Jurnal Teknik Informatika (JUTIF)
Vol. 3, No. 2, April 2022, hlm. 395-303

DOI: https://doi.org/10.20884/1.jutif.2022.3.2.213
p-ISSN: 2723-3863
e-ISSN: 2723-3871

RESTFUL WEB SERVICE IMPLEMENTATION USING SPRING FRAMEWORK IN ROOM ASSETS MANAGEMENT SYSTEM

Syahren Aulia Achsan[*1], Yerymia Alfa Susetyo[2]

[1,2]Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Indonesia
Email: [1]672018190@student.uksw.edu, [2]yeremia.alfa@uksw.edu

(Naskah masuk: 14 Maret 2022, Revisi: 18 Maret 2022, Diterbitkan: 25 April 2022)

Abstract

Room is an asset owned by every agency and company. The ability to process information about data of an asset is very important in order to improve performance and efficiency of an agency or company. Good room management skills are needed by every agency and company. Manual room management activities that are still carried out conventionally with paper media are considered less effective and efficient,, so that room management system is needed. In this research, a room management system is created in order to solve that problem. The room management system is built by implementing RESTful web service using Spring framework. RESTful web service is a technology that can be used to perform system integration and application development with multiplatform base using REST architecture. This research produces a room management system that is can used by admin and user to do room management activities such as borrowing the room, and others.

Keywords: *RESTful web service, Room, Spring*

PENERAPAN *RESTFUL WEB SERVICE* DENGAN *FRAMEWORK SPRING* PADA SISTEM PENGELOLAAN ASET RUANG

Abstrak

Ruang adalah aset yang dimiliki oleh setiap instansi dan perusahaan. Kemampuan mengolah informasi mengenai data suatu aset sangatlah penting guna untuk memperbaiki kinerja dan efisiensi suatu instansi atau perusahaan. Kemampuan pengelolaan ruang yang baik dibutuhkan pada setiap instansi dan perusahaan. Kegiatan pengelolaan ruang secara manual yang masih dilakukan secara konvensional dengan media kertas dinilai kurang efektif dan efisien, sehingga dibutuhkan sistem pengelolaan ruang. Pada penelitian ini, dibuat sistem pengelolaan ruang untuk mengatasi permasalahan tersebut. Sistem pengelolaan ruang dibangun dengan menerapkan *RESTful web service* menggunakan *framework Spring*. *RESTful web service* merupakan teknologi yang dapat digunakan untuk melakukan integrasi sistem dan pengembangan aplikasi dengan basis *multiplatform* menggunakan arsitektur REST. Penelitian ini menghasilkan sistem pengelolaan ruang yang dapat digunakan oleh admin dan *user* untuk melakukan kegiatan pengelolaan ruang seperti peminjaman, dan lain-lain.

Kata kunci: *RESTful web service, Ruang, Spring*

1. PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi menyebabkan manusia terus berinovasi untuk memenuhi kebutuhannya. Teknologi informasi merupakan alat bantu manusia untuk memproses kegiatan yang menghasilkan informasi. Selain menghasilkan informasi, teknologi informasi juga berfungsi memecahkan suatu masalah, membuka kreativitas, meningkatkan efektivitas dan efisiensi pada aktivitas manusia [1].

Organisasi dan instansi saat ini sedang menerapkan teknologi informasi ke berbagai kegiatan mereka, salah satunya sistem yang digunakan untuk mempermudah kegiatan pengelolaan aset-aset pada instansi tersebut. Aset merupakan barang atau benda yang terdiri dari benda yang bersifat bergerak dan benda yang bersifat tidak bergerak yang tercakup dalam kekayaan suatu instansi. Kebutuhan informasi mengenai data suatu aset sangatlah penting guna untuk memperbaiki kinerja atau efisiensi di dalam suatu instansi atau lembaga. Ruang adalah salah satu aset yang harus dikelola oleh sebuah instansi [2].

Instansi dengan jumlah ruangan yang banyak seperti sekolah, universitas, dan perusahaan besar

memiliki kebutuhan akan kemampuan pengelolaan yang baik untuk mengelola aset ruang yang dimilikinya. Dengan banyaknya ruang dan banyaknya pengguna yang memiliki kebutuhan pada hal tersebut, maka akan menjadi kesulitan jika untuk melakukan kegiatan seperti peminjaman dan penjadwalan ruang masih dilakukan secara manual dengan media kertas dan calon pengguna ruangan harus mendatangi bagian administrasi untuk nantinya dapat melakukan kegiatan pengelolaan ruang seperti peminjaman ruang dan lain-lain. Sebagai contoh proses peminjaman ruang dilakukan dengan cara calon pengguna mendatangi bagian administrasi dan mencari tahu ruangan yang memiliki fasilitas yang sesuai dengan kebutuhan pengguna, kemudian mengurus segala kebutuhan data dengan mengisi form pengajuan peminjaman. Setelah itu bagian administrasi melakukan pengecekan ruang pada tanggal yang ditentukan apakah ruang dapat dipinjam atau tidak dan melakukan konfirmasi kepada calon pengguna apakah ruang bisa dipinjam atau tidak, seterusnya hingga ruangan selesai digunakan [3].

Dalam melakukan kegiatan peminjaman akan menjadi kendala jika calon pengguna sebelumnya tidak memiliki informasi mengenai ruangan tersedia yang dapat dipinjam serta informasi mengenai fasilitas-fasilitas ruangan yang akan dipinjam. Hambatan lain yang mungkin terjadi adalah bentrokan jadwal ruang yang ingin dipesan. Ruangan adalah aset yang selalu digunakan pada instansi-instansi tersebut, sehingga status dan informasi mengenai ruangan dapat berubah sewaktu-waktu. Dalam melakukan kegiatan pengelolaan seperti pemesanan dan peminjaman akan dibutuhkan data-data yang diambil dari sistem lain seperti calon pengguna ruangan yang akan diambil dari sistem sumber daya manusia pada instansi tersebut. Maka dari itu dibutuhkan adanya integrasi dari data-data tersebut baik dari data ruangan itu sendiri hingga data yang terpisah dengan ruangan seperti calon pengguna (anggota dari instansi) [4].

Untuk menyelesaikan masalah tersebut dibutuhkan sistem yang terintegrasi dan dapat diakses dari berbagai *platform*. Sistem pengelolaan berbasis *website* konvensional akan sulit untuk diterapkan pada kasus ini karena sifatnya yang saling tergantung antara satu sistem dengan yang lain. Ketika terjadi kesalahan atau ketidaksesuaian pada salah satu sistem seperti terjadinya perubahan struktur pada data pengguna, maka pengembang harus melakukan perubahan kepada keseluruhan sistem [5].

Web service merupakan teknologi yang dapat digunakan untuk melakukan integrasi sistem dan pengembangan aplikasi dengan basis *multiplatform*. *Web service* dikembangkan sebagai jembatan komunikasi untuk sistem-sistem yang berbeda satu dengan yang lain, sehingga aplikasi memiliki jaringan yang sama dengan standar protokol yang sama yang ditetapkan oleh *web service.* Oleh karena itu dengan menggunakan *web service*, kendala ketergantungan pada *website* konvensional akan dapat teratasi. Bentuk perkembangan *web service* saat ini adalah hadirnya arsitektur *Representational State Transfer* (REST*)* yang dapat diterapkan ke *web service* yang saat ini dikenal dengan nama *RESTful web service*. *RESTful web service* memiliki kinerja yang sangat baik dan optimal. Dalam perkembangannya *RESTful web service* sudah bisa di kembangkan melalui berbagai bahasa pemrograman dan *framework* [1].

Salah satu *framework* yang dapat digunakan untuk mengembangkan *RESTful web service* adalah *framework Spring* dengan bahasa pemrograman *Java*. *Spring* termasuk *framework* yang ringan untuk mendukung secara penuh dalam pengembangan aplikasi *Enterprise* siap pakai. *Spring* memiliki banyak *sub-framework*, salah satunya adalah *Spring Boot. Spring Boot* termasuk *framework* yang cukup ringan untuk digunakan, bersifat *open source*, memiliki cukup banyak modul-modul yang bisa digunakan dalam mengembangkan aplikasi. Aplikasi yang dibuat dengan *Spring Boot* dapat dikombinasikan dengan bahasa pemrograman lain karena *Spring Boot* mendukung pembuatan aplikasi berbasis *RESTful web service* [6].

Berdasarkan permasalahan tersebut, maka rumusan masalah dalam penelitian ini adalah bagaimana menerapkan *RESTful Web Service* dengan *Framework Spring* pada Sistem Pengelolaan Ruang. Oleh karena itu untuk menyelesaikan masalah pengelolaan ruang yang dialami oleh para pengguna ruangan pada instansi dan organisasi tersebut, dibutuhkan sistem yang dapat menyelesaikan masalah tersebut yaitu sistem pengelolaan ruang yang dibuat dengan menerapkan teknologi *RESTful web service*. Tujuan dari penelitian ini adalah menerapkan *RESTful web service* dengan *framework Spring* pada sistem pengelolaan ruang.

2. TINJAUAN PUSTAKA

Penelitian dengan judul "Sistem Informasi Manajemen Ruang (SIMERU) Kelas (Studi Kasus: FKTI Universitas Mulawarman)" membahas tentang sistem informasi manajemen ruang kelas membantu admin bagian akademik dalam proses penjadwalan ruang kuliah dan membantu kegiatan pemesanan ruang kelas pengganti. Pada penelitian terdahulu, sistem informasi manajemen ruang kelas sangat membantu admin dalam proses penjadwalan ruang kuliah dan pemesanan ruang kelas pengganti. Dari keberhasilan penelitian terdahulu, maka dibuat penelitian ini yang nantinya tidak hanya membantu sisi admin, namun juga membantu sisi calon pengguna ruang dalam memperoleh

informasi dan memesan ruang. Penelitian terdahulu dibuat dengan bahasa pemrograman *VB Net* 2010 dan belum menggunakan teknologi *RESTful web service*, sedangkan pada penelitian ini sistem dibuat dengan menerapkan teknologi *RESTful web service* dengan menggunakan pemrograman *Java* dan *framework Spring*. Dalam penelitian terdahulu sistem dibuat untuk admin, sedangkan pada penelitian ini sistem akan dibuat untuk pengguna ruangan [7].

Penelitian dengan judul "Implementasi Teknologi *RESTful Web Service* dalam Pengembangan Sistem Informasi Perekaman Prestasi Mahasiswa Berbasis *Website* (Studi Kasus: Fakultas Teknologi Pertanian Universitas Brawijaya)" membahas tentang sistem informasi untuk merekam prestasi mahasiswa di Fakultas Teknologi Pertanian Universitas Brawijaya yang dibangun berbasis website dengan menerapkan *RESTful web service*. Penelitian ini memiliki kesamaan teknologi dengan penelitian terdahulu yaitu sama-sama menggunakan teknologi *RESTful web service*. Pada penelitian terdahulu pada penerapan aplikasi REST, setiap transaksi bersifat independen dan tidak terkait dengan transaksi lainnya (*stateless*) yang membuat aplikasi REST sederhana dan ringan. Perbedaan penelitian ini dengan penelitian terdahulu yaitu penelitian terdahulu memiliki kasus yang berbeda dengan penelitian ini, yaitu sistem informasi perekaman prestasi mahasiswa, juga penelitian terdahulu dibuat dengan *framework Laravel* dan bahasa pemrograman *PHP*, sedangkan penelitian ini dibuat dengan bahasa pemrograman *Java* dan *framework Spring* [8].

Penelitian dengan judul "Pembangunan Aplikasi *Web Event* Menggunakan *Framework Spring Boot* di PT XYZ" membahas tentang pembangunan aplikasi *Web Event* dengan menggunakan bahasa pemrograman *Java* dan *framework Spring Boot* untuk membantu peserta *event* serta admin untuk mengelola setiap *event* dengan mudah. Penelitian ini memiliki kesamaan dengan penelitian terdahulu yaitu sama-sama menggunakan *framework Spring.* Pada penelitian terdahulu disimpulkan bahwa pembangunan aplikasi menggunakan *framework Spring Boot* sangat mudah karena didukung dengan *library* yang lengkap dan aplikasi yang dibuat sangat ringan saat digunakan. Perbedaan penelitian ini dengan penelitian terdahulu adalah penelitian ini akan dibangun dengan menggunakan teknologi *RESTful web service*, sedangkan penelitian terdahulu tidak menggunakan teknologi *RESTful web service* dalam membangun aplikasinya. Penelitian terdahulu juga mengembangkan sistem yang berbeda dengan sistem pada penelitian ini yaitu aplikasi *web event*, sedangkan penelitian ini akan mengembangkan sistem pengelolaan ruang [6].

Web service adalah kumpulan layanan web dengan menggunakan jaringan protokol HTTP yang dapat diakses dan digunakan oleh pengguna dengan berbagai macam bahasa pemrograman, arsitektur, dan sistem operasi yang berbeda. *Web service* dapat diterapkan menggunakan arsitektur *Simple Object Access Protocol* (SOAP) atau *Representational State Transfer* (REST). Layanan *Web Service* dapat dibuat dalam bentuk format JSON, XML, dan teks. [9]

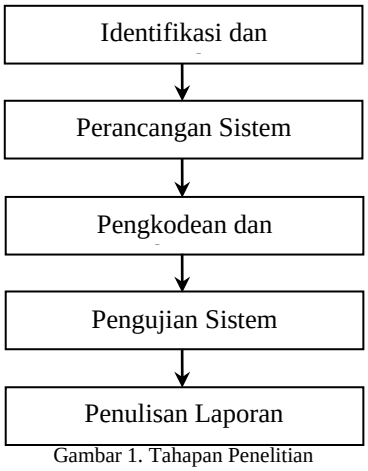
RESTful adalah arsitektur dari *web service* yang bekerja menggunakan *resource* untuk membangun sistem terdistribusi, sehingga arsitektur ini didesain dengan menekankan kesederhanaan, skalabilitas, dan kegunaan. *Web Service* yang berorientasi pada *resource* dalam implementasinya banyak menggunakan REST, dengan menyediakan *resource-resource* sebagai layanan [1].

Framework adalah komponen pemrograman yang *re-use* (bisa digunakan ulang) kapan saja, sehingga *programmer* tidak harus membuat ulang *script* yang sama untuk tugas yang sama. Misalkan programmer ingin halaman-halaman web menampilkan data dengan paginasi (*paging*) halaman, *framework* telah menyediakan fungsi *paging* tersebut sedangkan *programmer* cukup menggunakan fungsi tersebut pada saat *coding*, tetapi tentu dengan kaidah-kaidah yang ditetapkan oleh masing-masing *framework* [10].

Framework Spring adalah *framework open source* berbasis *Java* yang menyediakan infrastruktur yang komprehensif dalam mengembangkan aplikasi *Java* dengan mudah dan cepat. *Spring* dapat digunakan untuk melakukan pengaturan deklarasi manajemen transaksi, *remote access* dengan menggunakan RMI atau layanan web lainnya, fasilitas *mailing*, dan beragam opsi untuk pengaturan data ke *database* [11].

3. METODE PENELITIAN

Sistem pengelolaan ruang adalah sistem yang digunakan untuk mengelola ruang (CRUD) termasuk didalamnya terdapat proses peminjaman. Data yang digunakan pada sistem ini adalah data ruang, fasilitas, lokasi cabang / gedung, pengguna, dan data transaksi peminjaman yang memiliki atribut datanya masing-masing. Tantangan dari penelitian ini adalah bagaimana data dari setiap sumber yang berbeda-beda dapat diintegrasikan dan diakses dari satu sistem / aplikasi. *Web service* adalah solusi dari permasalahan tersebut, dimana *web service* dapat mengintegrasikan data dari sumber yang berbeda-beda menjadi satu aplikasi. Metode penelitian yang digunakan pada penelitian ini didasarkan pada diagram yang ada di gambar 1.



Gambar 1. Tahapan Penelitian

Tahapan pertama pada penelitian ini adalah melakukan identifikasi dan mengumpulkan data yang diperlukan terkait dengan sistem pengelolaan ruang yang akan dibangun. Data diambil dari studi kepustakaan dengan referensi dari jurnal, artikel, dan situs internet yang berkaitan dengan penelitian ini.

Tahap kedua adalah melakukan perancangan sistem menggunakan *Unified Modeling Language* (UML). Pada tahapan ini, UML yang digunakan dalam perancangan adalah *Use Case Diagram* yang menghasilkan aktor *user* dan admin dengan interaksinya masing-masing dan *Entity Relationship Diagram* (ERD) yang menghasilkan lima entitas yaitu ruang, tipe ruang, transaksi, lokasi, dan user.

Tahap ketiga adalah melakukan pengkodean dan implementasi program berdasarkan rancangan sistem yang telah dibuat sebelumnya. Tahap ini dimulai dengan pembuatan basis data dengan menggunakan MySQL berdasarkan ERD yang telah dibuat pada tahap sebelumnya. Pembangunan basis data ini menghasilkan lima tabel dan relasinya sama seperti entitas pada ERD. Pada tahap ini juga dibangun sistem dengan menerapkan *RESTful web service* menggunakan *framework Spring* yang hasil penerapanya adalah *Application Programming Interface* (API).

API merupakan sebuah sistem yang menjembatani komunikasi antara satu aplikasi dengan aplikasi lainnya. Model API yang dibuat berdasarkan *RESTful web service* ini adalah REST API, dimana pada model ini terdapat beberapa metode yang dapat digunakan seperti GET, POST, PUT, dan DELETE yang nantinya akan mengembalikan *response* dalam format JSON [12],[13].

Setelah pembuatan API, pada tahapan ini juga dibuat *front-end* beserta *User Interface (UI)* dengan menggunakan *framework Spring* dan *library Thymeleaf* sebagai pengolah data dari *server* ke HTML dan *framework Bootstrap* sebagai tampilan UI.

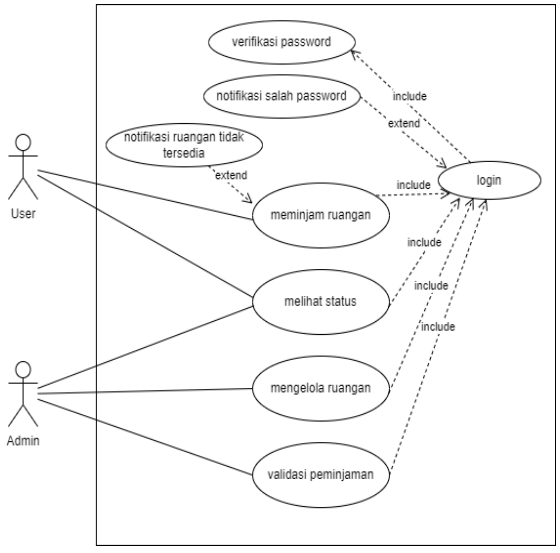
Tahap keempat adalah melakukan pengujian terhadap sistem. Pengujian yang dilakukan pada penelitian ini adalah pengujian dengan metode *black box*. Setiap API *endpoint* beserta metode seperti GET dan POST dari setiap *controller* akan diuji untuk mengetahui apakah *response* sudah sesuai dengan hasil yang diharapkan. Setelah dilakukan pengujian, ketika terjadi kesalahan maka akan dilakukan perbaikan.

Tahap terakhir setelah pengujian selesai adalah menyusun laporan dari hasil penelitian yang telah dilakukan.

4. HASIL DAN PEMBAHASAN

Hasil dari penelitian ini adalah mengimplementasikan teknologi *RESTful web service* menggunakan *framework Spring* pada sistem pengelolaan ruang. Implementasi *web service* yang dilakukan pada penelitian ini adalah *Application Programming Interface (*API). Terdapat tiga tahap utama dalam pembahasan ini, dimulai dari perancangan sistem, pengkodean dan implementasi, dan pengujian sistem.

Tahap perancangan sistem dilakukan dengan membuat *use case diagram* dan *Entity Relational Diagram* (ERD). *Use case diagram* adalah diagram yang menggambarkan fungsionalitas dari sebuah sistem. *Use case* merepresentasikan interaksi antara aktor dengan sistem, seperti melakukan *login* dan lain-lain [13].

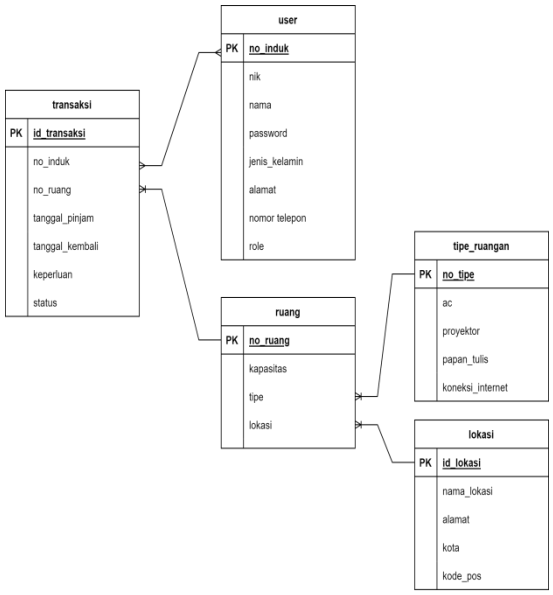


Gambar 2. Use Case Diagram

Gambar 2 adalah *use case* dari sistem pengelolaan ruang. *Use case* ini melibatkan dua aktor yaitu *user* dan admin. *User* pada sistem ini dapat melakukan interaksi meminjam ruang dan melihat status, sedangkan admin dapat melakukan interaksi mengelola ruangan, validasi peminjaman, dan melihat status dari transaksi, ruangan, dan *user*. Pada *use case* ini semua interaksi yang dilakukan oleh aktor harus melalui interaksi *login*

terlebih dahulu.

ERD adalah diagram yang berguna untuk menggambarkan relasi antara tabel / entitas dari sebuah *database* [14]. Gambar 3 merupakan ERD dari Sistem Pengelolaan Ruang yang menghasilkan lima entitas yang masing-masing memiliki relasi satu sama lain.



Gambar 3. Entity Relation Diagram

Tahap awal sebelum melakukan implementasi adalah membuat basis data berdasarkan ERD yang telah dibuat sebelumnya. Basis data dibuat dengan MySQL dan menghasilkan lima tabel, yaitu tabel transaksi, tabel *user*, tabel, ruang, tabel tipe_ruangan, dan tabel lokasi. Tabel ruang akan berelasi dengan tabel lokasi dan tipe ruangan. Tabel transaksi pada basis data ini akan berelasi dengan *user* dan ruangan.

Sebelum melakukan implementasi dan pengkodean hal yang harus dilakukan adalah melakukan konfigurasi *project Spring Boot* yang dilakukan dengan cara menambahkan *dependency* pada *project* sesuai kebutuhan. *Dependency* adalah salah satu fitur dari *Spring Boot* yang digunakan untuk mempermudah penulisan kode dalam pembangunan aplikasi. Selanjutnya adalah melakukan konfigurasi pada *Application Properties* yang berisi *file* konfigurasi yang berisi segala pengaturan yang akan digunakan pada aplikasi yang ditunjukkan pada kode program 1.

Kode Program 1. Konfigurasi *Application Properties*

```
1  spring.datasource.url=jdbc:mysql://l
   ocalhost:3306/db_tugas_akhir?useUnic
   ode=true&useJDBCCompliantTimezoneShi
   ft=true&useLegacyDatetimeCode=false&
   serverTimezone=UTC
2
3  spring.datasource.driver-class-
   name=com.mysql.jdbc.Driver
4
5  spring.datasource.username=root
6  spring.datasource.password=
7
8  spring.jpa.database-
   platform=org.hibernate.dialect.MySQL
   Dialect
9  server.port=8085
```

Application Properties tersebut berisi beberapa konfigurasi yang diperlukan dalam membangun sistem. Konfigurasi di baris pertama sampai keenam berfungsi untuk mengakses database yang mana pada aplikasi ini adalah *database* MySQL. Konfigurasi lain yang ada pada *Application Properties* ini adalah konfigurasi *Java Persistence* API (JPA) pada baris ke delapan yang berfungsi sebagai standarisasi pengelolaan data relasional yang terdiri atas API untuk menyimpan *object* (*entity*) ke dalam *database relational*. Konfigurasi terakhir adalah konfigurasi *port* dari sistem yang akan berjalan.

Pada penelitian ini, pembangunan sistem pengelolaan dengan *RESTful web service* menghasilkan enam *packages*, yaitu *common*, *controller*, *entities*, *repository*, *services*, dan *Data Transfer Object* (DTO). Hasil dari pembangunan *web service* ini adalah *Application Programming Interface* (API), API merupakan sebuah sistem yang menjembatani komunikasi antara satu aplikasi dengan aplikasi lainnya. API pada sistem ini nantinya akan menghasilkan data dari *database* dan *service* ke dalam bentuk JSON [12].

Langkah awal untuk mengimplementasikan *RESTful web service* adalah membuat *entity* pada *package entities*. *Entity* adalah sebuah kelas *Plain Old Java Object* (POJO) yang mewakili data yang disimpan pada *database*.

Kode Program 2. Kelas Entity

```
1  @Entity
2  @Table(name = "ruang")
3  @XmlRootElement
4  @NamedQueries(
   …
   )
10 public class Ruang implements
   Serializable {
```

Kode program 2 adalah kode dari kelas ruang yang merupakan kelas *Entity* pada sistem ini. Pada kelas *Entity*, anotasi @Entity digunakan sebagai pengenal bahwa kelas tersebut adalah kelas *Entity* dan anotasi @Table adalah nama tabel dari *database*. @Table adalah nama tabel dari database yang

diimplementasikan. Setelah membuat *Entity*, selanjutnya adalah membuat *interface repository* yang *meng-extends* dari kelas *interface JpaRepository* yang ditunjukkan di kode program 3.

Kode Program 3. Interface Repository

```
1  @Repository
2  public interface RuangRepository
   extends JpaRepository<Ruang, String>{
3      @Query(
       value = "SELECT * FROM ruang r
   WHERE r.lokasi = ?1", nativeQuery =
   true)
4      List<Ruang>
   findRuangByLokasi(String lokasi);}
```

Repository ini digunakan sebagai penghubung antara kelas *Entity* ke kelas *service*. *JpaRepository* sendiri adalah *repository* dari *Java Persistence* API yang digunakan untuk membuat operasi *Create Read Update* dan *Delete* (CRUD) pada data sehingga pengembang tidak perlu menuliskan *query* untuk mengolah data secara manual. Pada *repository* ini juga dibuat *custom query* dengan cara menambahkan anotasi @Query. Setelah membuat *repository*, yang selanjutnya dilakukan adalah membuat kelas *service* yang berguna sebagai pengolah data *insert*, *read*, *update*, dan *delete* sesuai kebutuhan data. Kelas *Service* disini melakukan pengolahan data *entity* yang telah dijembatani oleh *repository*.

Kode Program 4. Kelas Service

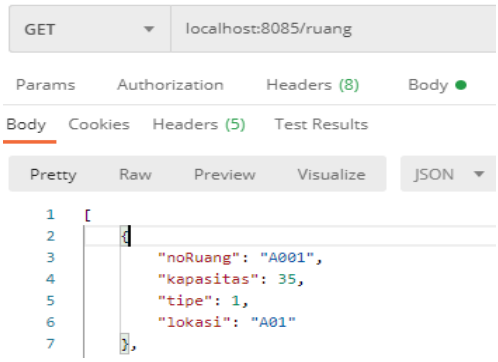
```
1  @Service
2  public class RuangService {
3      @Autowired
4      RuangRepository repository;
5
6       public List<Ruang> getAll(){
7          return repository.findAll();}
```

Kode program 4 adalah kode program dari kelas *service*. Anotasi @Service digunakan sebagai pengenal bahwa kelas tersebut adalah kelas *service*. Fungsi-fungsi yang dibuat pada kelas *service* sendiri dibuat sesuai kebutuhan dari aplikasi. Anotasi @Autowired pada baris tiga dan empat digunakan agar kelas *service* dapat menginjeksi *repository*. Setelah membuat *service* langkah terakhir untuk mengimplementasikan *RESTful web service* agar dapat digunakan adalah pembuatan kelas *controller* yang nantinya akan digunakan untuk mengakses data.

Kode Program 5. Kelas Controller

```
1   @RestController
2   @RequestMapping("ruang")
3   public class RuangController {
4
5       @Autowired
6       RuangService service;
7
8       @GetMapping("")
9       public List<Ruang> getAll(){
10          return service.getAll();}
```

Kode program 5 adalah kode dari kelas *Controller*. Anotasi @RestController digunakan sebagai pengenal kepada sistem bahwa kelas tersebut adalah kelas *Controller* dengan arsitektur REST. @RequestMapping digunakan sebagai *path* tambahan setelah *Universal Resource Identifier* (URI) yang digunakan pada *controller* tersebut. @Autowired pada baris lima dan enam digunakan agar kelas *Controller* dapat melakukan injeksi pada kelas *Service*. Terdapat beberapa anotasi yang digunakan pada kelas *Controller* tergantung *request handling* dari metode yang ingin dibuat seperti POST, GET, PUT, dan DELETE. Salah satu *request* pada kelas ini adalah @GetMapping yang digunakan untuk memberitahu sistem bahwa ketika *user* melakukan *request* dengan metode GET kepada *server*. Ketika masuk ke anotasi @GetMapping, maka *server* akan mengembalikan fungsi yang mengambil *response* dari kelas *service*. Terakhir, *controller* ini akan mengembalikan *response* berupa data dalam bentuk JSON.



Gambar 4. Pengujian API dengan Postman

Gambar 4 adalah hasil pengujian salah satu API dengan Postman. API yang diuji adalah API ruang menggunakan metode GET dan menghasilkan *response* berupa data dalam format JSON. Keuntungan penggunaan *web service* adalah kebebasan pemilihan *platform* yang akan digunakan sebagai *front-end application*. Pada penelitian ini, *front-end* dibangun berbasis web dengan *framework Spring* dan *library Thymeleaf* sebagai pengolah data dari *server* ke dalam HTML, dan *Bootstrap* sebagai *user interface*.

Kode program 6. Service ruang pada front-end

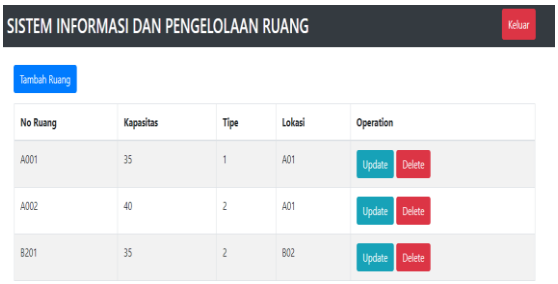
```
1   public List<Ruang> getAll() {
2           List<Ruang> result;
3           ResponseEntity<List<Ruang>>
    response = restTemplate.exchange(
4                   uri+"ruang",
5                   HttpMethod.GET,
6                   null,
7                   new
8   ParameterizedTypeReference<List<Ruang
    >>(){});
9           result = response.getBody();
10          System.out.println(uri);
            return result;
11      }
```

Kode program 6 adalah kode program yang berisi kode untuk memanggil *service* dengan *framework Spring* untuk mendapatkan data ruangan dalam format JSON. *Library Thymeleaf* yang sudah terintegrasi dengan *framework Spring* akan menangkap data dan memunculkan data tersebut pada tabel HTML yang ditunjukkan oleh kode program 7.

Kode program 7. Penggunaan Thymeleaf pada HTML

```
1   <tr th:each="as : ${ruang}">
2   <td th:text="${as.noRuang}"></td>
3   <td th:text="${as.kapasitas}"></td>
4   <td th:text="${as.tipe}"</td>
5   <td th:text="${as.lokasi}"</td>
```

Hasil dari penerapan *Thymeleaf* dan UI dengan *framework Bootstrap* ditunjukkan pada gambar 5. Gambar 5 adalah contoh halaman pengelolaan ruang yang ditampilkan pada sisi admin.



Gambar 5. Tampilan UI pada *front-end application*

Tahap terakhir yang dilakukan setelah pengkodean dan implementasi adalah pengujian sistem. Metode pengujian yang digunakan saat melakukan penelitian ini adalah metode *black box*. Pengujian black box adalah pengujian perangkat lunak dengan menguji fungsionalitas tanpa menguji kode dan desain program untuk mengetahui apakah *input*/*output* dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan [15]. Pengujian dilakukan dengan mengetes setiap API pada *controller*, Adapun hasil dari pengujian tersebut :

Tabel 1. Hasil Pengujian

No	Pengujian		Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
1	Authentication Controller	Method POST (Autentikasi Login dengan memasukkan username dan password)	Proses berhasil dengan response berhasil ketika user sudah terdaftar dan gagal ketika user belum terdaftar	Proses berhasil	Valid
2	Ruang Controller	Method GET (Mengambil semua data ruang)	Proses berhasil dengan response semua data ruang	Proses berhasil	Valid
		Method GET (Mengambil data ruang sesuai id)	Proses berhasil dengan response data ruang sesuai id	Proses berhasil	Valid
		Method GET (Mengambil data ruang berdasarkan lokasi)	Proses berhasil dengan response data ruang berdasarkan lokasi	Proses berhasil	Valid
		Method POST (Melakukan insert data ruang)	Proses berhasil dengan response berhasil ketika data berhasil di input	Proses berhasil	Valid
		Method PUT (Melakukan update data ruang berdasarkan id)	Proses berhasil dengan response berhasil ketika id ditemukan dan data berhasil diupdate	Proses berhasil	Valid
		Method DELETE (Melakukan penghapusan data ruang berdasarkan id)	Proses berhasil dengan response berhasil ketika id data ditemukan dan data berhasil di hapus	Proses berhasil	Valid

3	Lokasi Controller	Method GET (Mengambil semua data lokasi)	Proses berhasil dengan response semua data lokasi	Proses Berhasil	Valid
		Method GET (Mengambil data lokasi berdasarkan id)	Proses berhasil dengan response data lokasi sesuai id	Proses berhasil	Valid
		Method POST (Melakukan insert data lokasi)	Proses berhasil dengan response berhasil ketika data berhasil di input	Proses berhasil	Valid
		Method PUT (Melakukan update data lokasi berdasarkan id)	Proses berhasil ketika id ditemukan dan data berhasil di update	Proses berhasil	Valid
		Method DELETE (Melakukan penghapusan data lokasi berdasarkan id)	Proses berhasil ketika id ditemukan dan data berhasil di hapus	Proses berhasil	Valid
4	User Controller	Method GET (Mengambil semua data user)	Proses berhasil dengan response semua data user	Proses berhasil	Valid
		Method GET (Mengambil data user sesuai id)	Proses berhasil dengan response data user sesuai id	Proses berhasil	Valid
5	Transaksi Controller	Method GET (Mengambil semua data transaksi)	Proses berhasil dengan response data semua data transaksi	Proses berhasil	Valid
		Method GET (Mengambil data transaksi sesuai id)	Proses berhasil dengan response data transaksi sesuai id	Proses berhasil	Valid
		Method POST (Melakukan insert data transaksi)	Proses berhasil dengan response berhasil ketika data berhasil di insert	Proses berhasil	Valid
		Method PUT (Melakukan update data transaksi sesuai id)	Proses berhasil dengan response berhasil ketika id ditemukan dan data berhasil di update	Proses berhasil	Valid

Dari hasil pengujian yang telah dilakukan, didapatkan hasil bahwa setiap test case yang disimulasikan sudah sesuai dengan hasil yang diharapkan.

5. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan beberapa hal. Proses Implementasi *RESTful web service* yang dilakukan dengan *framework Spring* membuat kode menjadi sangat sederhana karena didukung oleh *library* yang ada pada *framework*, seperti yang sudah dipaparkan pada hasil penelitian dan pembahasan untuk mengakses data pada *database* tidak perlu menuliskan semua *query* secara manual. Data pada sistem pengelolaan ruang dapat diakses dari berbagai *platform* karena pengaksesan dapat dilakukan dengan hanya mengakses *link web service* yang telah dibuat.

Sistem pengelolaan ruang dapat membantu *user* dalam melakukan peminjaman ruang. Pada sisi *admin*, *admin* dimudahkan dalam melakukan pengelolaan ruang, mulai dari melihat data ruang dan transaksi, hingga melakukan proses CRUD pada data. Dari pengujian yang telah dilakukan, data yang didapatkan sudah sesuai dengan data yang ada pada *database* dan terintegrasi setiap kali terdapat perubahan data, sehingga diambil kesimpulan bahwa data yang ada adalah valid dan akurat.

Pada penelitian lebih lanjut dapat menggunakan hasil penelitian ini sebagai referensi dan dapat menambahkan fitur keamanan pada sistem ini seperti menggunakan *Spring Security* atau juga bisa menggunakan JSON *Web Token* (JWT).

DAFTAR PUSTAKA

[1] R. Choirudin and A. Adil, "Implementasi Rest Api Web Service dalam Membangun Aplikasi Multiplatform untuk Usaha Jasa," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 18, no. 2, pp. 284–293, May 2019, doi: 10.30812/matrik.v18i2.407.

[2] R. Amrullah, A. Megayanti, and A. Yusta, "Sistem Informasi Manajemen Asset Berbasis Web (Studi Kasus : PT. Krakatau It Cilegon)," *Jurnal Ilmiah Sains dan Teknologi*, vol. 4, no. 2, pp. 109–121, Aug. 2020.

[3] Y. Brianorman and B. C. Octariadi, "Perancangan Sistem Pengelolaan Ruang Berbasis Web di Universitas Muhammadiyah Pontianak," *CYBERNETICS*, vol. 1, no. 02, p. 131, Dec. 2017, doi: 10.29406/cbn.v1i02.752.

[4] D. Listiani and G. Sutjahjo, "Sistem Informasi Pengelolaan Ruang Kuliah Di Uniba Berbasis Web Menggunakan PHPdan Database MySql," *Zona Komputer: Program Studi Sistem Informasi Universitas Batam*, vol. 10, no. 2, Mar. 2021, doi: 10.37776/zk.v10i2.404.

[5] L. M. Alchuluq and F. Nurzaman, "Analisis pada Arsitektur Microservice Untuk Layanan Bisnis Toko Online," *TEKINFO*, vol. 22, no. 2, pp. 61–68, Nov. 2021..

[6] W. C. Umbu Dagha and Y. A. Susetyo, "Web Event, Spring Boot, Java Pembangunan Aplikasi Web Event menggunakan Framework Spring Boot di PT XYZ," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 8, no. 3, pp. 1457–1469, Sep. 2021, doi: 10.35957/jatisi.v8i3.1052.

[7] D. M. Khairina, S. Maharani, and H. R. Hatta, "Sistem Informasi Manajemen Ruang (Simeru) Kelas (Studi Kasus: FKTI Universitas Mulawarman)," *Informatika Mulawarman : Jurnal Ilmiah Ilmu Komputer*, vol. 13, no. 1, p. 30, Feb. 2018, doi: 10.30872/jim.v13i1.1023.

[8] W. Wardhana, I. Arwani, & B. Rahayudi, "Implementasi Teknologi Restful Web Service Dalam Pengembangan Sistem Informasi Perekaman Prestasi Mahasiswa Berbasis Website (Studi Kasus: Fakultas Teknologi Pertanian Universitas Brawijaya)," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 4, no. 2, p. 680-689, May 2020. ISSN 2548-964X

[9] R. Rizal and A. Rahmatulloh, "Restful Web Service Untuk Integrasi Sistem Akademik Dan Perpustakaan Universitas Perjuangan," *JURNAL ILMIAH INFORMATIKA*, vol. 7, no. 01, p. 54, Mar. 2019, doi: 10.33884/jif.v7i01.1004.

[10] A. D. Kasman, *Framework laravel 5 : panduan praktis dan trik jitu*. CV Asfa Solution, 2015

[11] Written By Irene Anindaputri Iswanto, S.Kom., M.Sc.Eng. Lecturer Specialist S2 – Artificial Intellgent, School of Computer Science, "Framework Spring Java," School of Computer Science. https://socs.binus.ac.id/2017/10/04/framework-spring-java/ (accessed Feb. 11, 2022).

[12] D. K. Ladiba, W. A. Dewa, and S. Arifin, "Analisis dan Pengembangan API Siakad Menggunakan Arsitektur Restful Web Service pada Infrastruktur Microservice," *Prosiding SeNTIK*, vol. 5, no. 1, pp. 255–265, Aug. 2021.

[13] M. R. Nashrulloh, R. Setiawan, D. Heryanto, A. Sutedi, and R. Elsen, "Designing Microservices Architecture for Software Product in Startup," *Jurnal Teknik Informatika (JUTIF)*, vol. 3, no. 1, pp. 44–48, Feb. 2022, doi: https://doi.org/10.20884/1.jutif.2022.3.1.124 .

[14] M. H. Maulana, A. A. Raka, C. D. Andini, and F. Nadziroh, "Sistem Peminjaman Ruangan Online (SPRO) dengan Metode UML (Unfield Modeling Language)," *Jurnal Teknologi dan Terapan Bisnis*, vol. 1, no. 1, pp. 1–8, Mar. 2018, doi: 10.0301/jttb.v1i1.35.

[15] M. Tabrani, S. Suhardi, and H. Priyandaru, "Sistem Informasi Manajemen Berbasis Website Pada Unl Studio Dengan Menggunakan Framework Codeigniter," *JURNAL ILMIAH M-PROGRESS*, vol. 11, no. 1, Jan. 2021, doi: 10.35968/m-pu.v11i1.598.

[16] W. N. Cholifah, Y. Yulianingsih, and S. M. Sagita, "Pengujian Black Box Testing pada Aplikasi Action & Strategy Berbasis Android dengan Teknologi Phonegap," *STRING (Satuan Tulisan Riset dan Inovasi Teknologi)*, vol. 3, no. 2, p. 206, Dec. 2018, doi: 10.30998/string.v3i2.3048.