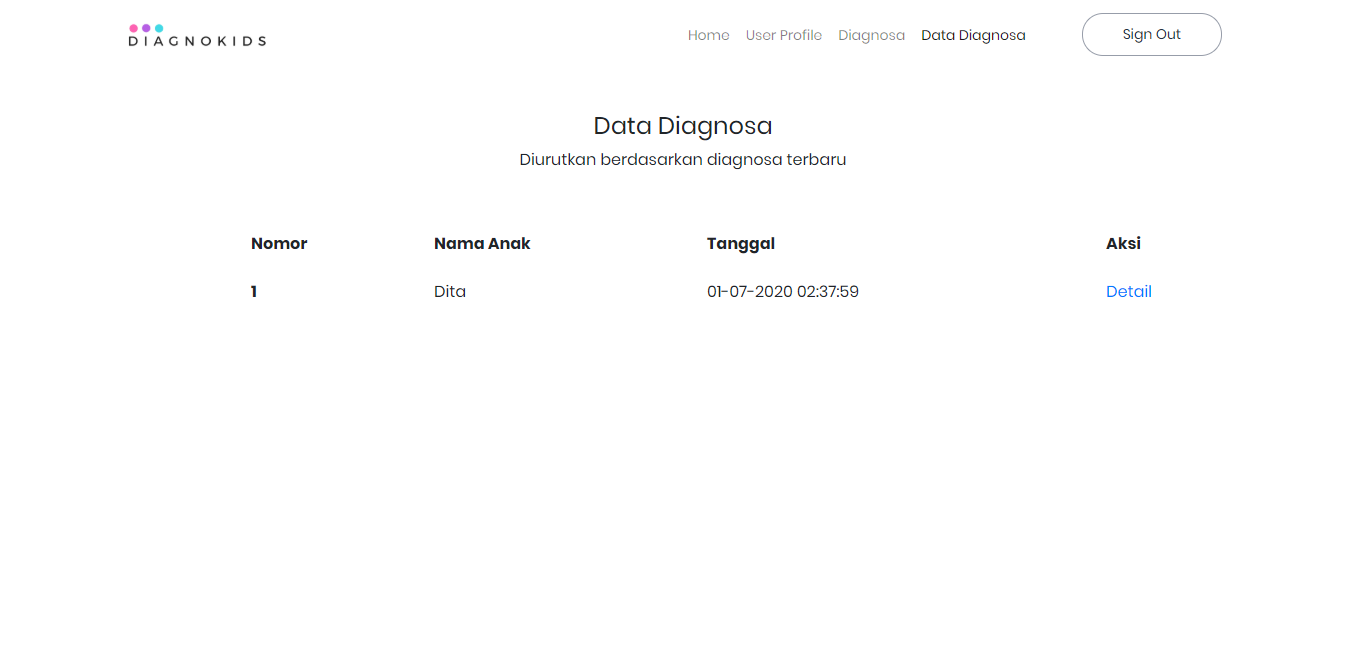
Gambar 4. 6 Perancangan Halaman Home

Gambar merupakan hasil rancangan untuk halaman home. Sesuai dengan referensi desain di atas. Pada halaman terdapat navbar yang pada bagian kirinya terdapat logo aplikasi dan pada sebelah kanan terdapat menu .aplikasi. Terdapat pula ilustrasi, tulisan dan tombol mulai diagnosa untuk menjalankan fitur diagnosa anak.

Gambar 4. 7 Perancangan Halaman Profil

Gambar merupakan hasil rancangan untuk halaman profil. Pada halaman terdapat navbar yang pada bagian kirinya terdapat logo aplikasi dan pada sebelah kanan terdapat menu .aplikasi. Terdapat pula ilustrasi, data pengguna, data anak (jika sudah ditambahkan) dan tombol tambah anak untuk menjalankan fitur tambah data anak.

Gambar 4. 8 Perancangan Halaman Diagnosa

Gambar merupakan hasil rancangan untuk halaman diagnosa. Terdapat ilustrasi, tulisan dan tombol mulai untuk menjalankan fitur diagnosa penyakit pada anak.

Gambar 4. 9 Perancangan Halaman Data Diagnosa

Gambar merupakan hasil rancangan untuk halaman diagnosa. Terdapat data hasil diagnosa yang telah tersimpan. Terdapat tombol detail untuk melihat detail dari hasil diagnosa.

* 1. Fitur-Fitur

Dalam aplikasi yang akan dibuat terdapat beberapa fitur yang dapat dijalankan. Fitur tersebut dibuat friendly user sehingga pengguna mudah dalam menjalankan setiap fitur yang ada. Fitur-fitur tersebut antara lain:

1. Aplikasi diagnosa dapat menambahkan menampilkan data profil pengguna dan data anak dari pengguna. Serta dapat menambahkan data anak apabila data anak belum ditambahkan.
2. Aplikasi dapat mendiagnossa penyakit pada anak pengguna yang terlah ditambahkan ke dalam sistem. Apabila data anak belum ada maka pengguna akan diarahkan untuk menambahkan data anak terlebih dahulu.
3. Aplikasi dapat menampilkan hasil diagnose yang telah dilakukan. Setelah data ditampilkan pengguna dapat melihat detail hasil diagnose dengan memilih tombol detail pada data diagnose yang ingin dilihat detail diagnosanya.
   1. Analisis Data

Dalam pembuatan aplikasi ini dibutuhkan data untuk membuat aturan dalam mendiagnosa penyakit. Data ini didapatkan dari jurnal yang sudah dicantumkan.

|  |  |
| --- | --- |
| Kode | Keluhan |
| K1 | Batuk |
| K2 | Diare |
| K3 | Deman |

Tabel 4. 2 Data keluhan

|  |  |  |  |
| --- | --- | --- | --- |
| Kode | Klasifikasi Penyakit | Kode | Klasifikasi Penyakit |
| P1 | Tanda Bahaya Umum | P10 | Disentri |
| P2 | Batuk | P11 | Demam |
| P3 | Pneumonia | P12 | Demam dengan Tanda Bahaya Umum |
| P4 | Pneumodia Berat | P13 | Campak |
| P5 | Diare | P14 | Campak dengan Komplikasi Berat |
| P6 | Diare Dehidrasi Ringan | P15 | Campak dengan komplikasi |
| P7 | Diare Dehidrasi Berat | P16 | Demam Mungkin DBD |
| P8 | Diare Persisten | P17 | DBD |
| P9 | Diare Persisten Berat | P18 | Demam bukan DBD |

Tabel 4. 3 Data Penyakit

|  |  |  |  |
| --- | --- | --- | --- |
| Kode | Gejala | Kode | Gejala |
| G1 | Anak tidak bisa minum atau menyusu | G20 | Suhu badan melebihi 37.5º C |
| G2 | Anak memuntahkan makanan yang dimakan | G21 | Kaku kuduk (anak tidak bisa menunduk hingga dagu mencapai dada) |
| G3 | Anak menderita kejang | G22 | Ruam kemerahan di kulit |
| G4 | Anak tampak letargis atau tidak sadar | G23 | batuk pilek atau mata merah |
| G5 | Napas Normal | G24 | Luka di mulut yang dalam atau luas |
| G6 | Napas cepat | G25 | Kekeruhan pada kornea mata |
| G7 | Tarikan dinding dada ke dalam | G26 | Luka di mulut |
| G8 | Stridor | G27 | Mata bernanah |
| G9 | Berak cair atau lembek | G28 | Demam 2 - 7 hari |
| G10 | Mata cekung | G29 | Demam mendadak tinggi dan terus menerus |
| G11 | Cubitan kulit perut kembali lambat | G30 | Nyeri di ulu hati |
| G12 | Gelisah, rewel/mudah marah | G31 | bintik bintik merah |
| G13 | Haus, minum dengan lahap | G32 | Muntah bercampur darah / seperti kopi |
| G14 | Cubitan kulit perut sangat lambat | G33 | Tinja berwarna hitam |
| G15 | Anak tampak letargis atau tidak sadar | G34 | Perdarahan dihidung dan gusi |
| G17 | Tidak bisa minum atau malas minum | G34 | Syok dan gelisah |
| G18 | Diare 14 hari atau lebih | G35 | Infeksi |
| G19 | Ada darah dalam tinja | G36 | Pilek |

Tabel 4. 4 Data gejala

|  |  |  |
| --- | --- | --- |
| Rule | IF | THEN |
| 1 | G1 OR G2 OR G3 OR G4 | P1 |
| 2 | K1 AND G5 | P2 |
| 3 | K1 AND G6 | P3 |
| 4 | K1 AND P1 OR G7 OR G8 | P4 |
| 5 | K2 AND G9 | P5 |
| 6 | P5 AND G10 AND G11 OR G12 OR G13 | P6 |
| 7 | P5 AND G10 AND G14 OR G15 OR G16 | P7 |
| 8 | P5 AND G17 | P8 |
| 9 | P8 AND P6 OR P7 | P9 |
| 10 | P5 AND G18 | P10 |
| 11 | K3 AND G19 | P11 |
| 12 | P1 AND P11 OR G20 | P12 |
| 13 | P11 AND G21 AND G22 OR G25 | P13 |
| 14 | P13 AND P1 AND G23 OR G24 | P14 |
| 15 | P13 AND G25 OR G26 | P15 |
| 16 | P11 AND G27 AND G28 AND G29 OR G30 | P16 |
| 17 | P11 AND G27 AND G28 AND G31 OR G32 OR G33 OR G34 | P17 |
| 18 | P11 AND G35 OR G36 | P18 |

Tabel 4. 5 Data aturan diagnosa

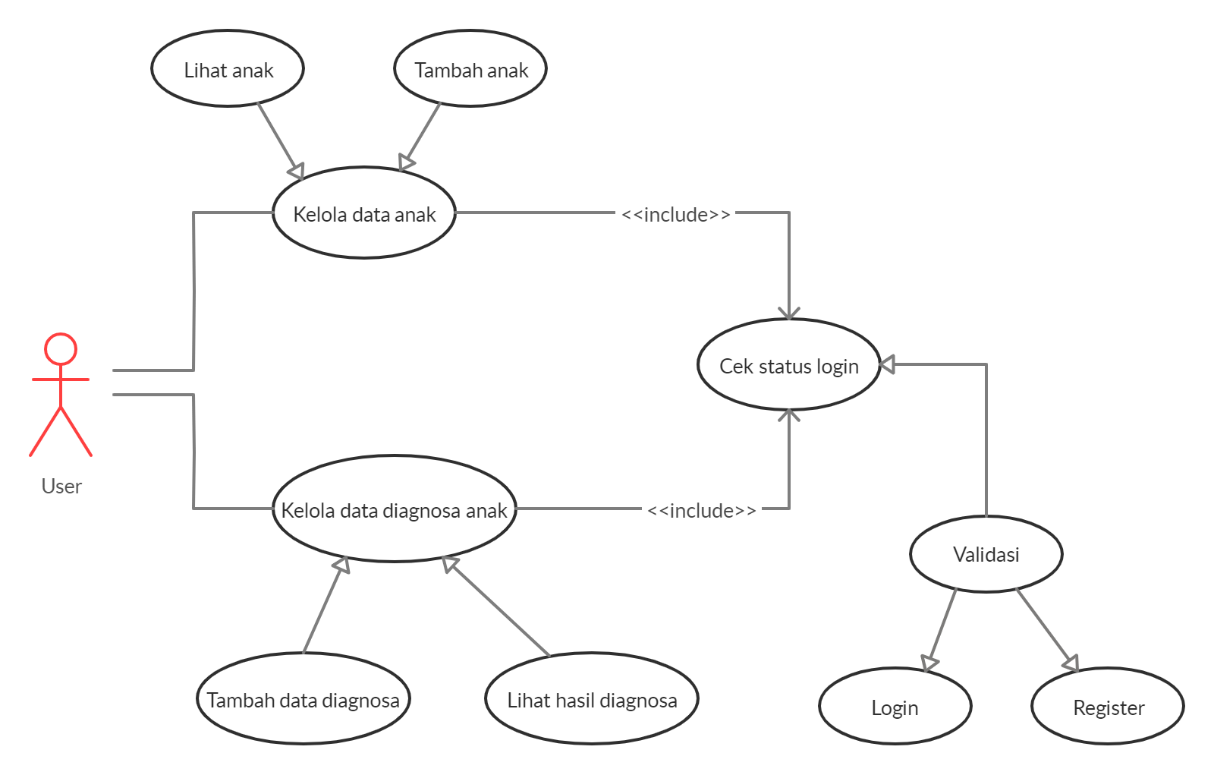
1. Perancangan Aplikasi
   1. Unified Modeling Language (UML)
      1. Use Case Diagram

*Use Case Diagram* adalah gambaran interaksi antara pengguna (*user/actor*) dengan sistem informasi yang akan dibuat. *Use case* menggambarkan siapa saja actor yang terlibat dan funcsi apa saja yang dapat digunakan *actor* pada sistem informasi tersebut.

*Use Case Diagram* terdiri dari:

* *Use case*
* *Actor*
* *Relationship*
* *System Boundary /* batas sistem *(*opsional*)*

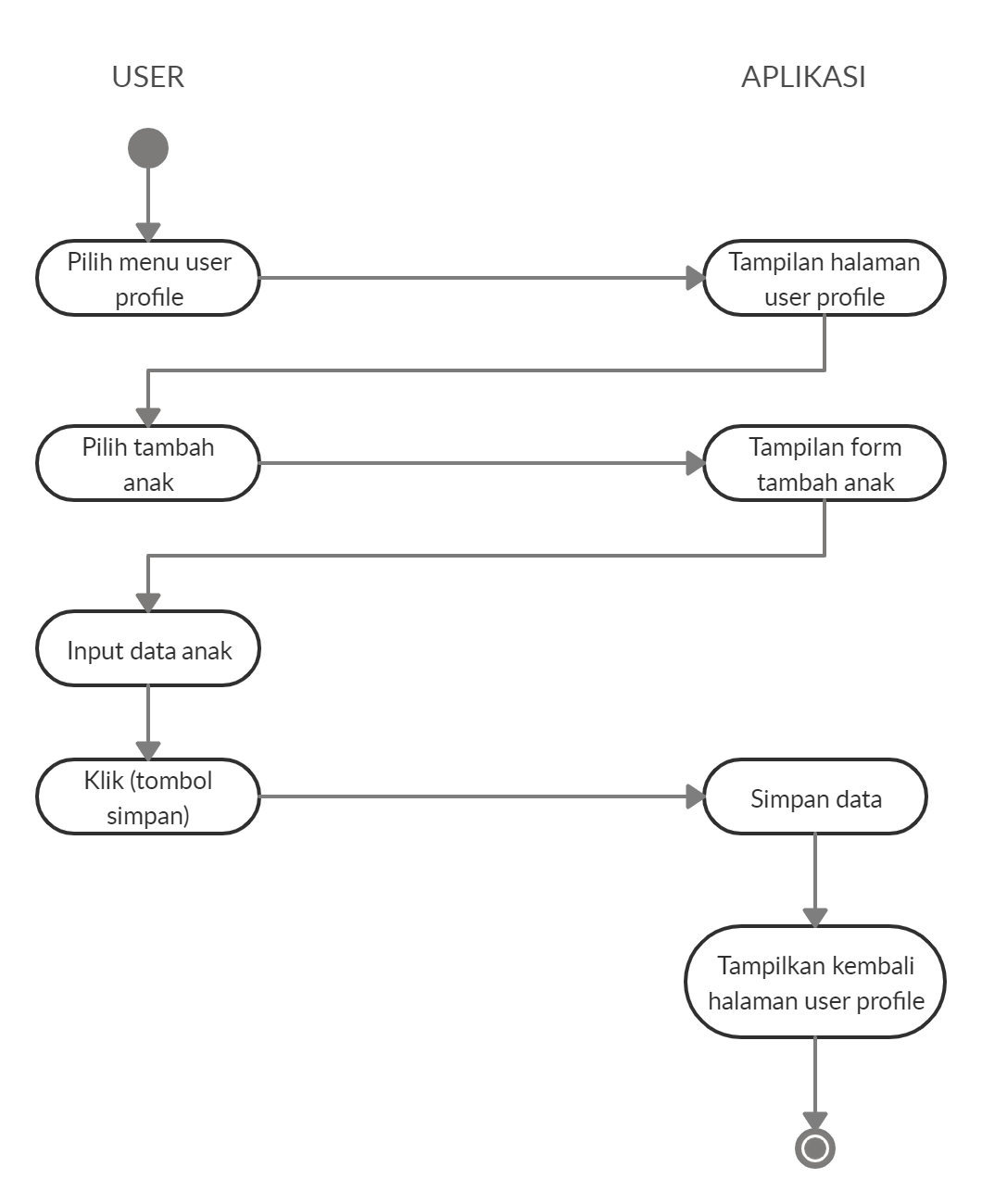
Berikut merupakan use case diagram user untuk aplikasi diagnosa untuk anak dibawah 5 tahun:



Gambar 4. 10 Use case user

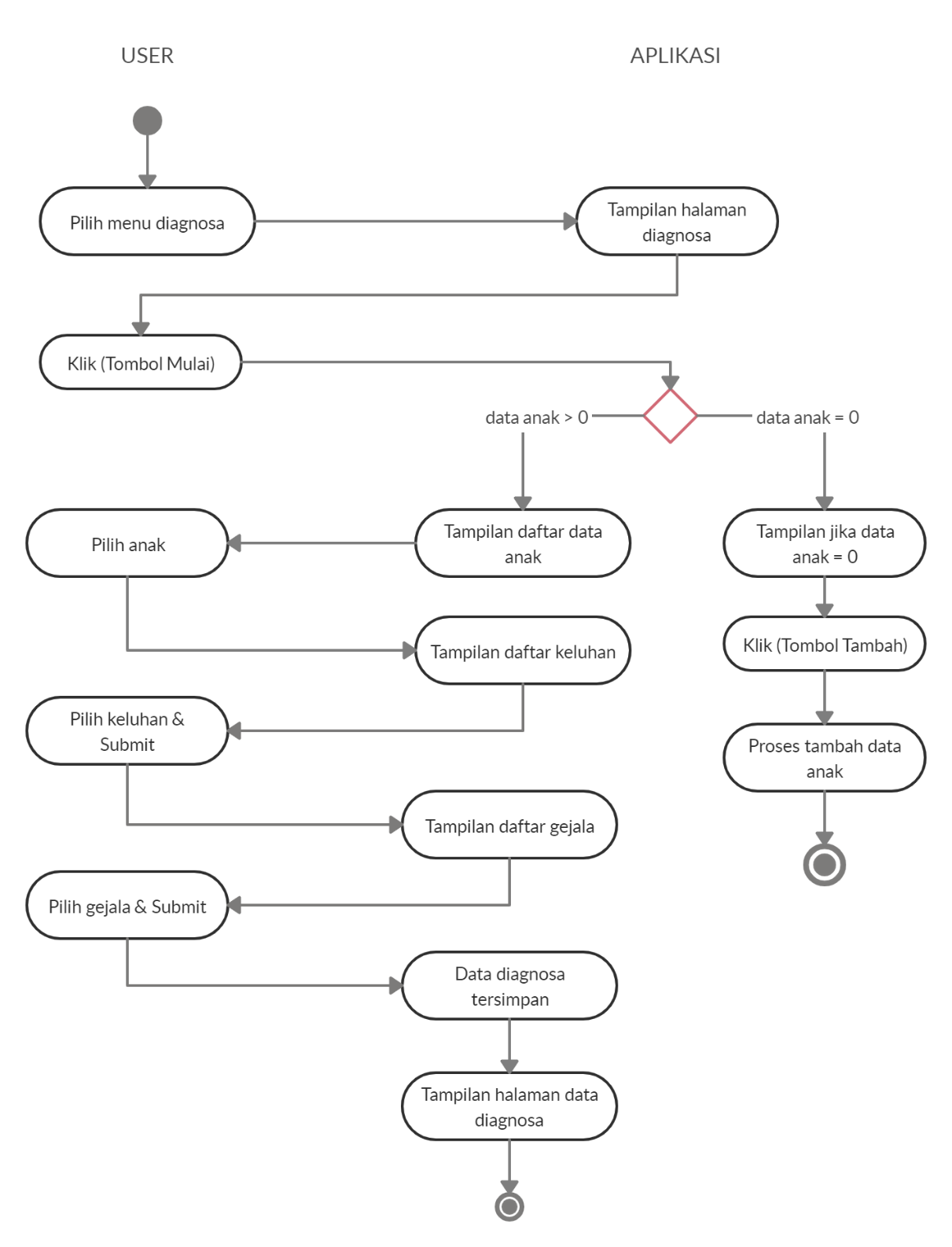
* + 1. Activity Diagram

*Activity diagram* atau diagram aktivitas menggambarkan urutan kegiatan atau urutan aktivitas dari sebuah sistem. Tujuan dibuatnya activity diagram adalah untuk memudahkan dalam memahami proses bisnis sistem. Berikut merupakan *activity diagram* untuk aplikasi diagnosa untuk anak dibawah 5 tahun.



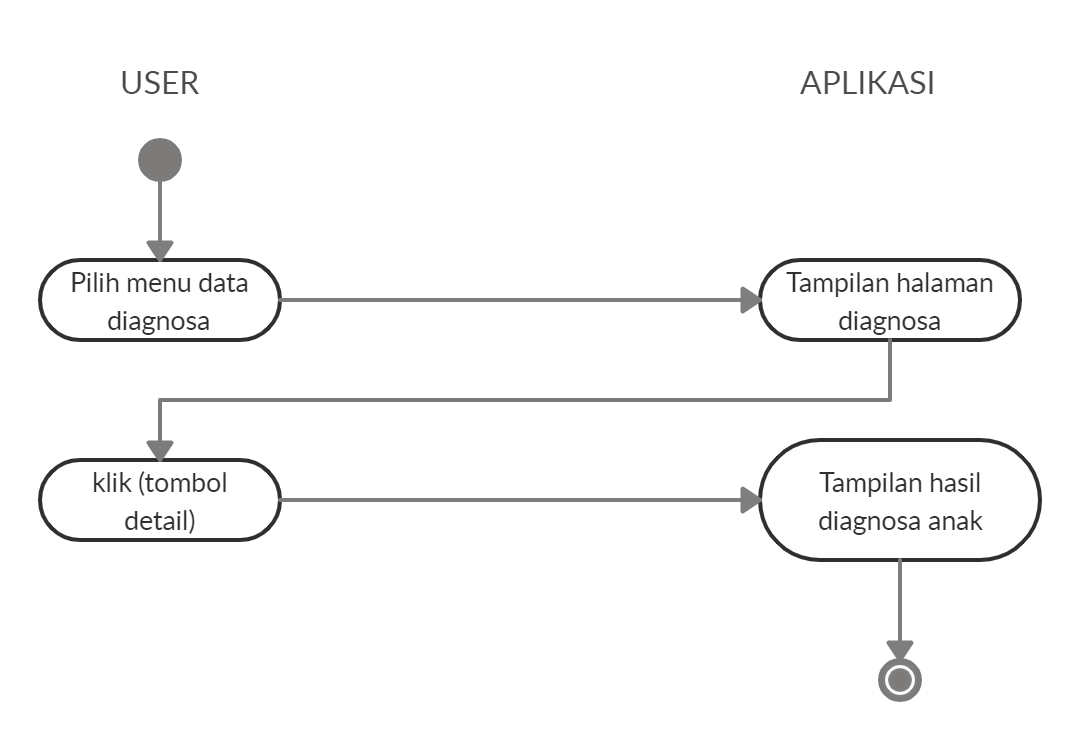
Gambar 4. 11 Activity diagram tambah data anak

Dapat diperhatikan pada gambar 4. 2 dapat dilihat bahwa untuk menambahkan data anak pengguna dapat memilih menu user profile lalu akan tampil halaman user profile, pilih tombol tambah anak lalu akan tampil form input untuk data anak, masukkan data anak pada form tersebut lalu klik tombol simpan, maka data anak berhasil ditambahkan.



Gambar 4. 12 Activity diagram proses diagnosa

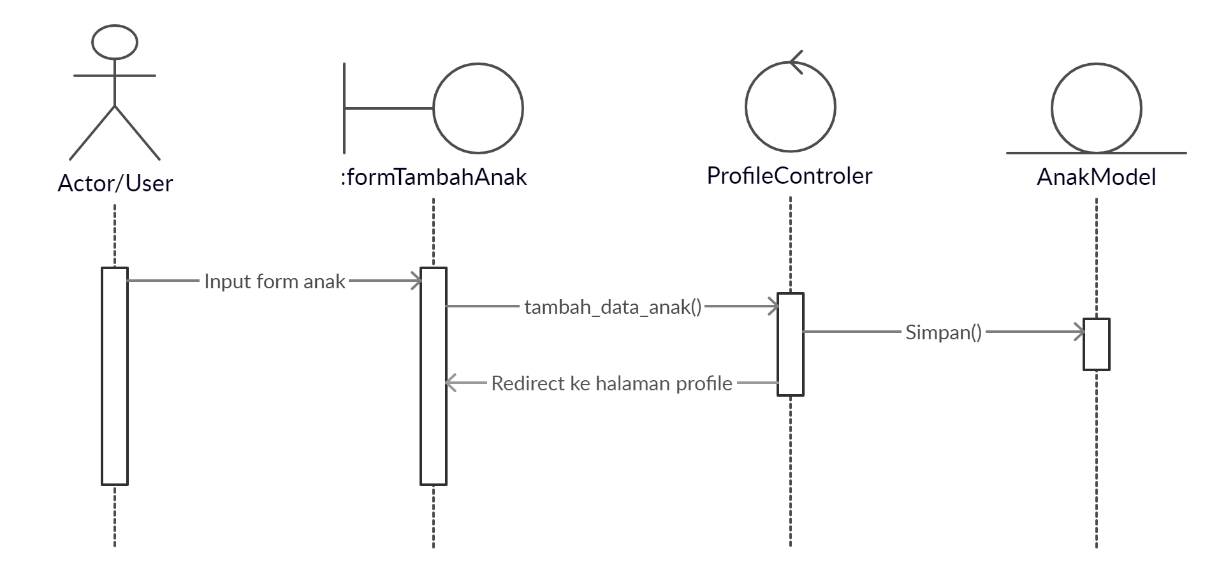
Pada gambar 4.3 dapat dilihat bahwa untuk melakukan diagnosa pengguna dapat memilih menu diagnosa lalu akan tampil halaman diagnosa, pilih mulai maka akan tampil data anak yang sudah pengguna tambahkan dan jika belum ada data anak yang ditambahkan maka akan tampil tombol tambah anak untuk menambahkan data anak terlebih dahulu agar dapat melakukan diagnosa. Lalu setelah muncul data anak pilih pada data anak yang ingin dilakukan diagnosa, maka akan tampil data keluhan, lalu pengguna dapat pilih keluhan yang terdapat pada anaknya dan klik tombol submit. Lalu tampil data gejala maka pengguna dapat memilih semua gejala yang terdapat pada anaknya dan klik tombol submit. Kemudian data diagnosa yang baru saja dilakukan akan tersimpan dan pengguna akan di alihkan ke halaman data diagnosa.



Gambar 4. 13 Activity diagram lihat hasil diagnosa

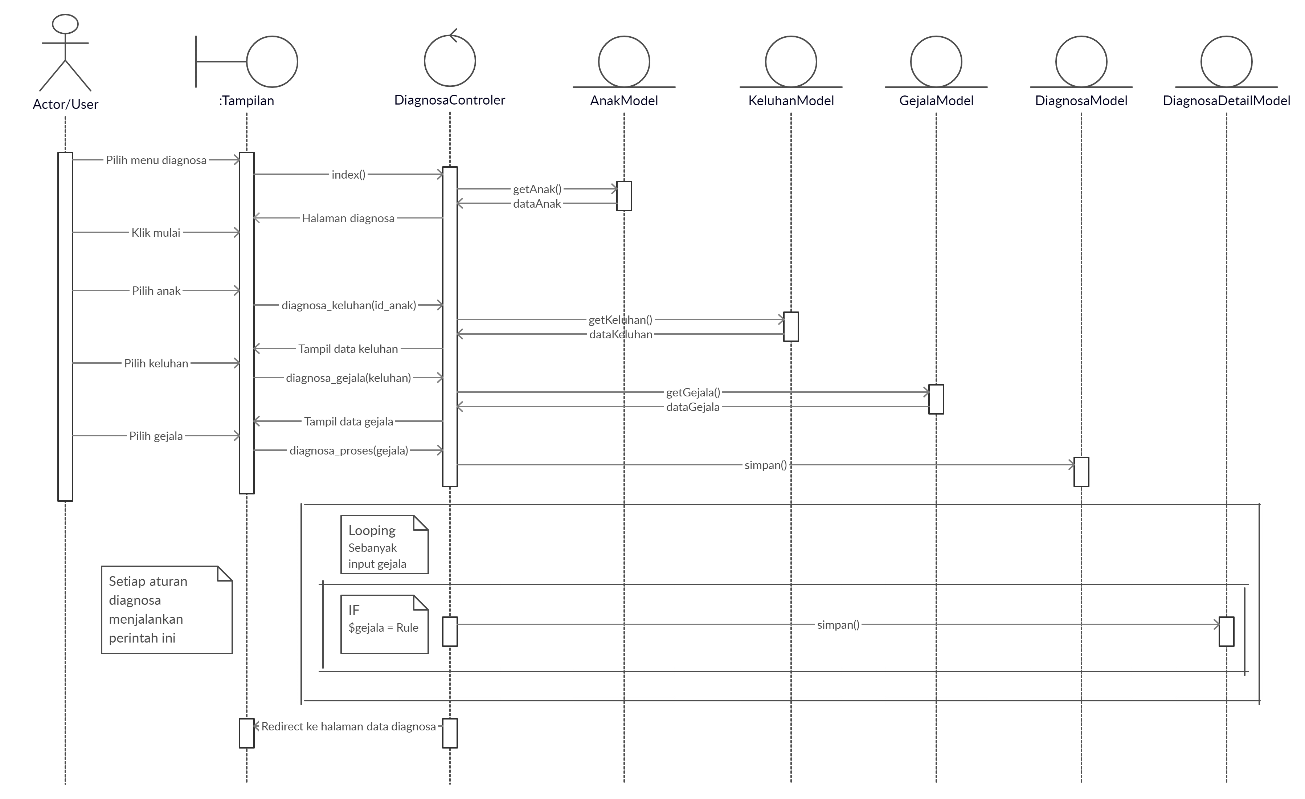
Dapat diperhatikan pada gambar 4.4 dapat dilihat bahwa untuk melihat hasil diagnosa dapat memilih menu data diagnose lalu akan tampil halaman data diagnosa, pilih tombol detail pada anak yang ingin dilihat hasil diagnosanya lalu akan tampil hasil diagnosa pada anak pengguna tersebut.

* + 1. Sequence Diagram

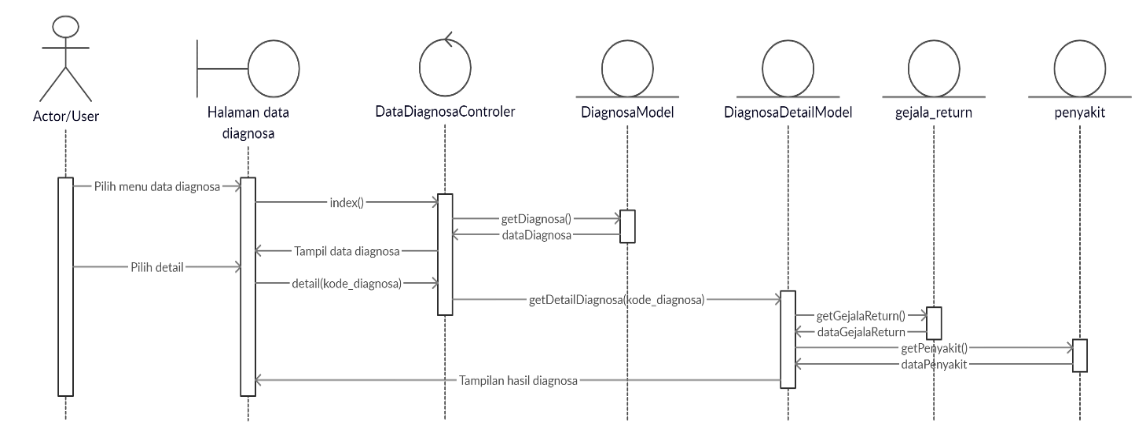
*Sequence diagram* menggambarkan interaksi antar obyek dalam sistem. *Sequence diagram* digunakan untuk menggambarkan scenario pada *use case*.Jumlah *Sequence diagram* harus sama dengan jumlah *use case.*

Gambar 4. 14 Sequence diagram tambah data anak

Dapat diperhatikan pada gambar 4.5 merupakan *sequence diagram* dari proses tambah data anak. Pertama pengguna menginputkan data pada form yang telah disediakan dihalaman user profilelalu klik simpan. Maka data anak yang telah diinputkan akan dikirim ke sebuah fungsi yang bernama tambah\_data\_anak() yang diterdapat disebuah controller yang bernama ProfileController dan data anak tersebut akan disimpan ke dalam sebuah database melalui sebuah model yang bernama AnakModel. Setelah data anak berhasil tersimpan ProfileController akan me-*redirect* ke halaman user profile dan data anak yang sudah tersimpan tadi akan langsung dapat terlihat / ditampilkan.

­

Gambar 4. 15 Sequence diagram proses diagnosa

Dapat diperhatikan pada gambar 4.6 merupakan sequence diagram dari proses diagnosa pada anak. Pertama pilih menu diagnosa maka akan dijalankan suatu fungsi yang bernama index() pada suatu controller yang bernama DiagnosaController. DiagnosaController akan mengambil data anak melalui model yang bernama AnakModel. Lalu DiagnosaController akan me-redirect ­ke halaman diagnosa. Pengguna dapat pilih tombol mulai untuk memulai proses diagnosa. Dan akan langsung ditampilkan data anak yang tadi telah diambil melalui DiagnosaController. Pilih nama anak yang akan dilakukan diagnosa. Maka akan menjalankan fungsi diagnosa\_keluhan(id\_anak) yang terdapat pada DiagnosaController dengan membawa parameter id\_anak yang telah dipilih tadi. DiagnosaController akan mengambil data keluhan pada model KeluhanModel. Lalu data keluhan akan ditampilkan dan pengguna dapat memilih semua keluhan yang sama / terdapat pada anaknya. Fungsi diagnosa\_gejala(kode\_keluhan) yang terdapat pada DiagnosaController dengan membawa kode\_keluhan yang pengguna sudah pilih.DiagnosaController akan mengambil data gejala melalui model GejalaModel. Lalu data gejala akan ditampilkan dan pengguna dapat memilih semua gejala yang sama / terdapat pada anaknya. Setelah pengguna memilih gejala-gejala dan menekan tombol submit maka akan dijalankan fungsi diagnossa\_proses(gejala) pada DiagnosaController dan data diagnosa akan disimpan ke database diagnosa melalui model DiagnosaModel. Lalu DiagnosaController akan menjalankan aturan-aturan / prosses diagnosa untuk menentukan penyakit yang diidap oleh anak berdasarkan keluhan-keluhan dan gejala-gejala yang sudah dipilih Setiap penyakit yang terdiagnosa datanya akan tersimpan ke database diagnosa\_detail melalui DiagnosaDetailModel. Proses akan terus berjalan sampai semua aturan untuk menentukan penyakit yang diidap berhasil dijalankan. Lalu DiagnosaController akan me-redirect ke halaman data diagnosa.

Gambar 4. 16 Sequence diagram lihat hasil diagnosa

Dapat diperhatikan pada gambar 4. 7 merupakan sequence diagram dari proses menampilkan hasil diagnosa. Pertama pilih menu data diagnosa maka akan dijalankan suatu fungsi yang bernama index() pada suatu controller yang bernama DiagnosaDetailController. DiagnosaDetailController akan mengambil data diagnosa pada tabel diagnosa melalui model DiagnosaModel. DiagnosaDetailController kemudian akan menampilkan data diagnosa. Untuk melihat detail dari hasil diagnosa pengguna dapat meng-klik tombol detail pada diagnosa anak yang ingin dilihatnya. Maka akan dijalan fungsi detail(kode\_diagnosa) pada DiagnosaDetailController dengan membawa parameter kode\_diagnosa yang pengguna pilih. DiagnosaDetailController akan mengambil data pada tabel diagnosa\_detail berdasarkan kode\_diagnosa yang menjadi parameter melalui DiagnosaDetailModel dan menampilkan data hasil diagnosa tersebut

* 1. ERD

Data untuk kebutuhan aplikasi akan disimpan dalam bentuk relasional maksudnya adalah membawa data dalam bentuk table. Agar table yang digunakan untuk menyimpan data terbentuk dengan benar dan diperlukan pembuatan desain terlebih dahulu dengan Teknik ERD.

Berikut komponen ERD dan simbolnya. Lihat table.

|  |  |  |
| --- | --- | --- |
| Simbol | Nama | Keterangan |
| nama\_entitas | Entitas/ *Entity* | * Entitas merupakan suatu objek yang mampu dibedakan dengan objek lain * Nama entitas merupakan nama objek tunggal dan menggunakan nama yang mudah dipahami * Nama entitas biasanya benda * Entitas nantinya sebagai table di basis data |
| nama\_entitas | Relasi | * Hubungan antar entitas * Biasanya menggunakan kata kerja |
|  | Garis Relasi | * Penghubung antar relasi dan entitas * Kerelasian memiliki kardinalitas atau derajat hubungan * Ukuran kardinalitas antar entitas dilambangkan dengan   1-1 (*One to One*)  N-N (*Many to Many*)  N-1 (*Many to One*)  1-N (*One to Many*) |
|  | Atribut | * Semua data atau informasi yang berkaitan dengan entitas * Atribut nantinya sebagai field atau kolom didalam table |
|  | Atrubut primary key | * Data atau informasi dan entitas yang bersifat unik. Pada table database. Kolom ini sebagai *primary key* |

Tabel 4. 6 Simbol dan komponen Entitas Relationship Diagram

Alur kerja pada aplikasi:

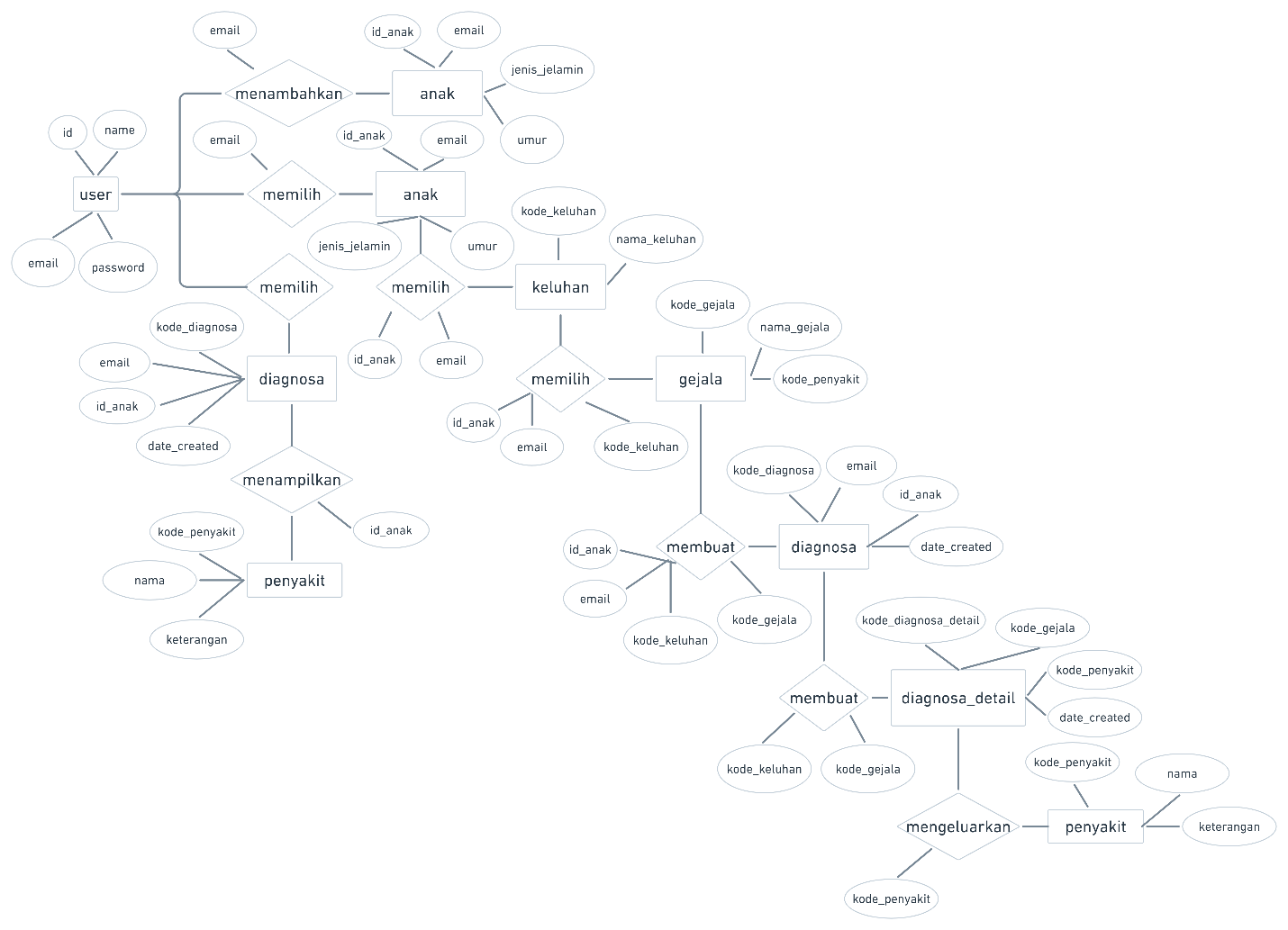
1. User melakukan register terlebih dahulu
2. User login setelah memiliki akun
3. Menambahkan data anak di halaman profil
4. User melakukan diagnosa dan memilih data anak yang akan dilakukan diagnose
5. User memilih keluhan yang terdapat pada anaknya
6. User memilih gejala yang terdapat pada anaknya
7. Diagnosa selesai,
8. User dapat melihat hasil diagnosa di halaman data diagnosa
9. Lalu pilih link detail pada anak yang ingin dilihat hasil diagnosanya
10. Muncul tampilan hasil diagnosa. Terdapat gejala yang dipilih dan penyakit yang didapatkan berdasarkan gejala yang dipilih.

Ditentukan entitas:

1. user
2. anak
3. diagnosa
4. keluhan
5. gejala
6. diagnosa\_detail
7. gejala\_return
8. penyakit

Atribut dari Entitas:

1. Atribut user: id, name, email, password.
2. Atribut anak: id, email, jenis\_kelamin, umur.
3. Atribut diagnosa: kode\_diagnosa, email, id\_anak, date\_created.
4. Atribut keluhan: kode\_keluhan, nama\_keluhan
5. Atribut gejala: kode\_gejala, kode\_penyakit, nama\_gejala
6. Atribut diagnosa\_detail: kode\_diagnosa\_detail, kode\_gejala, kode\_penyakit, date\_created.
7. Atribut gejala\_return: id\_gejala\_return, kode\_penyakit, kode\_gejala\_return, keterangan.
8. Atribut penyakit: kode\_penyakit, nama, keterangan.



Tabel 4. 7 ERD

* 1. Struktur Tabel

Berikut merupakan struktur tabel yang digunakan untuk pembuatan aplikasi

Tabel user

|  |  |
| --- | --- |
| Field | Type |
| Id | int(11) |
| Name | varchar(255) |
| Email | varchar(255) |
| password | varchar(128) |

Tabel 4. 8 User

Tabel anak

|  |  |
| --- | --- |
| Field | Type |
| id\_anak | int(11) |
| Email | varchar(25) |
| nama | varchar(50) |
| jenis\_kelamin | varchar(1) |
| umur | varchar(1) |

Tabel 4. 9 Anak

Tabel diagnosa

|  |  |
| --- | --- |
| Field | Type |
| kode\_diagnosa | int(11) |
| email | varchar(25) |
| id\_anak | int(11) |
| date\_created | int(11) |

Tabel 4. 10 Diagnosa

Tabel diagnosa\_detail

|  |  |
| --- | --- |
| Field | Type |
| kode\_diagnosa\_detail | int(11) |
| kode\_diagnosa | int(11) |
| Kode\_gejala | varchar(11) |
| kode\_penyakit | varchar(11) |
| date\_created | int(11) |

Tabel 4. 11 Detail Diagnosa

Tabel gejala

|  |  |
| --- | --- |
| Field | Type |
| kode\_gejala | varchar(11) |
| kode\_penyakit | varchar(11) |
| nama\_gejala | varchar(11) |

Tabel 4. 12 Gejala

Tabel gejala\_return

|  |  |
| --- | --- |
| Field | Type |
| id\_gejala\_return | int(11) |
| kode\_penyakit | varchar(11) |
| kode\_gejala\_return | varchar(11) |
| keterangan | varchar(255) |

Tabel 4. 13 Gejala Return

Tabel keluhan

|  |  |
| --- | --- |
| Field | Type |
| kode\_keluhan | varchar(11) |
| nama\_keluhan | varchar(11) |

Tabel 4. 14 Keluhan

Tabel penyakit

|  |  |
| --- | --- |
| Field | Type |
| kode\_penyakit | varchar(11) |
| Nama | varchar(11) |
| gambar | varchar(11) |
| keterangan | varchar(255) |
| referensi | varchar(128) |
| date\_created | int(11) |

Tabel 4. 15 Penyakit

1. Perancangan Pengujian
   1. Perancangan Analisis Performa
2. Pengujian Request Per Second (RPS)

Pengujian *request per second* ini dilakukan dengan menggunakan Apache Benchmark (ab). Parameter yang dihasilkan dari pengujian yang dilakukan adalah *request per second* (r/s) yang merupakan besarnya request/permintaan yang dapat ditangani dalam waktu 1 detik.

1. Pengujian Time Per Request

Pengujian *time per second* ini dilakukan dengan menggunakan Apache Benchmark (ab). Parameter yang dihasilkan dari pengujian yang dilakukan adalah *time per request* (ms) merupakan waktu yang dibutuhkan untuk menangani 1 *request*/permintaan.

1. Pengujian Execute Time (Waktu Eksekusi)

Pengujian *execute time* ini dilakukan dengan menggunakan fungsi microtime() pada php. Dengan fungsi ini dapat diketahui berapa waktu yang dibutuhkan untuk menjalankan suatu skrip yang ingin diuji kecepatannya. Dengan begitu dapat dilakukan pengujian pada aplikasi diagnosa anak untuk mengetahui *execute time* (waktu eksekusi) per-fitur pada anak. Fitur tersebut antara lain:

* Registrasi akun
* Login
* Melihat profil dan tambah data anak
* Diagnosa penyakit anak
* Lihat data hasil diagnosa
  1. Perancangan Analisis Ukuran

Analisis ukuran dilakukan dengan membandingkan dan menganalisis struktur direktori dan besarnya total file yang terdapat pada direktori. Kebanyakan framework PHP yang menganut pola MVC (Model-View-Controller) menggunakan skema direktori dengan nama "Model", "View" dan "Controller" yang seluruhnya dikumpulkan kedalam sebuah direktori utama yang bernama "src" atau "app" / "application" seperti Codeigniter. Sedikit berbeda dengan Codeigniter, direktori dengan nama “View” pada Laravel justru diletakkan di luar direktori “app”. Direktori “View” digunakan untuk menyimpan file-file yang berhubungan dengan tampilan aplikasi .Lalu pada direktori “app” baik pada Framework Laravel atau Framewok Codeigniter terdapat direktori “Model” yang digunakan untuk menyimpan class PHP yang berhubungan dengan model database. Kemudian direktori “Controller” digunakan untuk menyimpan class PHP yang berhubungan dengan *application logic.*

* 1. Perancangan Analisis Cara Akses Database

Cara akses database dilakukan dengan membandingkan dan menganalisis akses database untuk tata cara pengaksesan tabel dalam operasi Create, Read, Update, Delete (CRUD). Cara pengaksesan database di Laravel menggunakan Eloquent ORM dan cara pengaksesan database di CodeIgniter menggunakan Query Builder.

# BAB V IMPLEMENTASI DAN PENGUJIAN

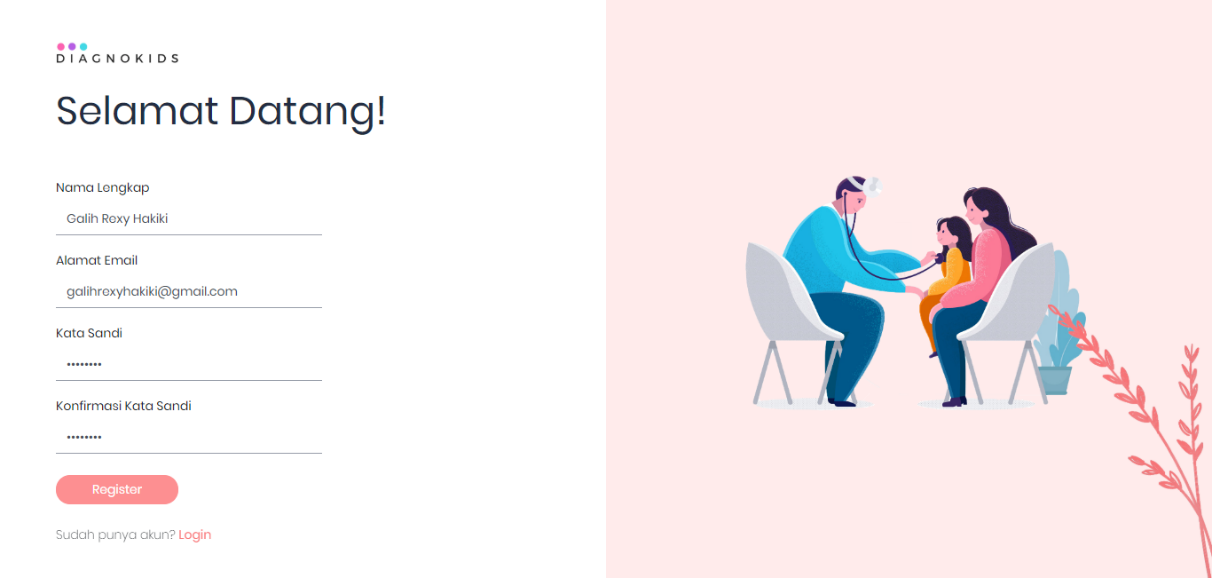
1. Implementasi

Implementasi adalah tahap penerapan sistem berdasarkan hasil analisis dan perancangan yang dilakukan pada bab IV. Pada bab V ini merupakan implemenasi hasil rancangan menjadi Aplikasi Diagnosa Pada Anak Umur 5 ke bawah untuk membandingkan Framework Laravel dengan Framework Codeigniter.

* 1. Implementasi User Interface

Pada implementasi ini dijelaskan setiap halaman dan tampilan yang terdapat pada aplikasi serta hubungannya dengan tampilan yang lainnya.

1. Tampilan Halaman Registrasi



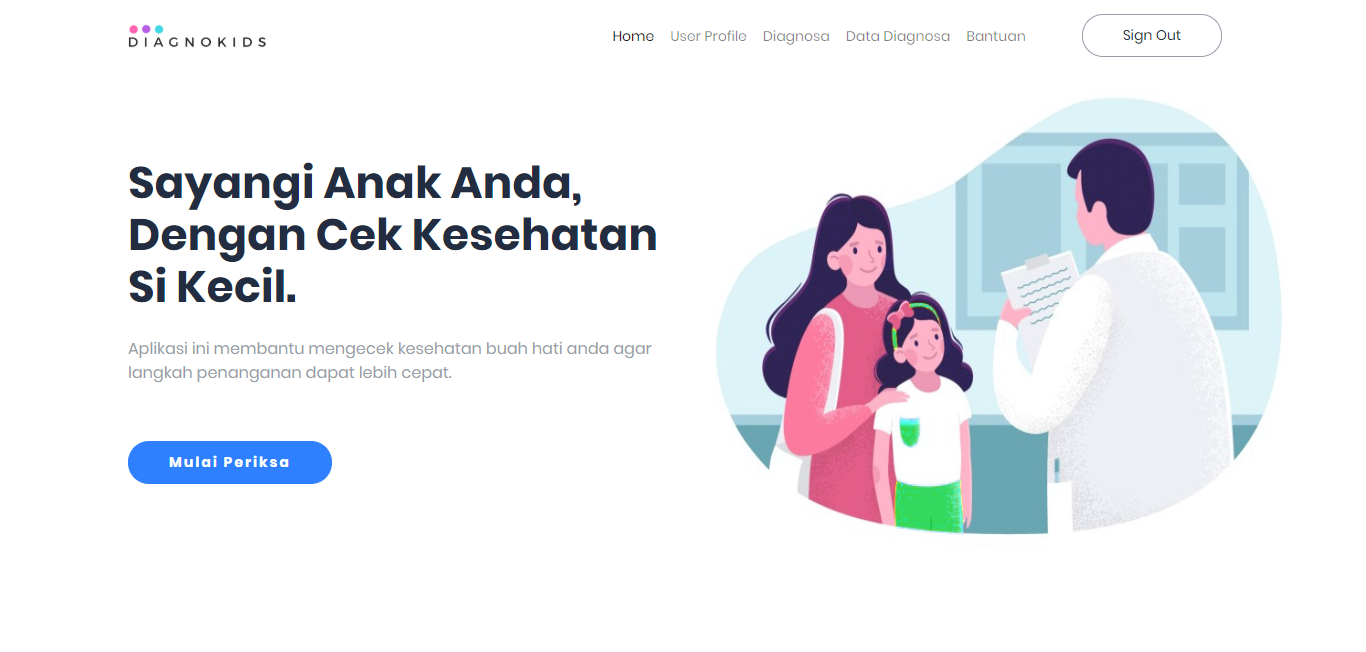
Pada gambar di atas adalah tampilan halaman registrasi pada aplikasi diagnosa pada anak. Pada bagian kiri terdapat *form* pengisian data *user* dan terdapat tombol registrasi untuk memproses data user yang telah dimasukkan.

1. Tampilan Halaman Login



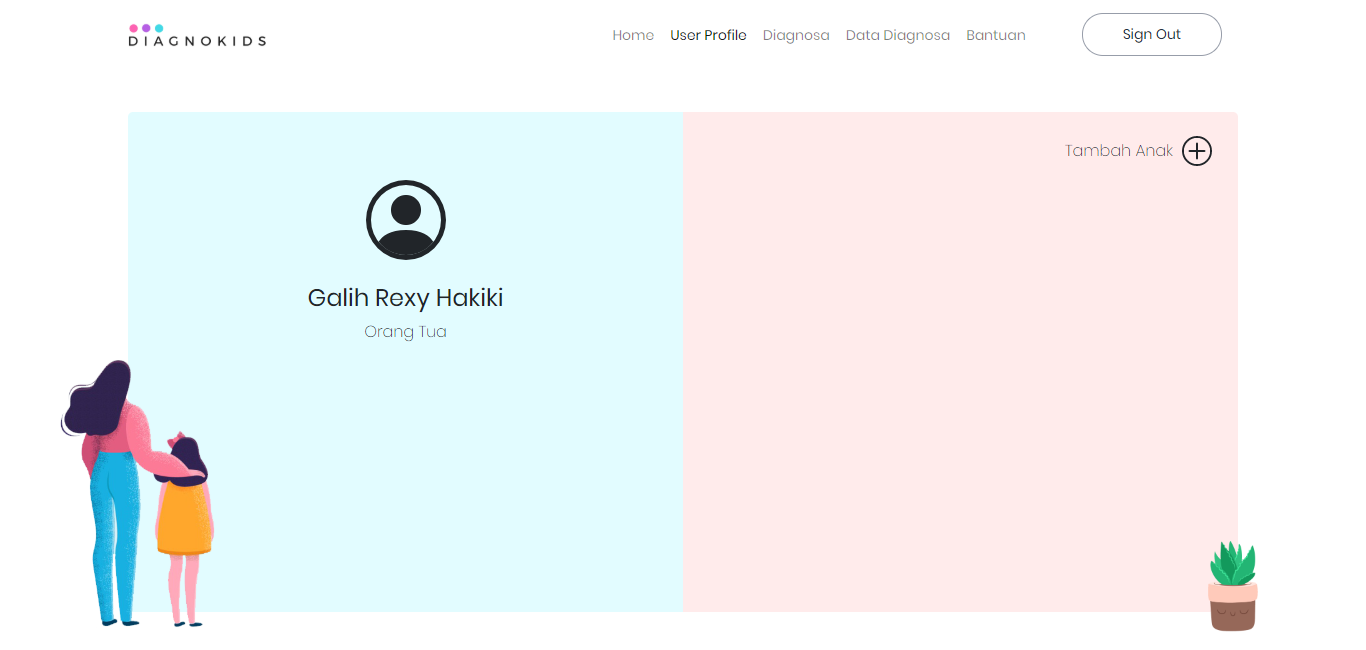
Gambar 5. 1 Halaman Login

Pada gambar di atas adalah tampilan halaman login pada aplikasi diagnosa pada anak. Pada bagian kiri terdapat *form* login dan terdapat tombol login untuk memproses/memvalidasi data user yang telah dimasukkan. Apabila data yang dimasukkan terdaftar pada database maka user akan di arahkan ke halaman home.

1. Tampilan Halaman Home

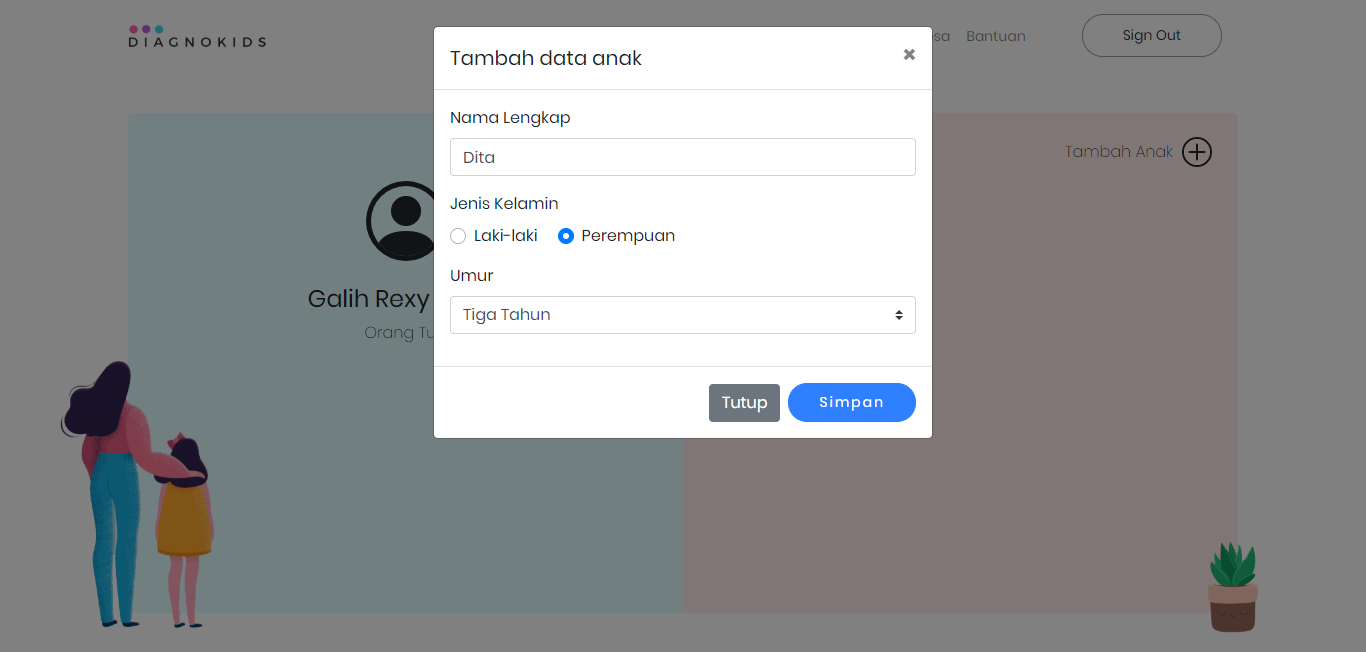
Gambar 5. 2 Halaman Home

Gambar di atas adalah tampilan halaman home. Ketika user memiliki data yang terdaftar pada database dan melakukan login dengan akun tersebut maka user akan diarahkan ke halaman home. Tombol Mulai Diagnosa pada tampilan di atas akan mengarahkan *user* untuk masuk ke halaman diagnosa.

1. Tampilan Halaman Profil

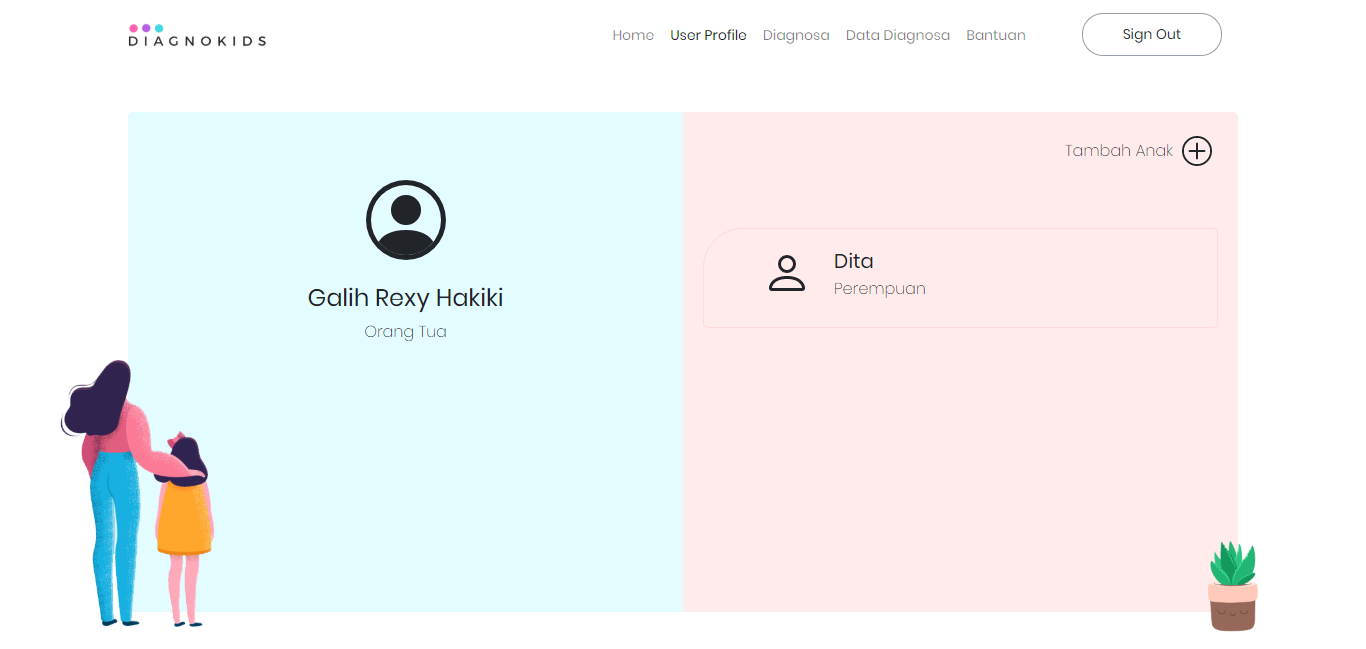
Gambar 5. 3 Halaman Profil

Gambar di atas merupakan tampilan halaman profil. Pada bagian kiri terdapat nama *user* sebagai orang tua. Dan pada bagian kiri terdapat tombol tambah anak untuk menambahkan data anak *user* yang nantinya akan digunakan untuk data diagnosa.

1. Tampilan tambah data anak

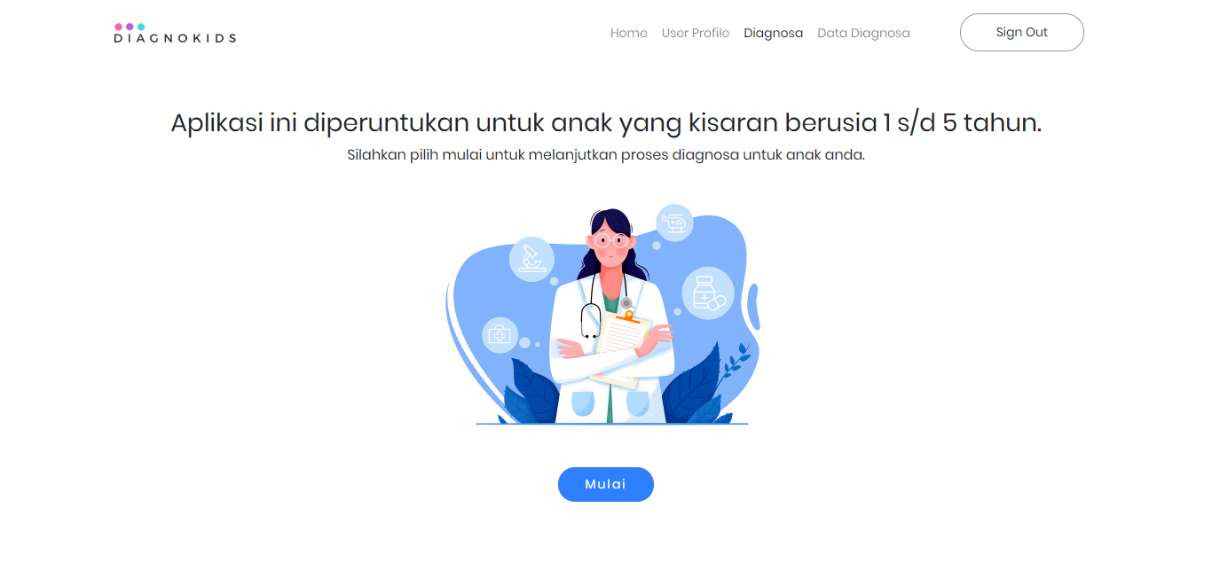
Gambar 5. 4 Tambah Data Profil

Ketika tombol tambah anak pada halaman profil diklik maka akan muncul tampilan seperti gambar di atas. *User* dapat memasukkan data anak sesuai form yang tersedia lalu pilik tombol Simpan untuk menyimpan data anak.

1. Tampilan data anak

Gambar 5. 5 Data Anak

Ketika data anak telah ditambahkan maka data anak yang telah ditambahkan akan ditampilkan pada halaman profil.

1. Tampilan Halaman Diagnosa

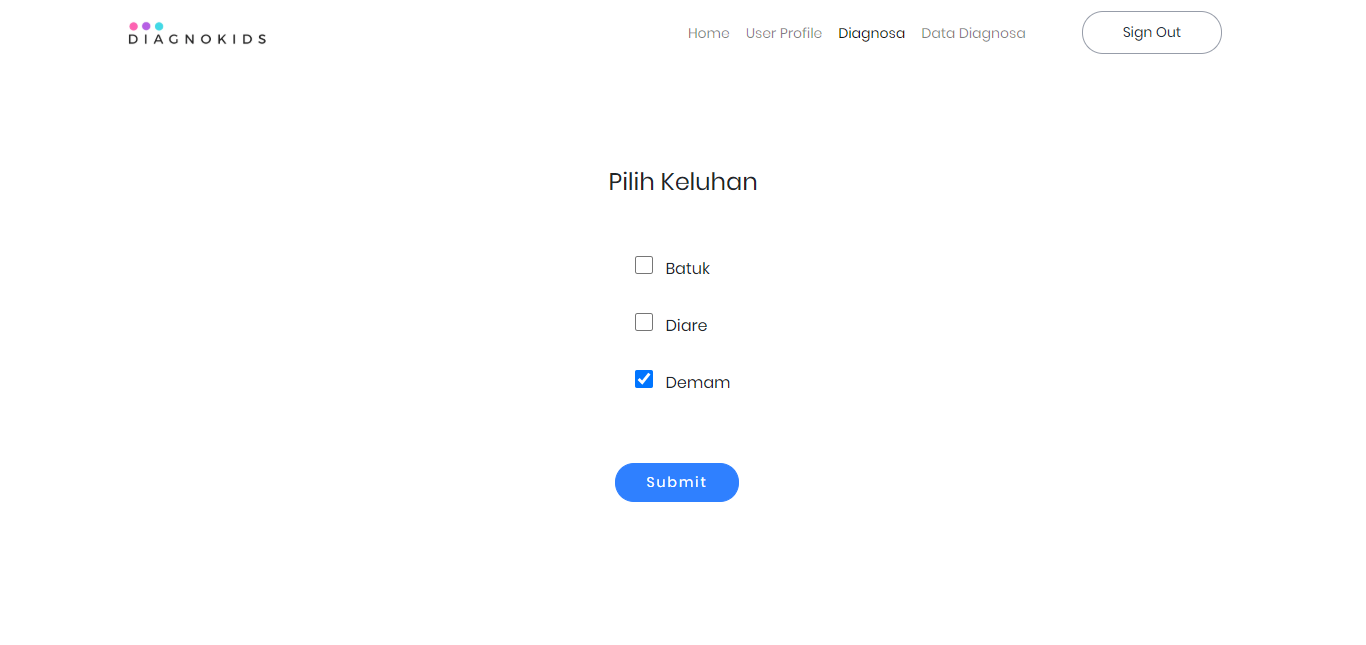
Gambar 5. 6 Halaman Diagnosa

Gambar di atas merupakan tampilan halaman diagnosa. Ketika tombol Mulai diklik oleh user maka proses diagnosa akan dimulai.

1. Tampilan Proses Diagnosa (Memilih Data Anak)

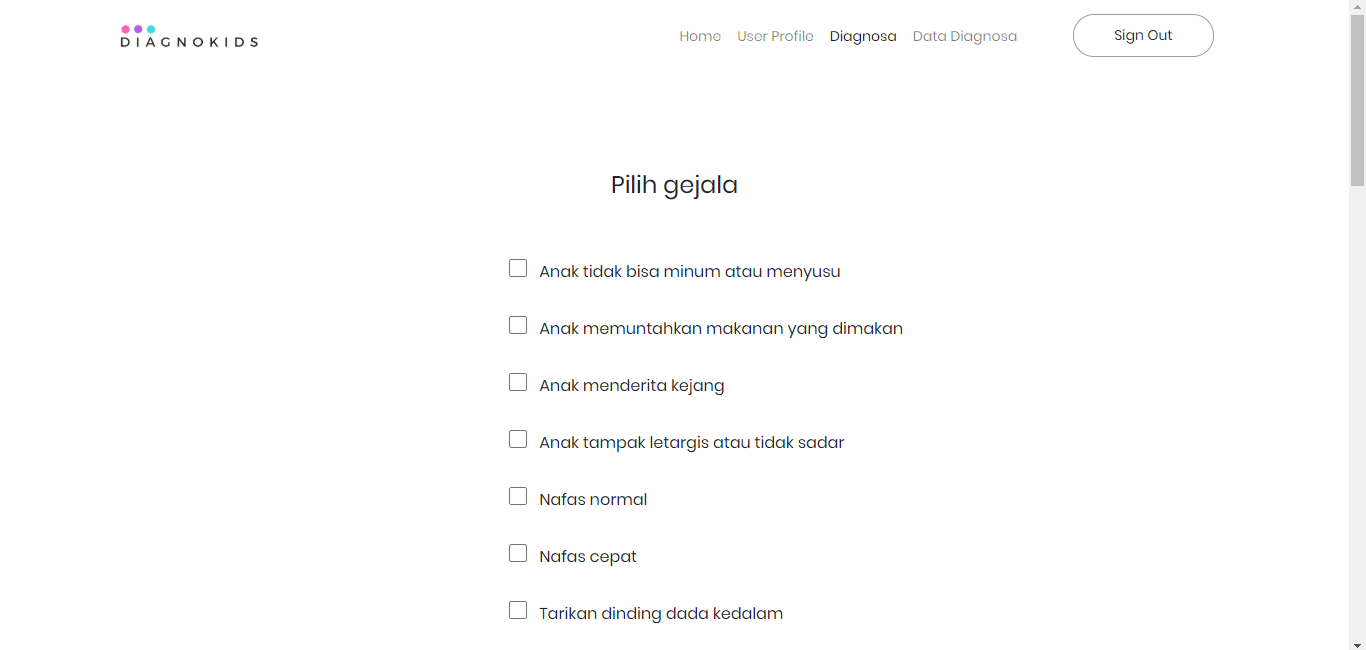
Gambar 5. 7 Pilih data anak

Ketika tombol Mulai diklik maka akan muncul tampil seperti diatas. User dapat memilih anak yang akan dilakukan diagnosa.

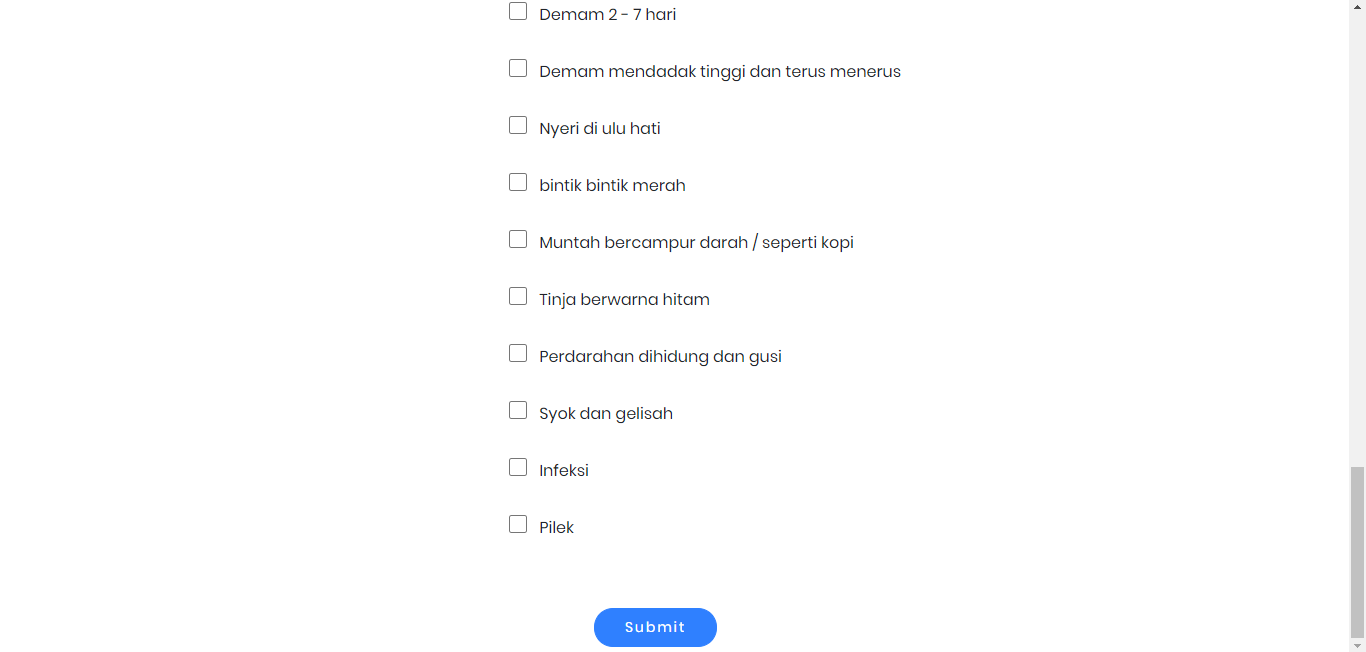
1. Tampilan Proses Diagnosa (Memilih Keluhan)

Gambar 5. 8 Pilih Keluhan

Ketika anak yang akan dilakukan diagnosa telah dipilih maka akan muncul tampilan seperti diatas. User dapat memilih keluhan apa saja yang terdapat pada anak lalu pilih submit.

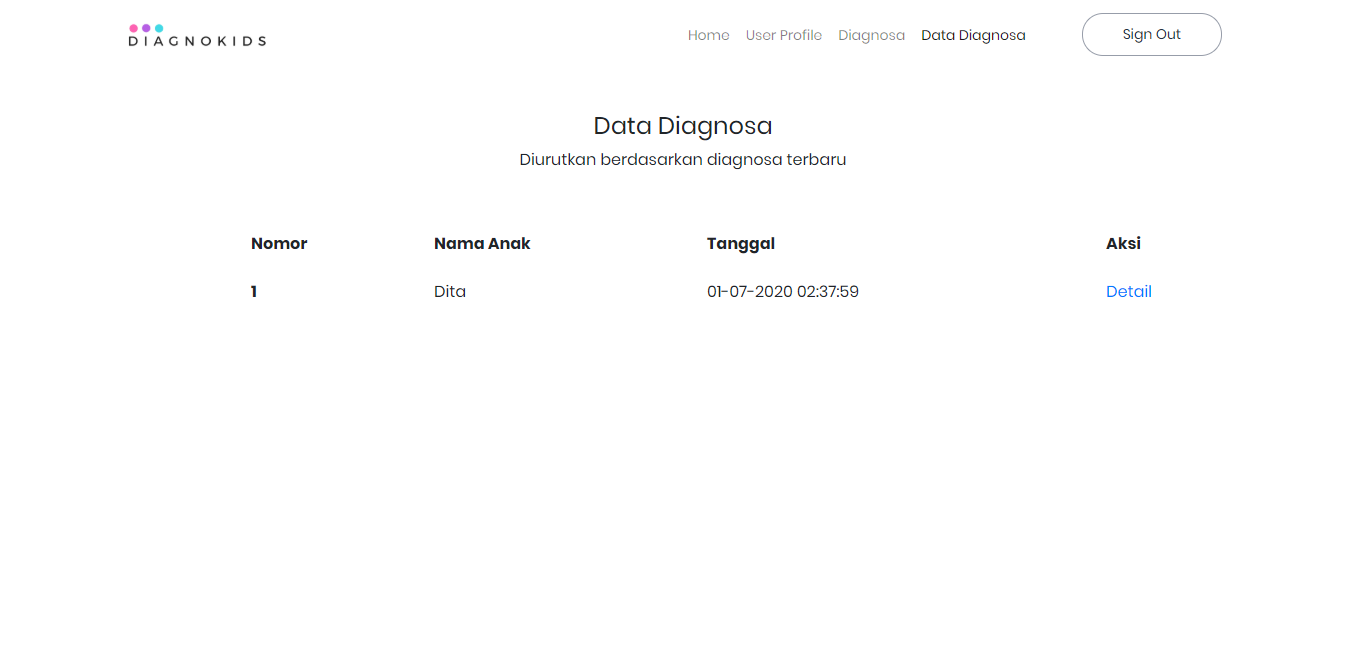
1. Tampilan Proses Diagnosa (Gejala)

Gambar 5. 9 Pilih Gejala 1



Gambar 5. 10 Pilih Gejala 2

Kedua gambar di atas adalah tampilan daftar gejala yang dapat dipilih oleh user. User dapat memilih gejala apa saja yang sesuai dengan yang terdapat pada anaknya. Ketika *user* telah memilih gejala maka user dapat menekan tombol submit di bawah untuk memproses data yang user telah *input*-kan.

1. Tampilan Halaman Data Diagnosa

Gambar 5. 11 Data Diagnosa

Gambar di atas adalah tampilan untuk halaman data diagnosa. Setelah user melakukan diagnosa maka akan mengeluarkan hasil diagnosa seperti pada gambar di atas. Terdapat tombol detail yang Ketika diklik akan menampilkan detail hasil diagnosa.

1. Tampilan Detail Diagnosa

Gambar 5. 12 Detail Diagnosa

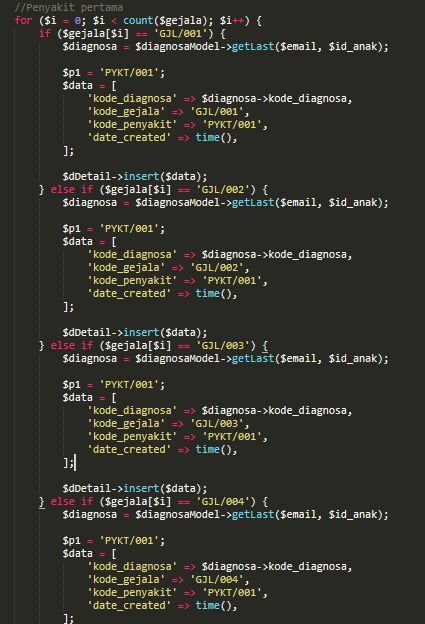
Gambar di atas merupakan tampilan hasil diagnosa. Terdapat kode diagnosa, nama dan tanggal diagnosa sebagai identitas hasil diagnosa. Lalu terdapat penyakit yang terdiagnosa dan juga gejala-gejala yang telah dipilih oleh user yang sesuai dengan gejala yang diderita oleh anak.

* 1. Implementasi Metode Forward Chaining

Aplikasi diagnosa anak ini menggunakan metode *forward chaining. Forward Chaining* merupakan metode yang sering digunakan dalam pembuatan aplikasi Sistem Pendukung Keputusan (SPK).

1. Penyakit Pertama

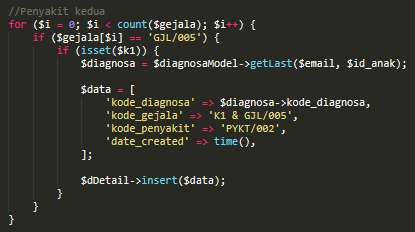
Di karenakan aturan untuk menghasilkan penyakit pertama adalah masukkan dari *user* harus lah (gejala pertama || gejala kedua || gejala ketiga || gejala keempat) maka implementasi *forward chaining* yang dihasilkan adalah:



Gambar 5. 13 Forward Chaining Penyakit 1

Penyakit Kedua

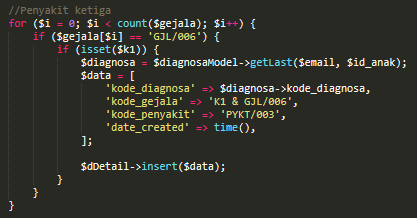
Di karenakan aturan untuk menghasilkan penyakit kedua adalah masukkan dari *user* harus lah (gejala kelima) maka implementasi *forward chaining* yang dihasilkan adalah:



Gambar 5. 14 Forward Chaining Penyakit 2

Penyakit Ketiga

Di karenakan aturan untuk menghasilkan penyakit ketiga adalah masukkan dari *user* harus lah (gejala ketiga) maka implementasi *forward chaining* yang dihasilkan adalah:



Gambar 5. 15 Forward Chaining Penyakit 3

1. Pengujian

Pengujian perbandingan framework Laravel dan framework Codeigniter ini dilakukan dengan membandingkan dari segi performa, ukuran dan cara akses database. Pengujian dari segi performa ada beberapa factor yang diujikan antara lain:

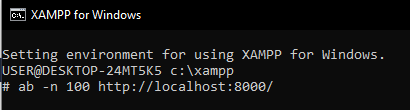
* *Request per second*
* *Time per request*
* *Execute time*

Dan untuk mengukuran ukuran dilakukan dengan membandingkan dan menganalisis struktur direktori dan besarnya total file yang terdapat pada direktori. Sedangkan untuk membandingkan cara pengaksesan *database* di Laravel menggunakan Eloquent ORM, Query Builder, dan Raw Query dan di CodeIgniter menggunakan Query Builder dan Query Basics.

* 1. Pengujian Segi Performa

1. *Request per second*

Pengujian dengan menggunakan Apache Benchmark (ab) *tool.* Pengujian dilakukan kepada server Laravel (Artisan) dan Codeigniter (Spark) dengan memberikan *request* sebanyak 100 dan pengujian dilakukan sebanyak 10 kali. Dan perintah yang dimasukkan pada ab *tool* adalah sebagai berikut:



Gambar 5. 16 Perintah untuk server Laravel



Gambar 5. 17 Perintah untuk server Codeigniter

Dan hasil yang akan ditampilkan pada ab *tool* adalah:



Gambar 5. 18 Tampilan hasil tool ab

Didapatlah waktu pengujian dan jumlah *request per second.* Pengujian pun dilakukan sebanyak 10 kali lalu dijumlahkan dan diambil nilai rata-ratanya. Dan setelah pengujian sebanyak 10 kali dilakukan didapatlah data sebagai berikut:

* Laravel

|  |  |  |  |
| --- | --- | --- | --- |
| Nomor Tes | Waktu Tes (detik) | Jumlah *Request* | *Request Per Second* |
| 1 | 53,203 s | 100 | 1,88 r/s |
| 2 | 44,410 s | 100 | 2,25 r/s |
| 3 | 59,491 s | 100 | 1,68 r/s |
| 4 | 47,741 s | 100 | 2,09 r/s |
| 5 | 43,743 s | 100 | 2,29 r/s |
| 6 | 50,589 s | 100 | 1,98 r/s |
| 7 | 46,763 s | 100 | 2,14 r/s |
| 8 | 46,994 s | 100 | 2,13 r/s |
| 9 | 44,282 s | 100 | 2,26 r/s |
| 10 | 51,326 s | 100 | 1,95 r/s |
| Rata-rata | 48,854 s | 100 | 2,06 r/s |

Tabel 5. 1 Request per second Laravel

Setelah dilakukan 10 kali pengujian dan diambil nilai rata-rata dari request per second. Rata-rata *request per second* yang dihasilkan oleh framework Laravel adalah 2,06 *request per second*.

* Codeigniter

|  |  |  |  |
| --- | --- | --- | --- |
| Nomor Tes | Waktu Tes (detik) | Jumlah *Request* | *Request Per Second* |
| 1 | 13,629 s | 100 | 7,34 r/s |
| 2 | 15,284 s | 100 | 6,54 r/s |
| 3 | 14,790 s | 100 | 6,76 r/s |
| 4 | 13,370 s | 100 | 7,48 r/s |
| 5 | 21,656 s | 100 | 4,62 r/s |
| 6 | 13,360 s | 100 | 7,47 r/s |
| 7 | 12,988 s | 100 | 7,70 r/s |
| 8 | 17,124 s | 100 | 5,84 r/s |
| 9 | 13,881 s | 100 | 7,20 r/s |
| 10 | 14,541 s | 100 | 6,88 r/s |
| Rata-rata | 15,062 s | 100 | 6,78 r/s |

Tabel 5. 2 Request per second Codeigniter

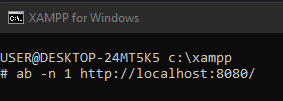
Setelah dilakukan 10 kali pengujian dan diambil nilai rata-rata dari request per second. Rata-rata *request per second* yang dihasilkan oleh framework Codeigniter adalah 6,78 *request per second*. Sehingga jika dibandingkan codeigniter lebih unggul karena request per second yang dihasilkan lebih banyak yaitu sebanyak 6,78 r/s jika dibandingkan dengan Laravel yang hanya menghasilkan request per second sebanyak 2,06 r/s

1. *Time per request.*

Pengujian dengan menggunakan Apache Benchmark (ab) *tool.* Pengujian dilakukan kepada server Laravel (Artisan) dan Codeigniter (Spark) dengan memberikan *request* sebanyak 1 dan pengujian dilakukan sebanyak 10 kali. Dan perintah yang dimasukkan pada ab *tool* adalah sebagai berikut:

**

Gambar 5. 19 Perintah time per request Laravel

**

Gambar 5. 20 Perintah time per request Codeigniter

Pengujian pun dilakukan sebanyak 10 kali lalu dijumlahkan dan diambil nilai rata-ratanya. Dan setelah pengujian sebanyak 10 kali dilakukan didapatlah data sebagai berikut:

* Laravel

|  |  |  |  |
| --- | --- | --- | --- |
| Nomor Tes | Waktu Tes (detik) | Jumlah *Request* | *Time Per Request* |
| 1 | 2,314 s | 1 | 2314 ms |
| 2 | 2,435 s | 1 | 2435 ms |
| 3 | 2,465 s | 1 | 2465 ms |
| 4 | 2,341 s | 1 | 2341 ms |
| 5 | 2,353 s | 1 | 2353 ms |
| 6 | 2,351 s | 1 | 2351 ms |
| 7 | 2,487 s | 1 | 2487 ms |
| 8 | 3,098 s | 1 | 3098 ms |
| 9 | 2,349 s | 1 | 2349 ms |
| 10 | 2,596 s | 1 | 2586 ms |
| Rata-rata | 2,477 s | 1 | 2477 ms |

Tabel 5. 3 Pengujian time per second Laravel

Setelah dilakukan 10 kali pengujian dan diambil nilai rata-rata dari *time per request*. Rata-rata *time per request* yang dihasilkan oleh framework Laravel adalah 2349 ms/r.

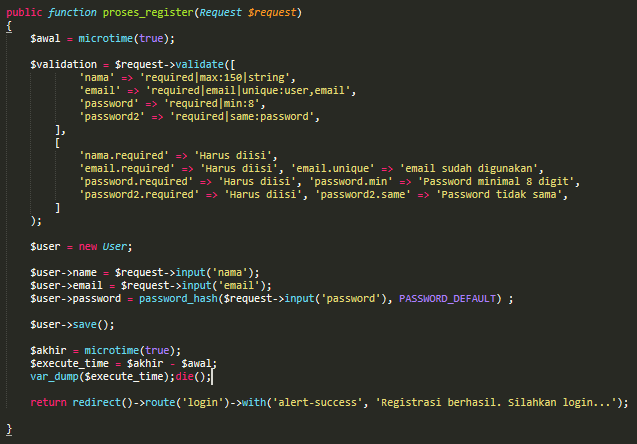
* Codeigniter

|  |  |  |  |
| --- | --- | --- | --- |
| Nomor Tes | Waktu Tes (detik) | Jumlah *Request* | *Time Per Request* |
| 1 | 0,153 s | 1 | 153 ms |
| 2 | 0,144 s | 1 | 144 ms |
| 3 | 0,128 s | 1 | 128 ms |
| 4 | 0,149 s | 1 | 149 ms |
| 5 | 0,122 s | 1 | 122 ms |
| 6 | 0,119 s | 1 | 119 ms |
| 7 | 0,113 s | 1 | 113 ms |
| 8 | 0,139 s | 1 | 139 ms |
| 9 | 0,437 s | 1 | 437 ms |
| 10 | 0,120 s | 1 | 120 ms |
| Rata-rata | 0, 162 s | 1 | 162 ms |

Tabel 5. 4 Pengujian time per second Codeigniter

Setelah dilakukan 10 kali pengujian dan diambil nilai rata-rata dari *time per request*. Rata-rata *time per request* yang dihasilkan oleh framework Codeigniter adalah 162 ms/r. Sehingga jika dibandingkan codeigniter lebih unggul karena *time per request* yang dihasilkan lebih cepat yaitu 162 ms/r jika dibandingkan dengan Laravel yang hanya menghasilkan request per second sebanyak 2477 ms/r.

1. *Execute time*

Pengujian execute time ini dilakukan dengan menggunakan fungsi microtime() pada php. Dengan fungsi ini dapat diketahui berapa waktu yang dibutuhkan untuk menjalankan suatu skrip yang ingin diuji kecepatannya. Dengan begitu dapat dilakukan pengujian pada aplikasi diagnosa anak untuk mengetahui execute time (waktu eksekusi) pada aplikasi. Lalu execute time akan ditampilkan dengan fungsi var\_dump(). Pengujian dilakukan sebanyak 5 kali lalu dihitung nilai rata-ratanya. Berikut contoh penerapan pengujian yang dilakukan dengan memanfaatkan fungsi microtime().

Gambar 5. 21 Penerapan pengujian execute time

Gambar di atas menunjukan bahwa fungsi microtime() yang diletakkan di antara skrip proses registrasi. Penerapan pengujian tersebut dilakukan kepada setiap fitur yang tersedia pada aplikasi diagnosa anak. Berikut hasil pengujian dari tiap fitur yang telah diuji:

* Proses Registrasi

|  |  |  |
| --- | --- | --- |
| No. Tes | Laravel | Codeigniter |
| 1 | 1.362 detik | 1.254 detik |
| 2 | 0.431 detik | 0.425 detik |
| 3 | 0.599 detik | 0.539 detik |
| 4 | 0.535 detik | 0.479 detik |
| 5 | 0.571 detik | 0.451 detik |
| Rata-rata | 0,699 detik | 0,629 detik |

Tabel 5. 5 Execute time proses registrasi

Setelah dilakukan 5 kali pengujian untuk proses registrasi dan diambil nilai rata-rata dari *execute time*. Rata-rata *execute time* yang dihasilkan oleh framework Laravel adalah 0,699 detik. Sementara rata-rata *execute time* yang dihasilkan oleh framework Codeigniter adalah 0,629 detik.

* Proses Login

|  |  |  |
| --- | --- | --- |
| No. Tes | Laravel | Codeigniter |
| 1 | 0.521 detik | 0.445 detik |
| 2 | 0.341 detik | 0.412 detik |
| 3 | 0.390 detik | 0.360 detik |
| 4 | 0.363 detik | 0.338 detik |
| 5 | 0.231 detik | 0.454 detik |
| Rata-rata | 0,369 detik | 0,402 detik |

Tabel 5. 6 Execute time proses login

Setelah dilakukan 5 kali pengujian untuk proses login dan diambil nilai rata-rata dari execute time. Rata-rata execute time yang dihasilkan oleh framework Laravel adalah 0,369 detik. Sementara rata-rata execute time yang dihasilkan oleh framework Codeigniter adalah 0,402 detik.

* Proses tambah data anak

|  |  |  |
| --- | --- | --- |
| No. Tes | Laravel | Codeigniter |
| 1 | 0.089 detik | 0.152 detik |
| 2 | 0.207 detik | 0.110 detik |
| 3 | 0.143 detik | 0.253 detik |
| 4 | 0.200 detik | 0.167 detik |
| 5 | 0.126 detik | 0.199 detik |
| Rata-rata | 0,153 detik | 0,176 detik |

Tabel 5. 7 Execute time proses tambah data anak

Setelah dilakukan 5 kali pengujian untuk proses tambah data anak dan diambil nilai rata-rata dari *execute time*. Rata-rata *execute time* yang dihasilkan oleh framework Laravel adalah 0,153 detik. Sementara rata-rata *execute time* yang dihasilkan oleh framework Codeigniter adalah 0,176 detik.

* Proses diagnosa

|  |  |  |
| --- | --- | --- |
| No. Tes | Laravel | Codeigniter |
| 1 | 0.221 detik | 0.276 detik |
| 2 | 2.874 detik | 1.966 detik |
| 3 | 0.206 detik | 0.345 detik |
| 4 | 0.289 detik | 0.358 detik |
| 5 | 1.047 detik | 0.888 detik |
| Rata-rata | 0,927 detik | 0,766 detik |

Tabel 5. 8 Execute time proses diagnosa

Setelah dilakukan 5 kali pengujian untuk proses diagnosa dan diambil nilai rata-rata dari *execute time*. Rata-rata *execute time* yang dihasilkan oleh framework Laravel adalah 0,927 detik. Sementara rata-rata *execute time* yang dihasilkan oleh framework Codeigniter adalah 0,766 detik.

* Proses data diagnosa

|  |  |  |
| --- | --- | --- |
| No. Tes | Laravel | Codeigniter |
| 1 | 0.046 detik | 0.057 detik |
| 2 | 0.281 detik | 0.118 detik |
| 3 | 0.031 detik | 0.041 detik |
| 4 | 0.038 detik | 0.053 detik |
| 5 | 0.058 detik | 0.099 detik |
| Rata-rata | 0,09 detik | 0,073 detik |

Tabel 5. 9 Execute time proses data diagnosa

Setelah dilakukan 5 kali pengujian untuk proses lihat data diagnosa dan diambil nilai rata-rata dari *execute time*. Rata-rata *execute time* yang dihasilkan oleh framework Laravel adalah 0,09 detik. Sementara rata-rata *execute time* yang dihasilkan oleh framework Codeigniter adalah 0,073 detik.

* Rata-rata pengujian execute time

|  |  |  |
| --- | --- | --- |
| Nama Tes | Laravel | Codeigniter |
| Proses register | 0,699 detik | 0,629 detik |
| Proses login | 0,369 detik | 0,402 detik |
| Proses tambah data anak | 0,153 detik | 0,176 detik |
| Proses diagnosa | 0,927 detik | 0,766 detik |
| Proses data diagnosa | 0,09 detik | 0,073 detik |
| Rata-rata | 0,447 detik | 0,409 detik |

Tabel 5. 10 Rata-rata Execute time

Setelah dilakukan perhitungan untuk mencari rata-rata dari setiap hasil pengujian fitur ada. Framework codeigniter lebih unggul dengan kecepatan *execute time* rata-rata 0,409. Sedikit unggul disbanding dengan framework Laravel yang memperoleh *execute time* rata-rata 0,447 detik.

* 1. Pengujian Segi Ukuran

Pengujian ukuran dilakukan dengan membandingkan dan menganalisis struktur direktori framework Laravel dan framework Codeigniter dan besarnya total file yang terdapat pada direktori.

Laravel

|  |  |  |
| --- | --- | --- |
| Nama direktori | keterangan | Jumlah Ukuran |
| /app | Berisi kode ini pada aplikasi, hamper semua class terdapat pada direktori ini. | 43.9 KB |
| /bootstrap | Berisi file app.php yang mana file tersebut framework bootstrap | 15.9 KB |
| /config | Berisi semua file konfigurasi aplikasi | 44.8 KB |
| /database | berisi migrasi database aplikasi. Direktori ini berhubungan dengan database | 3.32 KB |
| /public | Berisi file index dan juga sebagai penyimpanan asset seperti gambar, css dan javascript. | 4.82 MB |
| /resource | Berisi file *view* untuk aplikasi. Dan juga penyimpanan *un-compiled* asset seperti LESS, SASS atau Javascript | 48.8 KB |
| /routes | Berisi rute url untuk aplikasi | 3.21 KB |
| /storage | dapat digunakan untuk menyimpan file yang dibuat pengguna, seperti avatar profil, yang harus dapat diakses publik | 1.21 MB |
| /tests | Berisi fitur PHPUnit untuk tes aplikasi. | 1.11 KB |
| /vendor | Berisi ke composer depedensi. *Package* yang dapat digunakan. | 36.1 MB |
| file pada direktori root | File yang terdapat pada direktori root | 688 KB |
| Total Ukuran | | 43 MB |

Tabel 5. 11 Direktori Laravel

Codeigniter

|  |  |  |
| --- | --- | --- |
| Nama direktori | keterangan | Jumlah Ukuran |
| /app | Berisi semua file aplikasi yang pengguna buat. | 14.9 MB |
| /public | Berisi file index dan asset apa pun untuk kebutuhan aplikasi | 4.83 MB |
| /system | Berisi file sistem yang tidak boleh diubah | 2.31 MB |
| /writable | direktori ini menampung semua direktori yang mungkin perlu ditulis selama masa aplikasi. Seperti cache, log dan setiap unggahan yang dikirimkan pengguna | 681 KB |
| file pada direktori root | File yang terdapat pada direktori root | 13.7 KB |
| Total Ukuran | | 22.7 MB |

Tabel 5. 12 Direktori Codeigniter

* 1. Pengujian Cara Akses Database

Cara akses database dilakukan dengan membandingkan dan menganalisis akses database. Cara pengaksesan database di Laravel menggunakan Eloquent ORM dan cara pengaksesan database di CodeIgniter menggunakan Query Builder.

Laravel (Eloquent ORM)

Aplikasi diagnosa anak ini (dengan laravel) dibuat dengan Eloquent ORM dalam mengakses databasenya. Sehingga data dapat dikelola dengan mudah. Karena dengan Eloquent ORM pengembang web dapat mengelola data yang ada pada database dari hanya satu buah model. Berikut penerapan Eloquent ORM pada proses diagnosa anak.

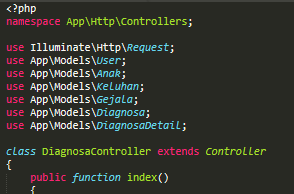
* Model Diagnosa



Gambar 5. 22 Model Diagnosa

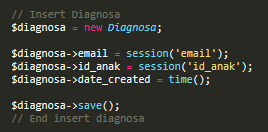
Gambar di atas merupakan model untuk tabel diagnosa. Terdapat $table dan $primarykey sebagai identitas model yang merujuk pada tabel. Terdapat juga fungsi di dalam model yang memudahkan pengembang web Ketika ingin menjalankan perintah pada fungsi yang berhubungan dengan model tertentu.

* Controller Diagnosa



Gambar 5. 23 Controller Diagnosa

Gambar di atas adalah file diagnosa controller. Dengan adanya model diagnosa yang telah dibuat, model hanya perlu dipanggil pada file diagnosa controller. Dengan begitu dalam melakukan CRUD akan lebih mudah Seperti pada gambar dibawah.



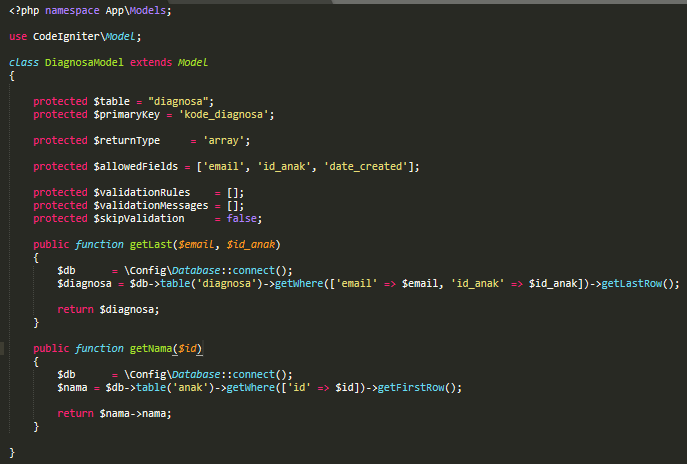
Gambar 5. 24 Create Laravel

Gambar di atas merupakan penerapan CRUD(*create/insert)* data pada tabel diagnosa melalui model diagnosa. Dalam penerapan eloquent yang diperlukan adalah buat representasi dari model lalu data dideklarasikan dalam sebuah objek pada modeldan gunakan operasi eloquent (CRUD).

Codeigniter (Query Builder)

Aplikasi diagnosa anak ini (dengan Codeigniter) dibuat dengan Query Builder dalam mengakses databasenya. Sehingga data dapat dikelola dengan mudah. Karena dengan Query Builder pengembang web dapat mengelola data yang ada pada database dari hanya satu buah model. Berikut penerapan Query Builder pada proses diagnosa anak.

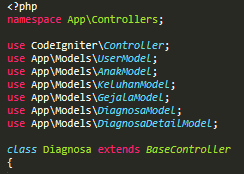
* Model Diagnosa



Gambar 5. 25 Model Diagnosa Codeigniter

Gambar di atas merupakan model untuk tabel diagnosa. Terdapat $table dan $primarykey sebagai identitas model yang merujuk pada tabel. Terdapat juga fungsi di dalam model yang memudahkan pengembang web Ketika ingin menjalankan perintah pada fungsi yang berhubungan dengan model tertentu.

* Controller Diagnosa



Gambar 5. 26 Controller Diagnosa Codeigniter

Gambar di atas adalah file diagnosa controller. Dengan adanya model diagnosa yang telah dibuat, model hanya perlu dipanggil pada file diagnosa controller. Dengan begitu dalam melakukan CRUD akan lebih mudah Seperti pada gambar dibawah.



Gambar 5. 27 Create Codeigniter

Gambar di atas merupakan penerapan CRUD(*create/insert)* data pada tabel diagnosa melalui model diagnosa. Dalam penerapan eloquent yang diperlukan adalah data dideklarasikan dalam sebuah *array* lalu buat representasi dari model dan gunakan operasi eloquent (CRUD).

Tidak terdapat perbedaan yang berarti dari segi cara akses database dengan Laravel (Eloquent ORM) dengan Codeigniter (Query Builder). Keduanya mampu merepresentasikan sebuah tabel pada sebuah model dan dapat membuat operasi CRUD jadi lebih mudah.

# BAB VI KESIMPULAN DAN SARAN

1. Kesimpulan

Dari pengujian dan pembahasan yang telah dilakukan, penulis dapat menarik kesimpulan analisis yang dibuat sebagai berikut:

Performa pada Framework Laravel dan Framework Codeigniter pada analisis *request per second, time per request,* dan *execute time* memiliki hasil yang berbeda. Pada analisis *response time,* Framework Codeigniter memiliki nilai yang lebih unggul 4,72 r/s dibandinng Framework Laravel. Pada analisis *time per request* Framework Codeigniter memiliki nilai yang lebih unggul 2315 ms dibanding Framework Laravel. Pada analisis *execute time,* baik Framework Laravel dan Framework Codeigniter memiliki nilai yang relative sama.

Ukuran pada Framework Laravel lebih besar dibanding Framework Codeigniter. Besarnya file Framework dikarenakan banyak vendor yang menyediakan package-package yang akan sanget membantu ketika aplikasi dibuat adalah aplikasi yang besar dan lumayan susah dalam pengelolaannya.

Cara akses database pada Framework CodeIgniter menggunakan Query Builder, sedangkan pada Framework Laravel menggunakan Eloquent ORM. Dalam penerapannya, semuanya memiliki satu tujuan yaitu agar dapat terkoneksi dengan database dan melakukan CRUD. Keduanya mampu merepresentasikan sebuah tabel pada sebuah model dan dapat membuat operasi CRUD jadi lebih mudah.

1. Saran

Dari pengujian yang telah dilakukan, penulis dapat mengumpulkan saran sebagai berikut:

Jika ingin membuat program dengan performa yang lebih cepat, dinilai dari *request per second, time per request*, dan *execute time*, Framework Codeigniter dapat dijadikan pilihan. Namun, jika ingin membuat program tanpa memikirkan tentang kecepatan, Framework Laravel dapat dijadikan sebuah pilihan.

Jika ingin membuat program dengan ukuran lebih kecil, Framework Codeigniter bisa dijadikan sebuah pilihan. Namun, jika ingin membuat program dengan *package* cukup lengkap, Framework Laravel dapat dijadikan sebuah pilihan.

Jika ingin membuat program dengan cara akses database yang mempermudah operasi CRUD, baik Framework CodeIgniter maupun Framework Laravel dapat dijadikan pilihan.

# DAFTAR PUSTAKA

Hamid, Muhammad Nur. 2019. Analisis Perbandingan Framework Codeigniter dan  
Framework Laravel (Studi Kasus Inventaris HMJ TI STMIK AKAKOM YOGRAKARTA). Yogyakarta: STMIK AKAKOM.

Erinton, Ruli., Ridha Muldina Negara dan Danu Dwi Sanjoyo. 2017. Analisis  
Performasi Framework Codeigniter dan Laravel Menggunakan Web Server Apache. Bandung: Telkom University.

Yanto, Bagus Fery., Indah Werdiningsih dan Endah Purwanti. 2017. Aplikasi Sistem  
Pakar Diagnosa Penyakit Pada Anak Bawah Lima Tahun Menggunakan Metode Forward Chaining. Surabaya: Universitas Airlangga.

Hidayatullah, Priyanto dan Jauhari Khairul Kawistara. 2017. Pemrogramman WEB.  
Bandung: Informatika.

Sidik, Betha. 2018. FRAMEWORK CODEIGNITER 3 Membangun Pemrograman  
Berbasis Web Dengan Berbagai Kemudahan & Fasilitas Codeigniter 3. Bandung: Informatika.

Maharani, Meilan Anastasia. 2018. Analisa dan Perancangan Sistem Informasi  
dengan Codeigniter dan Laravel. Yogyakarta: Lokomedia.

Rika Rosnelly. 2012. Sistem Pakar: Konsep dan Teori. Yogyakarta: Andi.

Rohi Abdulloh. 2018. 7 in 1 Pemrograman WEB Untuk Pemula. Jakarta: PT Elex Media Komputindo

Yurindra. 2017. Software Engineering. Yogyakarta: CV BUDI UTAMA

Request per second. (2020, Juli 21), Retrieved from https://support.loadimpact.com/3.0/test-configuration/what-are-requests-per-second-rps/

Apache Benckmark (ab). (2020, Juli 21), Retrieved from https://httpd.apache.org/docs/2.4/programs/ab.html

Ruli Erinton, Ridha Muldina Negara, Danu Dwi Sanjoyo. 2017. ANALISIS PERFORMASI FRAMEWORK CODEIGNITER DAN LARAVEL MENGGUNAKAN WEB SERVER APACHE. Bandung: Telkom University.

# LAMPIRAN

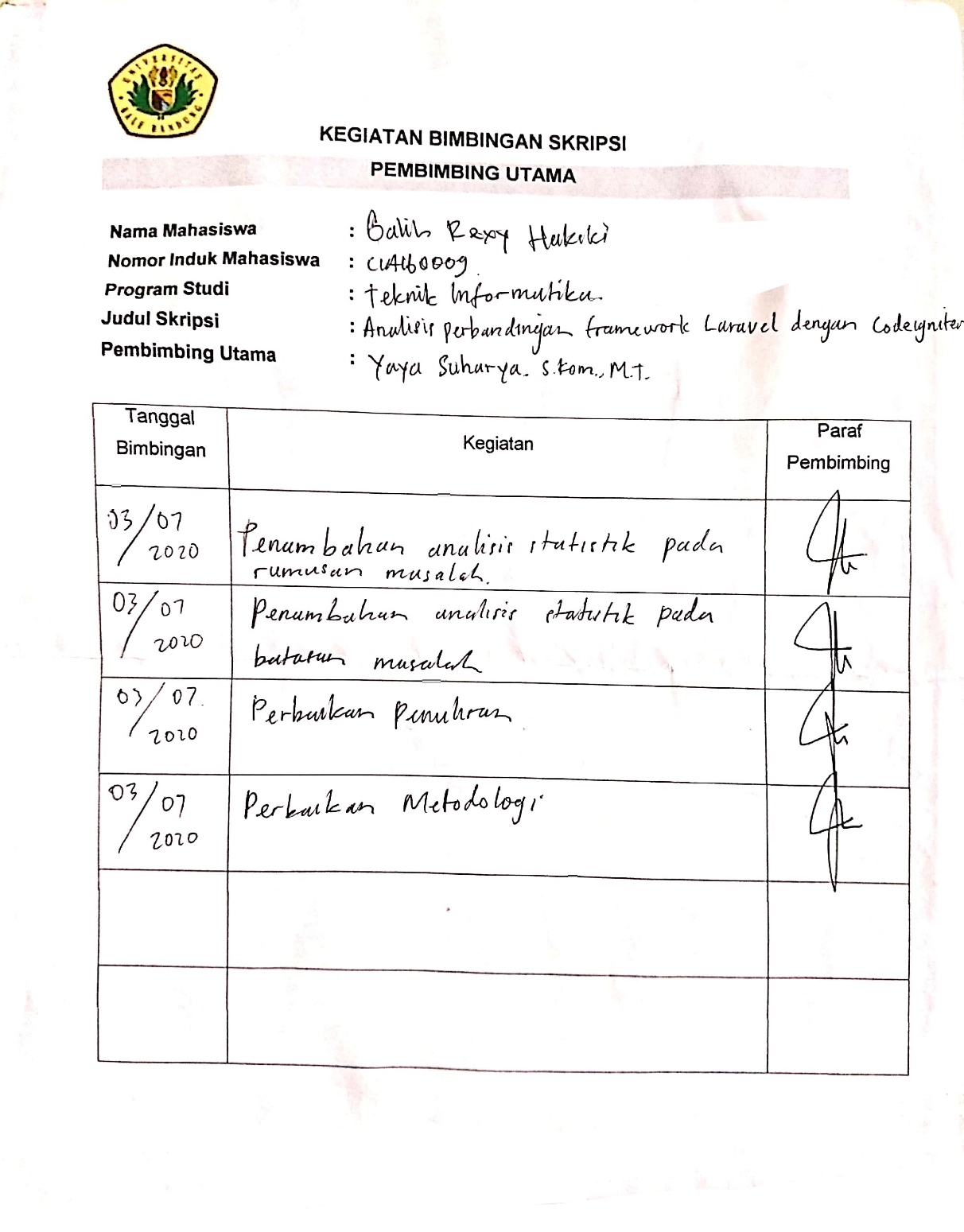
**Lampiran 1 Form Wawancara Perbandingan Framework Laravel dan Framework Codeigniter**

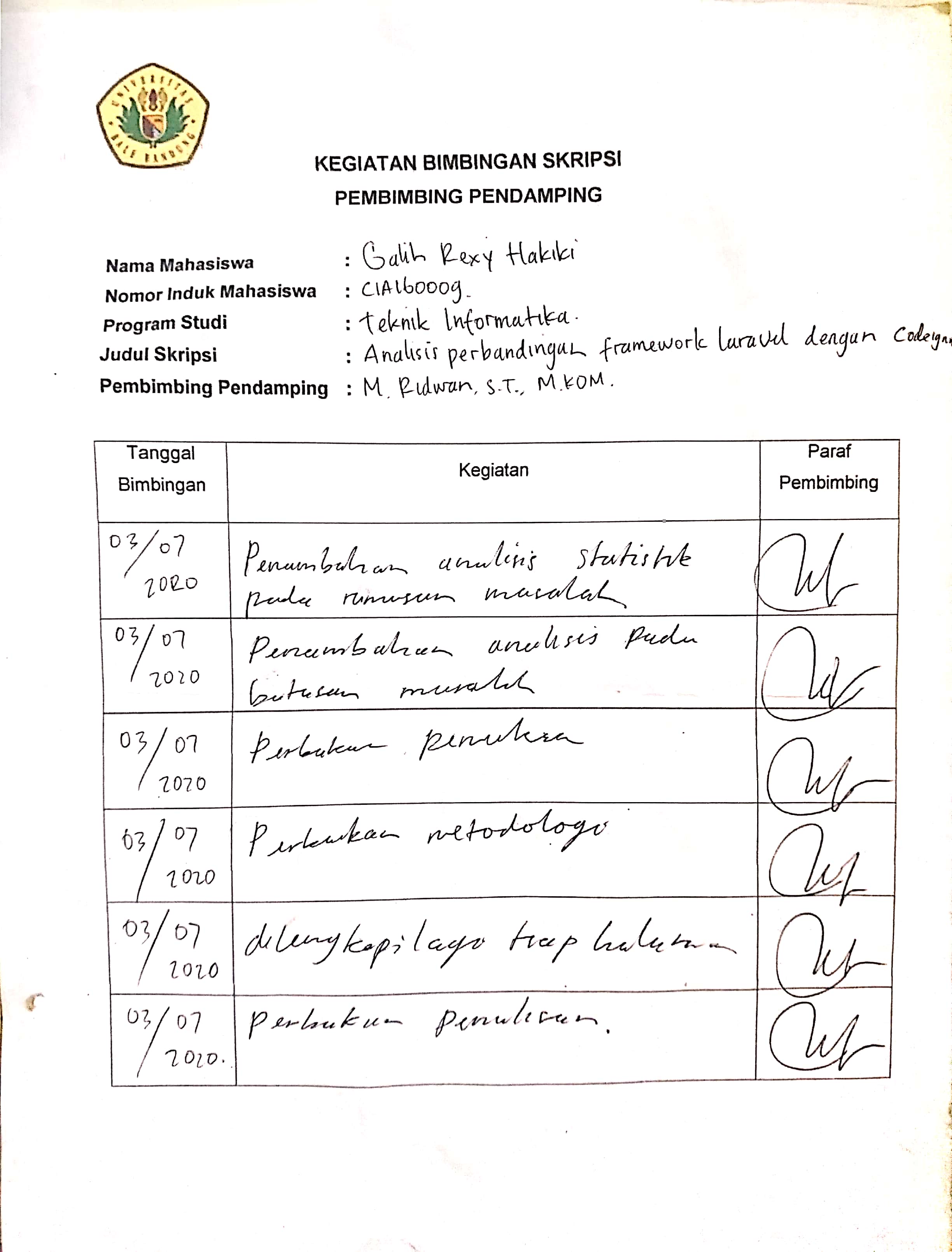
Dilaksanakan: Rabu 26-03-2020, Jam : 19:00

Dilaksanakan secara tatap muka langsung. Pertanyaanya adalah sebagai berikut

1. Apakah anda tau apa itu Framework ?
2. Apakah anda pernah menggunakan Framework ?
3. Apakah anda tau apa itu Framework Laravel ?
4. Apakah anda pernah menggunakan Framework Laravel ?
5. Apakah anda tau apa itu Framework Codeigniter ?
6. Apakah anda pernah menggunakan Framework Codeigniter ?
7. Apakah anda tau perbedaan di antara kedua Framework tersebut ?
8. Menurut anda mana yang lebih cepat ketika digunakan ?
9. Menurut anda mana yang lebih besar ukurannya ?
10. Menurut anda mana yang lebih lengkap dari segi cara akses database ?
11. Apakah anda tau mana yang lebih baik ?

**Lampiran 2 Bukti Kegiatan Bimbingan SKRIPSI pembimbing 1 dan pembimbing 2**





**Lampiran 3 Aplikasi dan Laporan SKRIPSI di Github.**

Aplikasi dengan menggunakan Laravel

https://github.com/GalihRexy/Diagnokids\_Laravel

Aplikasi dengan menggunakan Codeigniter

https://github.com/GalihRexy/Diagnokids\_CI

Laporan skripsi

https://github.com/GalihRexy/Laporan\_skripsi

**Lampiran 4 Source code aplikasi**

**Pengujian *execute time* pada Laravel (Proses tambah data anak)**

public function tambah\_data\_anak(Request $request)

{

$awal = microtime(true);

$nama = $request->input('nama');

$jenis\_kelamin = $request->input('jenis\_kelamin');

$umur = $request->input('umur');

$anak = new Anak;

$anak->email = session('email');

$anak->nama = $nama;

$anak->jenis\_kelamin = $jenis\_kelamin;

$anak->umur = $umur;

$anak->save();

$akhir = microtime(true);

$execute\_time = $akhir - $awal;

return redirect()->route('profile')->with('alert-success', 'Data berhasil ditambahkan...');

}

**Pengujian *execute time* pada Codeigniter (Proses tambah data anak)**

public function tambah\_data\_anak()

{

$awal = microtime(true);

$nama = $this->request->getPost('nama');

$jenis\_kelamin = $this->request->getPost('jenis\_kelamin');

$umur = $this->request->getPost('umur');

$data = [

'email' => session('email'),

'nama' => $nama,

'jenis\_kelamin' => $jenis\_kelamin,

'umur' => $umur,

];

$anakModel = new AnakModel();

$simpan = $anakModel->insert($data);

$akhir = microtime(true);

$execute\_time = $akhir - $awal;

session()->setFlashdata('alert-success', 'Data berhasil ditambahkan...');

return redirect()->to(base\_url('profile'));

}

**Pengujian *execute time* pada Laravel (Proses data diagnosa)**

public function index()

{

$awal = microtime(true);

if (session()->has('email')) {

$email = session('email');

$data['diagnosa'] = Diagnosa::where(['email' => $email])->orderBy('kode\_diagnosa', 'desc')->get();

$id = $data['diagnosa'][0]->id\_anak;

$akhir = microtime(true);

$execute\_time = $akhir - $awal;

var\_dump($execute\_time);die();

return view('data\_diagnosa', $data);

} else {

return redirect()->route('login')->with('alert-info', 'Silahkan login Terlebih dahulu...');

}

}

**Pengujian *execute time* pada Codeigniter (Proses data diagnosa)**

public function index()

{

$awal = microtime(true);

if(session()->has('email')){

$email = session('email');

$data['diagnosaModel'] = new DiagnosaModel();

$data['diagnosa'] = $data['diagnosaModel']->where(['email' => $email])->orderBy('kode\_diagnosa', 'desc')->findAll();

$akhir = microtime(true);

$execute\_time = $akhir - $awal;

var\_dump($execute\_time);die();

return view('data\_diagnosa', $data);

} else {

session()->setFlashdata('alert-info', 'Silahkan login terlebih dahulu...');

return redirect()->to(base\_url('auth'));

}

}

**Pengujian *execute time* pada Laravel (Proses registrasi)**

public function proses\_register(Request $request)

{

$awal = microtime(true);

$validation = $request->validate([

'nama' => 'required|max:150|string',

'email' => 'required|email|unique:user,email',

'password' => 'required|min:8',

'password2' => 'required|same:password',

],

[

'nama.required' => 'Harus diisi',

'email.required' => 'Harus diisi', 'email.unique' => 'email sudah digunakan',

'password.required' => 'Harus diisi', 'password.min' => 'Password minimal 8 digit',

'password2.required' => 'Harus diisi', 'password2.same' => 'Password tidak sama',

]

);

$user = new User;

$user->name = $request->input('nama');

$user->email = $request->input('email');

$user->password = password\_hash($request->input('password'), PASSWORD\_DEFAULT) ;

$user->save();

$akhir = microtime(true);

$execute\_time = $akhir - $awal;

return redirect()->route('login')->with('alert-success', 'Registrasi berhasil. Silahkan login...');

}

**Pengujian *execute time* pada Laravel (Proses registrasi)**

public function registrasi()

{

$awal = microtime(true);

$valid = $this->validate([

'nama' => [

'label' => 'Nama',

'rules' => 'required|trim',

'errors' => [

'required' => 'tidak boleh kosong',

]

],

'email' => [

'label' => 'Alamat Email',

'rules' => 'required|trim|valid\_email|is\_unique[user.email]',

'errors' => [

'required' => 'tidak boleh kosong',

'valid\_email' => 'alamat email tidak valid',

'is\_unique' => 'sudah digunakan',

]

],

'password' => [

'label' => 'Password',

'rules' => 'required|trim|min\_length[8]',

'errors' => [

'required' => 'tidak boleh kosong',

'min\_length' => 'terlalu pendek'

]

],

'password2' => [

'label' => 'Konfirmasi Password',

'rules' => 'required|matches[password]',

'errors' => [

'required' => 'tidak boleh kosong',

'matches' => 'password tidak sama'

]

],

]);

if (!$valid) {

$data['errors'] = \Config\Services::validation()->getErrors();

echo view('register', $data);

} else {

$userModel = new UserModel();

$nama = $this->request->getPost('nama');

$email = $this->request->getPost('email');

$pass = $this->request->getPost('password');

$pass2 = $this->request->getPost('password2');

$data = [

'name' => $nama,

'email' => $email,

'password' => password\_hash($pass, PASSWORD\_DEFAULT),

];

$simpan = $userModel->insert($data);

$akhir = microtime(true);

$execute\_time = $akhir - $awal;

session()->setFlashdata('alert-success', 'Berhasil Register! Silahkan login...');

return redirect()->to(base\_url('auth'));

}

}