

## PENGEMBANGAN SISTEM MANAJEMEN DOKUMENTASI REST API DI PT APPFUXION INDONESIA

Ananda Mukhammad Ikhsan<sup>1</sup>, Handaru Jati<sup>2</sup>

Universitas Negeri Yogyakarta

Email : <sup>1</sup>[Anandamukhammad.2017@student.uny.ac.id](mailto:Anandamukhammad.2017@student.uny.ac.id), <sup>2</sup> [handaru@uny.ac.id](mailto:handaru@uny.ac.id)

### ABSTRAK

Tujuan penelitian ini adalah: (1) Mengembangkan sistem manajemen dokumentasi REST API di PT Appfuxion Indonesia. (2) Menguji sistem manajemen dokumentasi REST API di PT Appfuxion Indonesia menggunakan pengujian functional suitability untuk uji kelayakan fungsi perangkat lunak dan pengujian usability untuk menguji kemudahan penggunaan sistem perangkat lunak oleh pengguna. Metode yang digunakan adalah penelitian dan pengembangan atau Research and Development (R&D) dengan prosedur pengembangan perangkat lunak yaitu model waterfall. Tahapan prosedur waterfall adalah komunikasi, perencanaan, pemodelan, konstruksi, dan penyerahan perangkat lunak kepada pelanggan/pengguna. Terdapat 2 responden ahli di bidang pengembangan perangkat lunak untuk pengujian functional suitability dan 9 responden dari karyawan PT Appfuxion Indonesia untuk pengujian functional usability. Metode pengumpulan data menggunakan observasi, wawancara dan kuesioner. Hasil dari penelitian ini adalah: (1) Perangkat lunak sistem manajemen dokumentasi REST API di PT Appfuxion Indonesia dikembangkan dengan model waterfall dan menggunakan framework Spring boot. (2) Hasil pengujian pada aspek functional suitability mendapatkan nilai 100% yang artinya semua fungsi yang dirancang berjalan baik dan "Sangat Layak" digunakan. Untuk hasil pengujian aspek usability yaitu aplikasi web acceptable atau diterima oleh pengguna, grade scale C, adjective rating good, dan SUS score percentile rank mendapatkan grade B artinya aplikasi web mudah digunakan.

**Kata kunci:** Aplikasi web, REST API, Dokumentasi, framework Spring boot, waterfall model.

### ABSTRACT

*The objectives of this research are: (1) To develop a REST API documentation management system at PT Appfuxion Indonesia. (2) To test the REST API documentation management system at PT Appfuxion Indonesia using functional suitability testing for software functionality feasibility and usability testing to assess the ease of use of the software system by users. The method used is research and development (R&D) with the software development procedure based on the waterfall model. The stages of the waterfall procedure include communication, planning, modeling, construction, and delivery of the software to the customer/user. The subjects for functional suitability testing are 2 expert respondents in the field of software development, while the subjects for usability testing consist of 9 respondents from PT Appfuxion Indonesia's employees. Data collection methods include observation, interviews, and questionnaires. The results of this study are: (1) The REST API documentation management system software at PT Appfuxion Indonesia was developed using the waterfall model and the Spring Boot framework. (2) The results of functional suitability testing achieved a score of 100%, indicating that all designed functions are working well and are "Highly Feasible" for use. The usability testing results show that the web application is acceptable to users, with a grade scale of C, an adjective rating of good, and a SUS score percentile rank achieving a grade of B, meaning the web application is easy to use.*

**Keyword:** Web application, REST API, documentation, waterfall model

## PENDAHULUAN

REST API (*Application Programming Interface dengan Representational State Transfer*) adalah salah satu teknologi yang digunakan dalam pengelolaan informasi. Standar REST API memungkinkan komunikasi antara aplikasi melalui jaringan internet atau intranet. Dengan menggunakan REST API, aplikasi dapat saling berinteraksi tanpa harus mengetahui bagaimana aplikasi lain dibangun dan dikelola.

Dalam pengembangan aplikasi, dokumentasi REST API merupakan hal yang sangat penting. Dokumentasi REST API menjelaskan bagaimana aplikasi tersebut digunakan, baik oleh *developer* maupun pengguna. Menurut Sohan [1], dokumentasi REST API dapat memberikan kemudahan dalam proses pengembangan perangkat lunak dimana di dalamnya terdapat spesifikasi dan contoh cara penggunaan sebuah aplikasi REST API [1]. Pengelolaan dokumentasi REST API sering menjadi masalah bagi organisasi. Hal ini dikarenakan dokumentasi yang tidak teratur dan tidak up to date dapat menyebabkan kesulitan dalam pengembangan aplikasi. Dokumentasi yang tidak dikelola dengan baik dapat menyebabkan kesalahpahaman dan kesalahan dalam pengembangan aplikasi.

Berdasarkan hasil observasi yang diperoleh di PT Appfuxion Indonesia, PT Appfuxion Indonesia belum melakukan pengelolaan dokumentasi REST API dari perangkat lunak yang mereka kembangkan. Berdasarkan wawancara

dengan developer di PT Appfuxion Indonesia, dokumentasi REST API dari perangkat lunak yang mereka kembangkan tidak disimpan di sebuah sistem terintegrasi dan akses kepada dokumentasi tersebut tidak dikelola. Selama ini untuk mengakses dokumentasi, developer harus meminta kepada developer lain yang memiliki atau mengetahui keberadaan dokumentasi tersebut. Developer harus bertanya kepada developer yang bertanggung jawab atas dokumentasi tersebut setiap ada pembaharuan spesifikasi. Hal tersebut kurang optimal dalam memberikan akses terhadap dokumentasi REST API dan dinilai kurang efektif oleh developer di PT Appfuxion Indonesia.

Berdasarkan latar belakang tersebut, perusahaan membutuhkan sistem perangkat lunak untuk mengelola dokumentasi REST API. Sistem ini dapat membantu dalam penyimpanan, pengelolaan, dan pemantauan dokumentasi REST API, sehingga dapat meningkatkan efisiensi dan kemudahan dalam pengembangan aplikasi yang menggunakan REST API. Sistem ini juga perlu mengimplementasikan OpenAPI agar dokumentasi API yang dikelola lebih terstruktur karena mengikuti standar yang sama. Analisis ini telah disampaikan ke developer PT Appfuxion dan telah disetujui untuk dikembangkan.

Berdasarkan hal itu, maka perlu diadakan pengembangan perangkat lunak web sebagai sistem manajemen pengelolaan dokumentasi REST API untuk keperluan PT Appfuxion Indonesia dengan judul penelitian

“PENGEMBANGAN SISTEM MANAJEMEN DOKUMENTASI REST API DI PT APPFUXION INDONESIA”. Perangkat lunak berbasis web dipilih karena jenis perangkat lunak tersebut dapat diakses dengan mudah melalui web browser. Dengan adanya perangkat lunak web ini diharapkan developer di PT Appfuxion Indonesia dapat mengelola dokumentasi REST API dengan baik dan mempermudah proses pengembangan perangkat lunak.

## METODE

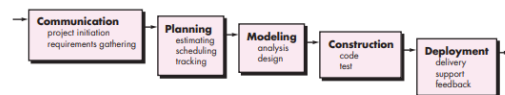
Pengembangan sistem manajemen dokumen REST API di PT. Appfuxion Indonesia menggunakan metode Penelitian dan Pengembangan atau *Research and Development* (R&D). Penelitian ini menghasilkan produk berupa perangkat lunak sistem manajemen dokumentasi REST API untuk mempermudah pengelolaan dan akses dokumentasi REST API. Pengembangan perangkat lunak ini menggunakan framework Springboot dan menggunakan proses pengembangan perangkat lunak *waterfall*.

### Sumber Data/Subjek Penelitian

Penelitian dan pengembangan Sistem Manajemen REST API di PT Appfuxion Indonesia melibatkan ahli dalam pengembangan perangkat lunak untuk uji *functional suitability*, serta karyawan/developer PT Appfuxion Indonesia untuk uji *usability*.

## Prosedur Pengembangan

Penelitian dan pengembangan ini menggunakan model *waterfall*. Pressman [2] mendefinisikan Model *Waterfall* sebagai pendekatan pengembangan perangkat lunak yang terstruktur dan berurutan. Terdapat lima tahapan dalam metode waterfall, meliputi: komunikasi (*communication*), perencanaan (*planning*), pemodelan (*modelling*), konstruksi (*construction*), dan penyerahan ke pengguna (*deployment*) [2].



Gambar 1. Model pengembangan perangkat lunak *waterfall*

### A. Komunikasi (*Communication*)

Komunikasi yang dilakukan peneliti bersama dengan developer di PT Appfuxion Indonesia. Observasi dan wawancara digunakan sebagai metode komunikasi pada tahap ini. Observasi dengan mengamati proses dokumentasi yang dilakukan di tiap pengembangan perangkat lunak di PT Appfuxion Indonesia. Kemudian dilakukan wawancara dengan developer di PT Appfuxion Indonesia yang bertujuan untuk mendapatkan informasi yang diperlukan untuk perangkat lunak yang dikembangkan.

Setelah semua informasi yang dibutuhkan cukup kemudian dilakukan analisis kebutuhan spesifik berupa perangkat lunak yang digunakan dan analisis fungsional untuk mengembangkan fitur pada perangkat lunak.

## B. Perencanaan (*Planning*)

Pada fase ini pengembang membuat rencana kerja atau rencana kerja. Ini terdiri dari waktu yang diperlukan untuk mengatur dan menganalisis persyaratan yang diperlukan dan mengembangkan produk setelah menguji produk jadi. Rencana kerja dibuat tabel langkah-langkah pekerjaan yang dilakukan serta waktu pembangunan proyek dan durasi pekerjaan.

## C. Pemodelan (*Modelling*)

Langkah selanjutnya adalah pemodelan. Langkah ini dilakukan setelah analisis kebutuhan selesai. Langkah pertama yang dilakukan adalah desain antarmuka pengguna, yaitu gambaran umum perancangan sistem perangkat lunak yang digunakan untuk merancang menu-menu yang ada sesuai dengan analisis kebutuhan. Kemudian mendesain diagram unified modeling language untuk menggambarkan aliran sistem yang dikembangkan dan desain basis data berupa entity relationship diagram untuk menggambarkan relasi antar tabel pada basis data.

## D. Konstruksi (*Construction*)

Konstruksi adalah fase inti dari pengembangan perangkat lunak. Konstruksi mengontrol arsitektur dan logika sistem secara keseluruhan, yang kemudian diimplementasikan dalam bahasa pemrograman. Pada langkah ini, kode program ditulis di dalam text editor untuk mengembangkan fitur-fitur yang sudah direncanakan.

Penelitian ini menggunakan framework Springboot dan menggunakan

basis data adalah PostgreSQL. Pengujian juga dilakukan pada tahap ini, aspek akan diuji adalah functional suitability dan usability. Tujuan pengujian adalah untuk menemukan bug pada perangkat lunak dan menguji kualitas produk yang diproduksi.

## E. Pendistribusian (*Deployment*)

Sistem manajemen dokumentasi REST API yang sudah dikembangkan kemudian diimplementasikan dan diberikan kepada pengguna. Produk diberikan kepada PT Appfuxion Indonesia. Perangkat lunak yang telah diserahkan kepada PT Appfuxion Indonesia kemudian akan dipergunakan untuk mendukung proses pengembangan perangkat lunak dalam hal manajemen dokumentasi REST API.

## Teknik Pengumpulan Data

### A. Observasi

Observasi dilakukan dengan memantau kegiatan pengembangan perangkat lunak di PT Appfuxion Indonesia. Peneliti dilibatkan langsung dalam memantau aktivitas karyawan dalam proses pengembangan perangkat lunak di PT Appfuxion Indonesia.

### B. Wawancara

Proses wawancara dilakukan dengan mengajukan pertanyaan, klarifikasi, dan informasi dengan mewawancarai karyawan di PT Appfuxion Indonesia sebagai bahan penelitian yang akan digunakan.

### C. Kuesioner

Kuesioner adalah rangkaian pertanyaan yang harus dijawab oleh subjek penelitian sebagai data yang akan

digunakan. Responden dapat menjawab sesuai dengan keadaan tanpa pengaruh dari orang lain. Kuesioner digunakan untuk menguji aspek *functional suitability* dan aspek *usability*. Peneliti membagikan kuesioner dengan beberapa jawaban. Kuesioner *functional suitability* diisi oleh para ahli di bidang perangkat lunak, dan kuesioner *usability* diisi oleh karyawan PT Appfuxion Indonesia.

### Instrumen Penelitian

Terdapat dua instrumen yang digunakan dalam penelitian ini, yaitu instrumen *functional suitability* dan instrumen *usability*.

Instrumen *functional suitability* yang digunakan adalah angket yang berisi *test case* untuk menguji fungsi pada perangkat lunak. *Test Case* adalah serangkaian langkah yang diambil untuk memverifikasi fitur atau fungsionalitas tertentu dari perangkat lunak [3]. Formulir pengujian *functional suitability* berisi daftar fungsional perangkat lunak, yang dijabarkan dari prosedur penggunaan perangkat lunak. Instrumen akan menggunakan skala Guttman yang memiliki pilihan “Ya” atau “Tidak”. Kisi-kisi instrumen untuk menguji aspek *functional suitability* terlihat seperti tabel 1.

Tabel 1. Kisi-kisi Instrumen *Functional Suitability*

No	Indikator	No. Butir
1	<i>Functional Completeness</i>	1 – 21
2	<i>Functional Correctness</i>	22 – 32
3	<i>Functional Appropriateness</i>	33 – 40

Instrumen Usability menggunakan kuesioner yang mengimplementasikan *System Usability Scale* (SUS). Kuesioner ini memungkinkan pengukuran aspek *usability* suatu perangkat lunak dari subjektivitas pengguna [4]. Kuesioner ini terdiri dari sepuluh pertanyaan [4] dan kuesioner ini menggunakan lima poin skala likert dengan pilihan jawaban 1 poin untuk Sangat Tidak Setuju (STS), 2 poin untuk Tidak Setuju (TS), 3 poin untuk Netral (N), 4 poin untuk Setuju (S), dan 5 poin untuk Sangat Setuju (SS). [5]

### Teknik Analisis Data

Tahapan ini merupakan penjelasan bagaimana data terkumpul akan diolah dan dianalisis berdasarkan perangkat penelitian yang digunakan. Teknik analisis data untuk pengujian aspek *functional suitability* dan *usability* dapat dijelaskan sebagai berikut.

#### A. Analisis Kualitas Karakteristik *Functional Suitability*

Analisis uji kualitatif karakteristik *functional suitability* dilakukan menurut metode *test case*, setelah itu digunakan skala Guttman untuk evaluasi. Menurut [2] instrumen yang menggunakan skala Guttman harus memiliki jawaban yang tegas dan konsisten, misalnya “Ya” atau “Tidak”. Jawaban “Ya” akan diberi skor 1 sedangkan jawaban “Tidak” akan diberi skor 0. Berikut rumus untuk mendapatkan persentasi kelayakan:

Persentasi Kelayakan (%) =

$$\frac{\text{Skor yang didapatkan}}{\text{Skor maksimal}} \times 100\%$$

Hasil persentasi diatas kemudian dikonversikan seperti pada tabel 2.

Tabel 2. Interpretasi Presentasi Hasil Pengujian Functional Suitability

No	Persentasi	Interpretasi
1	0% - 20%	Sangat Tidak Layak
2	21% - 40%	Tidak Layak
3	41% - 60%	Cukup Layak
4	61% - 80%	Layak
5	81% - 100%	Sangat Layak

#### B. Analisis Kualitas Karakteristik Usability

Pengujian karakteristik *usability* dilakukan menggunakan kuesioner *System Usability Scale* (SUS) dengan skala Likert lima tingkat: Sangat Tidak Setuju (STS), Tidak Setuju (TS), Netral (N), Setuju (S), dan Sangat Setuju (SS). Setiap pernyataan dalam kuesioner memiliki nilai kontribusi. Setiap item memiliki kontribusi 1-5. Untuk item 1, 3, 5, 7, dan 9, skor kontribusinya dihitung dengan mengurangi 1 dari posisi skala. Sedangkan untuk item 2, 4, 6, 8, dan 10, skor kontribusinya diperoleh dengan mengurangi posisi skala dari 5. Jumlah total skor kontribusi kemudian dikalikan dengan 2,5 untuk mendapatkan nilai keseluruhan kegunaan sistem. Skor SUS berada dalam rentang 0 hingga 100 (Brooke, 1996). Berikut rumus perhitungan skor SUS:

$$\text{Skor SUS} = ((R1 - 1) + (5 - R2) + (R3 - 1) + (5 - R4) + (R5 - 1) + (5 - R6) + (R7 - 1) + (5 - R8) + (R9 - 1) + (5 - R10)) * 2.5$$

Dengan:

$R$  = item ke-

Total skor SUS dapat mengindikasikan tingkat penerimaan pengguna terhadap perangkat lunak yang diuji. Skor SUS harus di atas 70 untuk masuk dalam kategori *Acceptable* [5]. Skor SUS dianggap *Good* jika di atas 70,4 [7].

#### HASIL DAN PEMBAHASAN

Penelitian ini bertujuan untuk mengembangkan sistem manajemen REST API untuk mengelola dokumentasi REST API di PT Appfuxion Indonesia. Pengembangan sistem manajemen dokumentasi REST API menggunakan model waterfall yang tahapannya terdiri dari *communication*, *planning*, *modelling*, *construction*, dan *deployment*. Pengujian fitur dilakukan dengan dua cara, yaitu uji *functional suitability* oleh ahli pengembangan perangkat lunak dan uji *usability* oleh karyawan/developer PT Appfuxion.

##### A. Komunikasi (*Communication*)

Proses tahapan ini melalui wawancara dengan developer PT Appfuxion Indonesia untuk mendapatkan kebutuhan fungsional dari sistem manajemen dokumentasi REST API yang akan dikembangkan. Hasil wawancara dengan developer PT Appfuxion Indonesia adalah sebagai berikut:

1. Saat ini developer PT Appfuxion Indonesia kesulitan mengelola akses dokumentasi rest api dari sistem-sistem yang pernah mereka kembangkan.
2. Saat ini PT Appfuxion Indonesia belum memiliki sistem untuk mengelola prubahan dokumentasi REST API.

3. PT Appfuxion Indonesia memerlukan cara untuk membagikan dokumentasi ke pihak lain dalam format PDF dan JSON.
4. PT Appfuxion Indonesia memerlukan cara untuk bisa melakukan pencarian dokumentasi dengan mudah.
5. PT Appfuxion Indonesia memerlukan fitur untuk menguji dokumentasi rest api yang telah dibuat.

Hasil wawancara tersebut kemudian dianalisis untuk mendapatkan kebutuhan fungsional yang akan diterapkan di sistem manajemen dokumentasi REST API yang akan dikembangkan. Analisis kebutuhan fungsional berdasarkan hasil wawancara tersebut adalah sebagai berikut:

1. Fitur untuk mengatur akses pengguna untuk masuk ke sistem.
2. Fitur menu dan *breadcrumb* untuk navigasi di dalam sistem.
3. Fitur untuk mengelola data pengguna.
4. Fitur untuk menambah, mengubah, dan menghapus dokumentasi.
5. Fitur untuk melakukan pencarian dokumentasi.
6. Fitur untuk melakukan pengujian dokumentasi.
7. Fitur untuk melakukan ekspor dokumentasi menjadi dokumen PDF dan JSON.

#### B. Perencanaan (*Planning*)

Perencanaan jadwal pengerjaan proyek dapat dilihat pada tabel berikut.

Tabel 3. Tabel Perencanaan Jadwal Pengerjaan Proyek

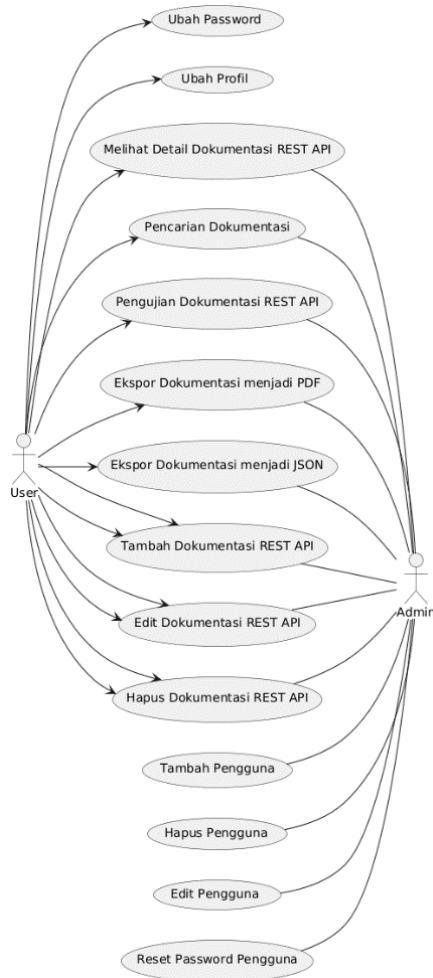
No	Nama Kegiatan	Durasi
1	Analisis Kebutuhan	1 Minggu
2	Pembuatan UML	2 Minggu
3	Pembuatan desain <i>database</i>	1 Minggu
4	Implementasi <i>Database</i>	1 Hari
5	Pengerjaan <i>frontend</i>	2 Minggu
6	Pengerjaan <i>backend</i>	2 Minggu
7	Pengujian <i>functional suitability</i>	3 Hari
8	Pengujian <i>usability</i>	3 Hari
9	<i>Deployment</i>	1 Hari

#### C. Pemodelan (*Modelling*)

Pemodelan sistem manajemen REST API ini menggunakan UML. Adapun jenis UML yang digunakan yaitu *use case diagram*, *activity diagram*, dan *sequence diagram*.

##### 1. Desain *Use Case Diagram*

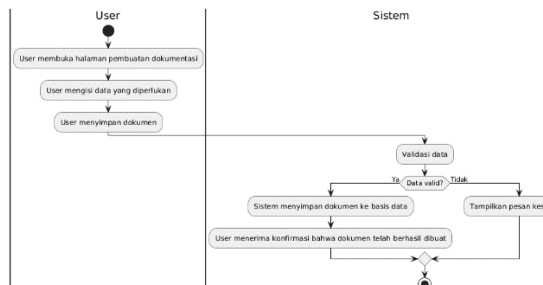
Diagram *use case* adalah pemodelan untuk tingkah laku (*behaviour*) sistem yang menunjukkan interaksi antara aktor dan perangkat lunak yang dibuat. Diagram *use case* untuk pengembangan sistem manajemen REST API ini, dapat diamati pada gambar 2 berikut.



Gambar 2. Desain Use Case Diagram

## 2. Activity Diagram

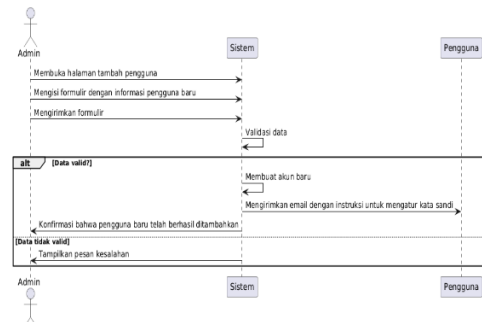
Activity diagram mendeskripsikan alur kerja dari sebuah perangkat lunak. Gambar 3 merupakan activity diagram untuk menambah data dokumentasi baru.



Gambar 3. Activity Diagram Proses Tambah Dokumentasi

## 3. Sequence Diagram

Diagram sekuens menggambarkan perilaku objek dalam use case dengan menunjukkan masa hidup objek dan pesan yang dikirim serta diterima di antara objek-objek tersebut. Gambar 4 di bawah merupakan sequence diagram untuk menambah pengguna baru.

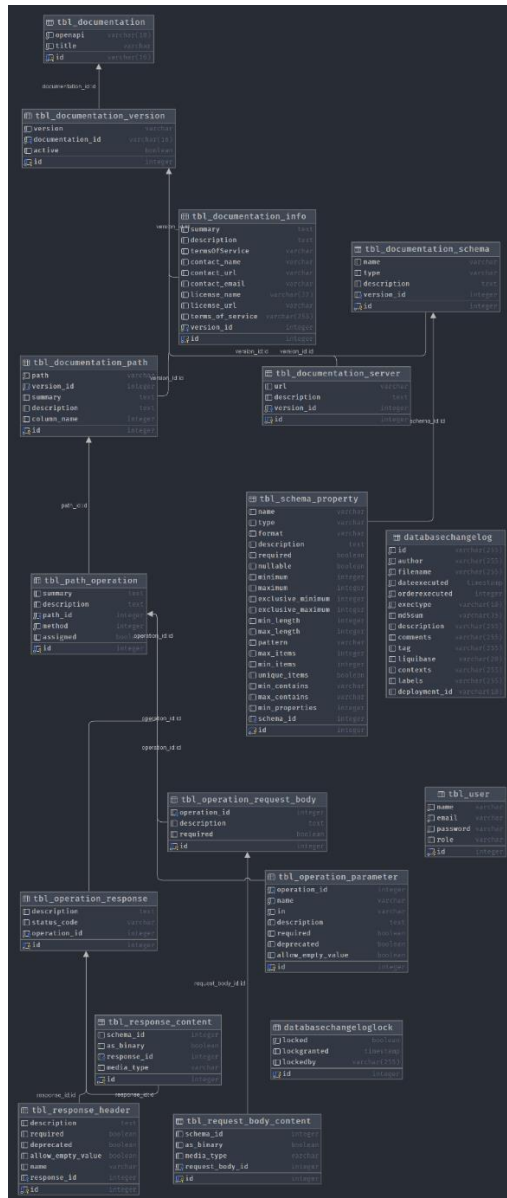


Gambar 4. Sequence diagram proses tambah pengguna

## 1. Desain Database

Desain database yang akan dikembangkan dapat dilihat pada gambar 5 berikut ini:





Gambar 5. Desain database

#### D. Konstruksi (Construction)

Tahapan konstruksi terdiri dari dua tahapan yaitu proses pembuatan dan pengujian perangkat lunak.

##### 1. Pembuatan Perangkat Lunak

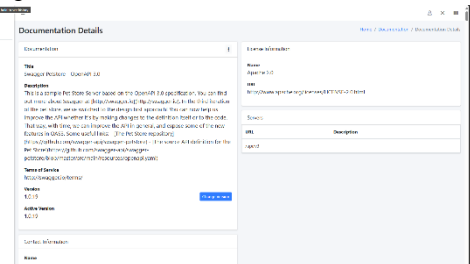
Tahap pertama pada proses pembuatan perangkat lunak adalah inisialisasi *framework* Spring boot. Inisialisasi dilakukan dengan mengakses <https://start.spring.io>, kemudian

mengisikan informasi proyek dan *dependency* yang akan digunakan, lalu menekan tombol *generate*.

Setelah proses inisialisasi selesai, langkah berikutnya adalah pembuatan halaman depan (*frontend*) perangkat lunak. Halaman depan perangkat lunak dibuat menggunakan HTML dan JavaScript untuk mendesain tampilan dan interaksi data dengan *backend*. Dalam sistem ini, halaman depan yang dibuat meliputi:

- Halaman tambah dan ubah dokumentasi
- Halaman detail dokumentasi
- Halaman daftar dokumentasi
- Halaman *login*
- Halaman tambah pengguna
- Halaman ubah pengguna
- Halaman reset *password* pengguna

Salah contoh tampilan halaman depan yang telah dibuat dapat dilihat pada gambar 6 berikut.



Gambar 6. Halaman detail dokumentasi

Setelah semua halaman depan selesai, tahap selanjutnya adalah pengimplementasian *database*. Tahapan ini merupakan proses implementasi rancangan *database* menggunakan PostgreSQL. Database yang diimplementasikan pada tahap ini mengacu pada desain *database* yang

telah dibuat pada tahap pemodelan (*modelling*).

Framework Spring boot menggunakan *entity* untuk merepresentasikan tiap tabel *database* ke dalam bentuk *code* sehingga manipulasi data pada *database* dapat lebih mudah. *Entity* merupakan sebuah *class* berisi *property* berdasarkan kolom yang ada pada tabel. Dalam *entity* juga terdapat *class property* yang mengatur relasi antar tabel. Contoh *class entity* dapat dilihat pada gambar 7.

```
@Table(name = "tbl_operation_response")
public class OperationResponseEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id", nullable = false)
    private Integer id;

    @Column(name = "description")
    @Type(type = "org.hibernate.type.TextType")
    private String description;

    @Column(name = "status_code")
    private String statusCode;

    @NotNull
    @ManyToOne(fetch = FetchType.LAZY, optional = false)
    @JoinColumn(name = "operation_id", nullable = false)
    private PathOperationEntity operation;

    @OneToMany(mappedBy = "response", cascade = CascadeType.ALL)
    private Set<ResponseContentEntity> content = new LinkedHashSet<>();

    @OneToMany(mappedBy = "response", cascade = CascadeType.ALL)
    private Set<ResponseHeaderEntity> headers = new LinkedHashSet<>();
}
```

Gambar 7. *Entity class* untuk *tbl\_operation\_response*

Tahap selanjutnya adalah pembuatan *controller*. *Controller* dalam *framework* Spring boot digunakan untuk menangkap HTTP *request api* yang didukung kemudian memproses *request* tersebut untuk dikembalikan lagi melalui HTTP *response*. *Controller class* berisi fungsi-fungsi yang merepresentasikan *endpoint* untuk memproses HTTP *request* dan mengembalikan HTTP *response* yang sesuai. *Data controller* yang dibuat dapat dilihat pada tabel 4.

Tabel 4. Daftar *Controller*

No	Controller	Endpoint
1	<i>Auth Controller</i>	/login
2	<i>Documentation Controller</i>	/documentation /documentation/add /documentation/edit /documentation/detail /documentation/delete
3	<i>User Controller</i>	/user/add /user/ /user/edit /user/delete /user/changePassword

Contoh kode dari *controller class* *AuthController.java* seperti terlihat di gambar 8.

```
@RestController
@RequestMapping("/auth")
public class AuthController {

    @Autowired
    private AuthenticationManager authenticationManager;

    @Autowired
    private AuthService service;

    @Autowired
    private TokenProvider tokenProvider;

    @PostMapping(value = "/login", produces = "application/json", consumes = "application/json")
    public AuthLoginResponse login(@RequestBody AuthLoginRequest request) {
        AuthLoginResponse response = new AuthLoginResponse();
        var usernamePassword = new UsernamePasswordAuthenticationToken(request.getBody().getEmail(), request.getBody().getPassword());
        var authUser = authenticationManager.authenticate(usernamePassword);
        var accessToken = tokenProvider.generateAccessToken(authUser.getPrincipal());
        response.setResponseBody(accessToken);
        ResponseHeader header = new ResponseHeader();
        header.setSuccess(true);
        response.setResponseHeader(header);
        return response;
    }
}
```

Gambar 8. Kode class *AuthController.java*

## 2. Pengujian Perangkat Lunak

Pada tahap ini pengujian perangkat lunak dilakukan agar perangkat lunak yang dikembangkan dapat memenuhi standar yang baik. Terdapat dua pengujian yang akan dilakukan. Pengujian pertama yaitu pengujian *alpha* yaitu pengujian dari sisi pengembang perangkat lunak yang menggunakan aspek *functional suitability*. Pengujian kedua adalah pengujian beta yaitu

pengujian dari sisi pengguna dengan menggunakan aspek *usability*.

a. Pengujian Aspek *Functional Suitability*

Pada pengujian aspek *Functional Suitability*, responden adalah dua orang ahli dalam bidang pengembangan perangkat lunak. Tujuan pengujian ini adalah untuk mengetahui fungsi-fungsi dalam perangkat lunak yang telah dikembangkan sudah berjalan dengan benar atau belum.

Hasil pengujian ini menerima skor sebesar 80 dari total 80 skor maksimal. Dari perolehan skor tersebut, kemudian dilakukan perhitungan persentase kelayakan untuk menghasilkan persentase kelayakan aspek *functional suitability*. Berikut perhitungan persentase kelayakan pada pengujian *functional suitability*.

Persentase (%)

$$= \frac{\text{Skor yang didapatkan}}{\text{Skor maksimal}} \times 100\%$$

$$\text{Persentase (\%)} = \frac{80}{80} \times 100\%$$

$$\text{Persentase (\%)} = 100\%$$

Persentase kelayakan yang diperoleh dari hasil pengujian aspek *functional suitability* adalah 100%. Dapat disimpulkan bahwa semua fungsi perangkat lunak sudah berjalan dengan baik. Hasil konversi kelayakan menurut Sudaryono [6] berdasarkan persentase kelayakan yang dihasilkan adalah "Sangat Layak" [6].

b. Pengujian Aspek *Functional Suitability*

Pengujian ini menggunakan system usability scale (SUS) yang berisi 10 pertanyaan. Jumlah responden yang berpartisipasi mengisi instrumen

pengujian adalah 9 orang software developer PT Appfuxion Indonesia. Hasil pengujian aspek *usability* dapat dilihat pada gambar berikut.

Skor Hasil Hitung										Jumlah	Nilai (Jumlah x 2,5)
Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10		
2	3	3	4	3	2	4	4	4	4	33	82,5
3	4	4	3	2	2	4	3	3	3	31	77,5
2	3	3	2	3	3	2	3	4	3	28	70
3	3	4	3	3	3	3	3	2	3	30	75
4	2	4	2	4	4	4	4	4	3	35	87,5
3	2	4	2	3	3	3	3	3	4	30	75
2	3	2	3	3	2	3	4	3	3	28	70
3	2	3	1	4	3	3	4	4	4	31	77,5
3	3	3	3	2	3	2	3	3	3	28	70
Skor Rata-rata (Hasil Akhir)										76,11111111	

Gambar 9. Hasil Uji SUS

Berdasarkan data yang diperoleh pengujian aspek *usability* menghasilkan skor rata-rata 76,1. Terdapat dua cara yang dapat digunakan untuk menentukan hasil penilaian berdasarkan rata-rata skor yang diperoleh. Cara penentuan yang pertama adalah dengan *acceptability*, *grade scale*, *adjective rating* dari tingkat penerimaan. Cara penentuan kedua adalah dengan SUS score percentile rank.

Penilaian *Acceptability*, *Grade Scale*, dan *Adjective rating* ini menentukan sejauh mana perspektif pengguna dalam bentuk dapat diterima atau tidak dapat diterima, skala nilai A – F, dan interpretasi dalam bentuk skala sangat buruk hingga sangat baik [7].

Hasil skor SUS yang diperoleh adalah 76,1 jadi hasil penilaian dari sistem manajemen dokumentasi REST API di PT Appfuxion Indonesia adalah:

- Tingkat penerimaannya adalah dapat diterima,
- Skala nilainya adalah C,
- Skala interpretasi nya adalah Baik

Sesuai dengan hasil penilaian tersebut maka sistem manajemen dokumentasi REST API di PT Appfuxion

Indonesia tergolong dapat diterima dan digunakan dengan mudah oleh pengguna.

Sedangkan untuk penilaian SUS *score percentile rank* hanya untuk membuat tingkat kegunaan berdasarkan perhitungan umum untuk mengkategorikan tingkat skor menjadi tingkat A, B, C, D, E, dan F. Ketentuan untuk penilaian SUS *score percentile rank* adalah sebagai berikut:

- a) *Grade A*: skor lebih besar atau sama dengan 80,3
- b) *Grade B*: skor lebih besar sama dengan 74 dan lebih kecil dari 80,3
- c) *Grade C*: skor lebih besar 68 dan lebih kecil dari 74
- d) *Grade D*: skor lebih besar sama dengan 51 dan lebih kecil dari 68
- e) *Grade F*: skor lebih kecil dari 51

Hasil skor SUS yang diperoleh adalah 76,1 jadi hasil SUS *score percentile rank* ada pada grade B. Dapat disimpulkan bahwa sistem manajemen dokumentasi REST API di PT Appfuxion Indonesia dapat digunakan dengan mudah oleh pengguna.

#### E. Penyerahan Perangkat Lunak kepada Pengguna (*Deployment*)

Setelah pengujian dilakukan dan mendapatkan hasil bahwa perangkat lunak berkualitas dan mudah digunakan, maka selanjutnya proses terakhir dari pengembangan sistem manajemen perangkat lunak di PT Appfuxion Indonesia adalah proses *deployment*. Proses ini dilakukan dengan cara penyerahan *source code* dan *database* kepada pihak PT Appfuxion Indonesia

yang kemudian akan dijalankan di infrastruktur yang tersedia di PT Appfuxion Indonesia.

## SIMPULAN

Berdasarkan uraian hasil penelitian dan pembahasan yang diperoleh, dapat diambil kesimpulan sebagai berikut.

1. Sistem manajemen dokumentasi REST API telah dikembangkan untuk memfasilitasi manajemen dokumentasi REST API di PT Appfuxion Indonesia. Sistem ini dibuat untuk memudahkan *developer* dalam mempublikasi dan mengakses dokumentasi REST API di PT Appfuxion Indonesia. Fitur yang telah dikembangkan sudah disesuaikan dengan kebutuhan pengguna dan telah diuji aspek fungsionalitas dan kemudahan penggunaannya. Pengembangan sistem manajemen dokumentasi REST API menggunakan model *waterfall* yang tahapannya terdiri dari *communication*, *planning*, *modelling*, *construction*, dan *deployment*. Aplikasi dikembangkan dengan menggunakan *framework* Spring boot.
2. Untuk menjamin kualitas sistem manajemen dokumentasi REST API yang dikembangkan bagi PT Appfuxion Indonesia, telah dilakukan pengujian komprehensif yang mencakup aspek fungsional (*functional suitability*) dan kemudahan penggunaan (*functional usability*). Hasil pengujian aspek *functional suitability* yang digunakan

untuk memeriksa kelayakan fungsionalitas perangkat lunak mendapatkan hasil 100%, yang artinya seluruh fungsi yang dirancang dan dikembangkan dapat berjalan dengan baik dan berkualitas. Hasil pengujian aspek *usability* yang digunakan untuk menguji kemudahan penggunaan perangkat lunak mendapatkan hasil dari penilaian *Acceptability, Grade Scale, Adjective Rating* yaitu dapat diterima, skala nilai C, dan dapat diinterpretasikan dengan skala Baik dan *SUS score percentile rank* mendapatkan skala B yang menunjukkan bahwa sistem manajemen dokumentasi REST API di PT Appfuxion Indonesia mudah digunakan oleh pengguna.

#### Saran Pengembangan

1. Perlu ditambahkan fitur manajemen hak akses untuk semua halaman agar akses dokumentasi menjadi lebih aman sesuai dengan hak akses pengguna.
2. Perlu adanya fitur log aktivitas pengguna untuk memantau perubahan data agar setiap perubahan data tercatat dalam sistem.

#### DAFTAR RUJUKAN

- [1] S. M. Sohan, F. Maurer, C. Anslow, and M. P. Robillard, "A study of the effectiveness of usage examples in REST API documentation," in Proc. IEEE Symp. Visual Languages and Human-Centric Computing (VL/HCC), Oct. 2017, pp. 53–61.
- [2] R. S. Pressman, *Rekayasa Perangkat Lunak: Pendekatan Praktisi*, Buku 1. Yogyakarta, Indonesia: Andi, 2015.
- [3] A. Bala and R. Chhillar, "Automatic test data generation using genetic algorithm using sequence diagram," *Int. J. Comput. Syst.*, vol. 3, no. 4, pp. 131–138, 2016.
- [4] J. Brooke, *SUS: A Quick and Dirty Usability Scale*. United Kingdom: Redhatch Consulting Ltd., 1996.
- [5] J. Brooke, "SUS: A retrospective," *J. Usability Stud.*, vol. 8, no. 2, pp. 29–40, 2013.
- [6] Sudaryono, *Metodologi Riset di Bidang TI (Panduan Praktik Teori dan Contoh)*. Yogyakarta, Indonesia: Andi, 2014.
- [7] A. Bangor, P. Kortum, and J. Miller, "Determining what individual SUS scores mean: Adding an adjective rating scale," *J. Usability Stud.*, vol. 4, pp. 114–123, 2009.