Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id

E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May

OPEN ACCESS

https://doi.org/10.35870/jimik.v5i2.712

Pembuatan *Website* REST API Ensiklopedia Tumbuhan dengan Kombinasi *Framework Spring Boot* dan *MyBatis Generator*

Michael Antonius Hartono [1], Evangs Mailoa [2*]

[1,2*] Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Kota Salatiga, Provinsi Jawa Tengah, Indonesia.

Email: 672020015@student.uksw.edu [1], evangs.mailoa@uksw.edu [2*]

Histori Artikel:
Dikirim 7 Maret 2024; *Diterima dalam bentuk revisi* 4 April 2024; *Diterima* 14 April 2024; *Diterbitkan* 10 Mei 2024.

Abstrak

Penelitian ini berfokus pada pengembangan REST API Ensiklopedia Tumbuhan, sebuah aplikasi website yang dirancang menggunakan teknologi Java, HTML, Spring Boot, MyBatis Generator, dan Swagger, dengan PostgreSQL sebagai database yang digunakan untuk tempat penyimpanan data tumbuhan yang didapat dari API. Penelitian ini menyoroti berkurangnya kemampuan pada generasi muda, terutama dalam aspek matematika, literasi, dan sains berdasarkan data statistik dari skor tes PISA dari tahun 2003 hingga 2022 yang dilakukan pada anak berumur 15 tahun di berbagai dunia yang menunjukkan puncak penurunan pada tahun 2022. Metode penelitian yang diterapkan mencakup analisis kebutuhan dan identifikasi masalah, perancangan sistem, perancangan aplikasi, implementasi, serta pengujian sistem. Penelitian ini bertujuan mengatasi kurangnya pengetahuan tentang tumbuhan yang dapat berdampak negatif pada ekosistem serta memberikan berbagai manfaat lainnya. Hasil dari penelitian ini adalah aplikasi website ensiklopedia tumbuhan yang dapat diakses oleh berbagai macam user untuk mencari informasi terkait tumbuhan dan website swagger untuk admin yang berfungsi untuk manajemen data tumbuhan.

Kata Kunci: Situs Web; Java; HTML; Kerangka Kerja Spring Boot; MyBatis.

Abstract

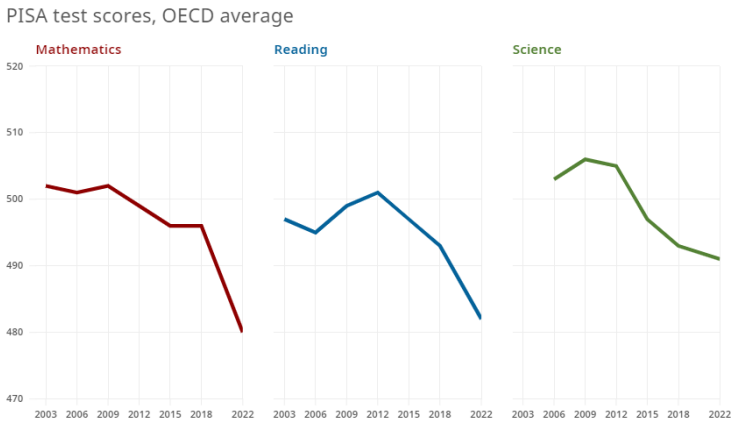
This research focuses on the development of the Plant Encyclopedia REST API, a website application designed using Java, HTML, Spring Boot, MyBatis Generator, and Swagger technologies, with PostgreSQL as the database used to store plant data from the API. This study highlights the declining abilities of the younger generation, especially in mathematics, literacy, and science. Statistical data from PISA test scores from 2003 to 2022 on 15-year-olds worldwide show a peak decline in 2022. The applied research methods include needs analysis, problem identification, system design, application design, implementation, and system testing. This research aims to overcome the lack of knowledge about plants that can have a negative impact on the ecosystem and provide various other benefits. The result of this research is a plant encyclopedia website application that can be accessed by various users to find information related to plants and a swagger website for administrators to manage plant data.

Keyword: Website; Java; HTML; Framework Spring Boot; MyBatis.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

1. Pendahuluan

Dalam era digital yang terus berkembang, aplikasi *web* dan layanan *web* REST API telah menjadi unsur penting dalam pengembangan perangkat lunak, yang mempermudah komunikasi dan pertukaran data yang efisien serta terstruktur antara berbagai platform. *Web* adalah ruang informasi global yang sumber dayanya dapat diakses melalui pertukaran data berdasarkan protokol HTTP (*Hypertext Transfer Protocol*) antara klien dan server (Pranata et al., 2018). Kemudahan layanan *web* tersebut meningkatkan kemampuan kita sebagai manusia untuk dapat mengakses informasi secara cepat melalui internet. Tumbuhan secara umum merupakan makhluk hidup yang berstruktur sel, multiseluler, berasal dari embrio, berakar dalam tanah, dan mampu mengubah karbon dioksida menjadi senyawa organik kompleks melalui proses fotosintesis (Taiz & Zeiger, 2014). Tumbuhan juga memegang peran yang krusial dalam memasok oksigen yang kita perlukan untuk bernafas, sekaligus sebagai sumber bahan makanan, energi, obat-obatan, bahan kosmetik, dan berbagai manfaat lainnya (Liantoni, 2015). Banyaknya kasus pembakaran hutan dan penebangan hutan secara liar yang merusak ekosistem disebabkan oleh kurangnya pengetahuan akan pentingnya peran lingkungan dalam kehidupan manusia.



Gambar 1. Data PISA test scores, OECD average
(PISA 2022 Results (Volume I), 2023)

Country	Math Score		Science Score		Reading Score		Overall S
Indonesia	Min	Max	Min	Max	Min	Max	Min
Indonesia	366		383		359		NaN
World	439		448		436		1528

Gambar 2. Data perbandingan rata-rata Indonesia dan rata-rata dunia
(PISA Scores by Country 2024, n.d.)

Gambar 1 menunjukkan penurunan skor anak 15 tahun dalam matematika, literasi, dan sains, dipengaruhi oleh kemajuan teknologi yang membuat banyak anak kurang bijak menggunakan teknologi (hingga 30% dari *website* OECD). Dampaknya termasuk kemalasan mencari pengetahuan, kurang literasi, dan waktu belajar yang terbatas. Faktor-faktor ini menjadi alasan utama penurunan prestasi di ketiga bidang tersebut. Sedangkan pada Gambar 2 menunjukkan perbandingan rata-rata skor PISA antara anak Indonesia dengan rata-rata seluruh negara, menunjukkan pengetahuan anak Indonesia cenderung lebih rendah. Penggunaan REST API dipilih karena kemudahan manajemen dan penyimpanan data, serta fleksibilitas untuk berintegrasi dengan aplikasi lainnya. Pemilihan *Framework Spring Boot* diputuskan karena kemudahan pengembangan aplikasi *website*, bobot yang ringan, lisensi *open source*, dan ketersediaan modul yang mendukung seluruh proses pengembangan aplikasi. Kelebihan tambahan dari *Spring Boot* adalah kemampuannya untuk berintegrasi dengan berbagai bahasa pemrograman lainnya, sehingga memungkinkan pembuatan aplikasi berbasis layanan *website* REST API (Dagha & Susetyo, 2021). Selain itu, Metode ORM dipilih karena sistem keamanan

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E - ISSN: 2723-7079 | P - ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

yang lebih baik dibandingkan dengan *query native* pada umumnya (Putra & Susetyo, 2020). Penggunaan *Object-Relational Mapping* (ORM) dalam penelitian ini menggunakan *MyBatis*, didukung oleh *MyBatis Generator* (MBG) sebagai alat bantu. MBG berperan dalam memeriksa struktur tabel dalam basis data dan menghasilkan komponen-komponen yang mempermudah akses ke tabel tersebut. Fokus utama MBG adalah meningkatkan efisiensi operasi dasar pada *database* seperti CRUD (*Create*, *Read*, *Update*, dan *Delete*) (MyBatis, 2023).

REST (*REpresentational State Transfer*) merupakan *framework* arsitektur komunikasi yang biasa digunakan dalam pengembangan *website* dan aplikasi berbasis layanan, sedangkan API (*Application Programming Interface*) merupakan *gateway* yang mendukung interaksi dan pertukaran data antar aplikasi (Hasanuddin et al., 2022). *Object Relational Mapping* (ORM) merupakan teknik otomasi dan transparansi dari object persistence ke dalam tabel pada basis data, menggunakan metadata yang mendeskripsikan pemetaan antara objek dan basis data (Riyanto et al., 2018);(Rahmat et al., 2023). Saat merancang aplikasi berbasis *web*, salah satu aspek yang krusial adalah efisien dan konsisten dalam mengelola data dari basis data. Prinsip dari ORM yaitu pemetaan entitas data ke dalam kelas atau sebaliknya (Resa et al., 2018). Dalam upaya untuk membantu memetakan koneksi antara tabel-tabel dalam basis data relasional dengan kelas dalam bahasa pemrograman berorientasi objek, penggunaan *web service* REST API (Chatterjee & Mamatha, 2020) dan teknologi *Object-Relational Mapping* (ORM) seperti *MyBatis Generator* menjadi hal umum untuk pengembangan situs *web* maupun aplikasi.

MyBatis Framework dalam pengembangan aplikasi berbasis *Java*, seringkali dikombinasikan dengan *framework* lain seperti; *Spring Framework* dan *Struts Framework* yang menjadi lapisan terluar dan menjadi penghubung dengan *Database Bridging Layer* dengan pengguna sistem yang dibangun menggunakan pemrograman *Java* (Ginanjar et al., 2021). *MyBatis Generator* (MBG) sendiri adalah *tools* yang berfungsi sebagai *generator* kode yang sepenuhnya terintegrasi dengan semua varian *MyBatis*. Penelitian ini dilakukan karena kepentingan informasi mengenai tumbuhan yang sangat vital bagi masyarakat secara umum. Informasi ini memiliki potensi untuk mendukung berbagai sektor, termasuk pendidikan, penelitian, pertanian, manajemen lingkungan, dan perkembangan teknologi. Selain itu, pengetahuan tentang tumbuhan dapat meningkatkan kesadaran akan pentingnya lingkungan untuk menjaga ekosistem alam, sekaligus memberikan manfaat dalam aspek kebutuhan pangan, ekonomi, dan kesehatan.

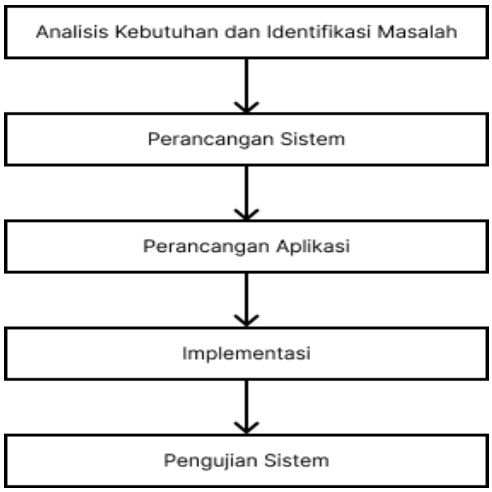
Hasil implementasi berbentuk REST API mengenai data tumbuhan menggunakan swagger dengan fungsi CRUD untuk *admin* dan ensiklopedia tumbuhan yang berbentuk halaman *website* karena kemudahan untuk akses oleh *user* di berbagai *device* seperti melalui komputer, *tablet*, maupun *smartphone* dan secara langsung dapat dijelajah secara online diharapkan untuk dapat membantu terkait masalah penurunan pengetahuan anak zaman sekarang terkhususnya pada bagian sains yang berhubungan dengan tumbuhan. Tujuan dari penelitian ini adalah menjelajahi pengimplementasian REST API untuk dapat mengelola informasi seputar tumbuhan dengan menggunakan *Spring Boot* dan *MyBatis Generator* yang mempermudah sinkronisasi terhadap *database* dan tabelnya dengan cara membaca isi *database*.

Dengan adanya penggunaan REST API, memungkinkan untuk melakukan pemisahan tugas pada bagian *frontend* dan *backend* yang memiliki banyak keuntungan berupa penghematan waktu pada proses pembuatan bagian sesuai keahlian, fleksibilitas teknologi berupa bahasa pemrograman yang berbeda, meningkatkan keamanan karena bagian *backend* yang terpisah dari bagian *frontend*, serta bermacam keuntungan lainnya. Selain itu, dengan mengintegrasikan struktur *Spring Framework* dan *MyBatis Framework*, diharapkan sistem yang dibangun mampu mengatasi kelemahan yang dimiliki oleh masing-masing *framework* (Ilman & Ginanjar, 2019).

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

2. Metode Penelitian

Dalam penelitian perancangan dan implementasi *website* REST API Ensiklopedia Tumbuhan dengan kombinasi *Framework Spring Boot* dan *MyBatis Generator* dilakukan melalui lima tahapan penelitian, yaitu: 1) Analisis kebutuhan dan identifikasi masalah. 2) Perancangan sistem. 3) Perancangan aplikasi. 4) Implementasi. 5) Pengujian sistem.

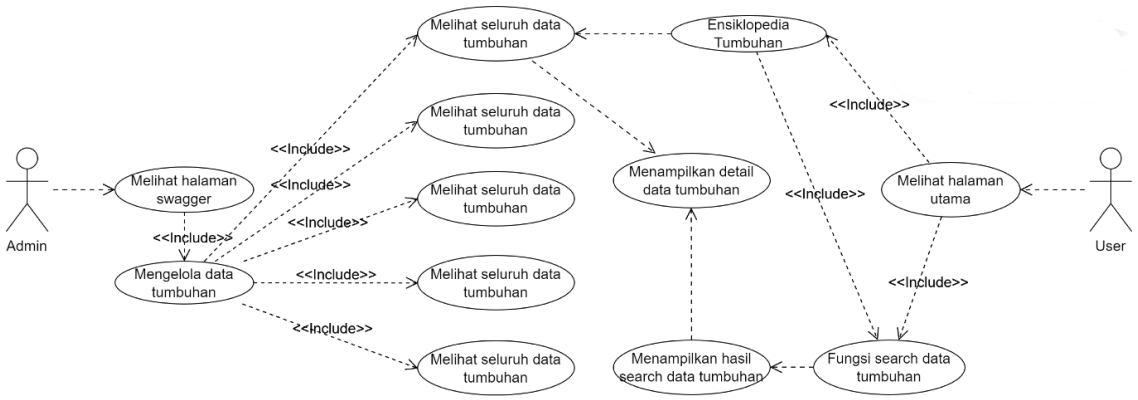


Gambar 1. Tahap Penelitian

Pada tahapan pertama yaitu analisis kebutuhan dan identifikasi masalah berfokus pada mengumpulkan data dan informasi mengenai bermacam tumbuhan yang dibutuhkan untuk ditampilkan oleh *Backend*, lalu dianalisa untuk digunakan dalam pembuatan aplikasi *website* REST API Ensiklopedia Tumbuhan. Banyaknya orang yang kurang berpengetahuan mengenai tumbuhan serta banyaknya kasus pembakaran hutan dan penebangan hutan secara liar menjadi alasan dilakukannya perancangan dan implementasi *website* REST API yang berguna untuk memberikan informasi mengenai tumbuhan dan pentingnya tumbuhan bagi kehidupan manusia.
Pada tahapan kedua dilakukan perancangan sistem yang berfokus untuk menggambarkan bagaimana proses *user* dalam berinteraksi dengan *website* ini untuk dapat mengakses informasi mengenai tumbuhan.

Setelah itu, pada tahapan ketiga yaitu tahap perancangan aplikasi meliputi pembuatan rancangan *database* dan rancangan *User Interface* pada *website*. Pada tahapan ke-empat, yaitu tahap implementasi berfokus pada pembuatan sistem API yang telah dirancang menggunakan *database* PostgreSQL dan bahasa pemrograman *Java* dengan *Framework Spring Boot* dan *ORM MyBatis*. Setelah menyelesaikan implementasi API dan *database*, maka dilanjutkan ke tahap pembuatan *Frontend* berupa *User Interface* sebagai tampilan dari *Website* Ensiklopedia Tumbuhan, lalu dilanjutkan dengan menggunakan API ke dalam *website* sehingga dapat memanggil dan menampilkan data yang diinginkan dari *database*. Setelah melalui proses implementasi, maka dilanjutkan ke tahapan pengujian sistem yang dilakukan dengan menguji coba REST API yang telah dibuat dengan menggunakan bantuan *Swagger* dan menguji coba *Website* Ensiklopedia Tumbuhan. Uji coba ini dilakukan untuk memastikan bahwa API berjalan dengan lancar dan dapat mengelola data sesuai yang diinginkan, serta memastikan bahwa *Frontend* dapat mengambil dan menampilkan data pada tampilan *website* dengan tepat.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712



Gambar 2. *Use Case Diagram* Sistem

Gambar 4 merupakan *use case* diagram dari sistem ensiklopedia tumbuhan Terra Floralis. Terdapat dua aktor yang melakukan aktivitas pada *website* enskilopedia tumbuhan Terra Floralis, aktor yang berada pada sisi kiri adalah user. Aktor *admin* memiliki akses untuk melihat halaman swagger yang berfungsi untuk mengelola data tumbuhan berupa *create*, *read*, *update*, dan *delete* (CRUD). Sedangkan aktor yang berada pada sisi kanan, yaitu *user* memiliki akses untuk menjelajahi halaman utama *website*, melihat *list* data tumbuhan yang ada, dapat menggunakan fungsi *search* untuk mempermudah pencarian tumbuhan secara spesifik, dan memiliki akses untuk melihat detail informasi dari suatu tumbuhan.

3. Hasil dan Pembahasan

Pada bagian ini menjelaskan hasil dari perancangan dan implementasi *website* REST API Ensiklopedia Tumbuhan dengan kombinasi *Framework Spring Boot* dan *MyBatis Generator* untuk mempermudah jalurnya informasi seputar tumbuhan yang akan memudahkan pengguna untuk dapat memanfaatkan informasi tersebut untuk berbagai hal. Pada *website* ini seperti dengan judulnya menggunakan *Framework Spring Boot* dan *MyBatis Generator*. *Framework Spring Boot* digunakan untuk mengelola *Java Project* yang berfungsi memudahkan pengelolaan dependensi dan mengurangi kesulitan dalam konfigurasi. Sedangkan *MyBatis Generator* untuk melakuan *generate code* yang berisi model dan *mapper database* sehingga memudahkan konfigurasi yang diperlukan untuk mengakses fungsi CRUD pada *database*. *Website* dirancang dengan dua macam tingkat pengguna yaitu *user* dan *admin* yang mengelola data pada *website* dengan menggunakan *swagger*.

3.1 Konfigurasi *Application Properties* dan pengunaan *Spring Boot* serta *MyBatis Generator*

Tahap awal dari pembuatan *project Spring Boot* adalah melakukan penambahan *dependency* sesuai dengan kebutuhan *project*. *Dependency* merupakan salah satu fitur pada *Spring Boot* yang berguna mempermudah penulisan *coding* dalam pembanguan aplikasi *Spring Boot* (Dagha & Susetyo, 2021). Tahap selanjutnya adalah konfigurasi pada *Application Properties*. *Application Properties* berisi mengenai file konfigurasi yang akan digunakan aplikasi. Berikut adalah konfigurasi *Application Properties* yang dilakukan:

Kode Program 1. Konfigurasi *Application Properties*

```
mybatis.type-aliases-package=com.example.tumbuhan.model
server.compression.enabled=true
server.compression.mime-
types=application/json,application/xml,text/html,text/xml,text/plain
server.port=8287
server.servlet.contextPath=/ensiklopedia-tumbuhan
spring.application.name=ensiklopedia-tumbuhan
```

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

```
spring.datasource.driver-class-name=org.postgresql.Driver
spring.datasource.hikari.auto-commit=true
spring.datasource.hikari.connection-timeout=10000
spring.datasource.hikari.idle-timeout=120000
spring.datasource.hikari.maximum-pool-size=10
spring.datasource.password=root
spring.datasource.url=jdbc:postgresql://localhost:5432/testingdb
spring.datasource.username=postgres
```

Kode Program 1 merupakan kode yang digunakan untuk konfigurasi project, terlihat baris pertama adalah *MyBatis*, baris kedua hingga kelima untuk menentukan konfigurasi API dan *path* untuk mencapai API tersebut. Baris ke-enam mengenai *Spring* untuk menentukan nama aplikasi. Lalu, yang terakhir dari baris ketujuh hingga selesai untuk menentukan *database* yang digunakan, aturan *database*, url dari *database* yang dituju, dan akses *username* serta password untuk dapat mengakses *database* tersebut.

Kode Program 2. *Spring Boot Application* Tumbuhan

```
@SpringBootApplication
@MapperScan("com.example.tumbuhan.postgresql.mapper")
public class MybatisApplication {

	public static void main(String[] args) {
		SpringApplication.run(MybatisApplication.class, args);
	}

	@Bean
	public RestTemplate restTemplate(RestTemplateBuilder builder) {
		builder.setConnectTimeout(10000);
		builder.setReadTimeout(5000);
		return builder.build();
	}

}
```

Kode Program 2 merupakan *main class* pada bahasa pemrograman *Java* denan menggunakan *Spring Boot*. Berfungsi untuk menjalankan aplikasi *Spring Boot* melalui anotasi *@SpringBootApplication*. Sedangkan anotasi *@MapperScan* untuk melakukan pemindaian package yang isinya berupa *interface mapper* hasil *generator*.

Kode Program 3. Konfigurasi *MyBatis Generator*

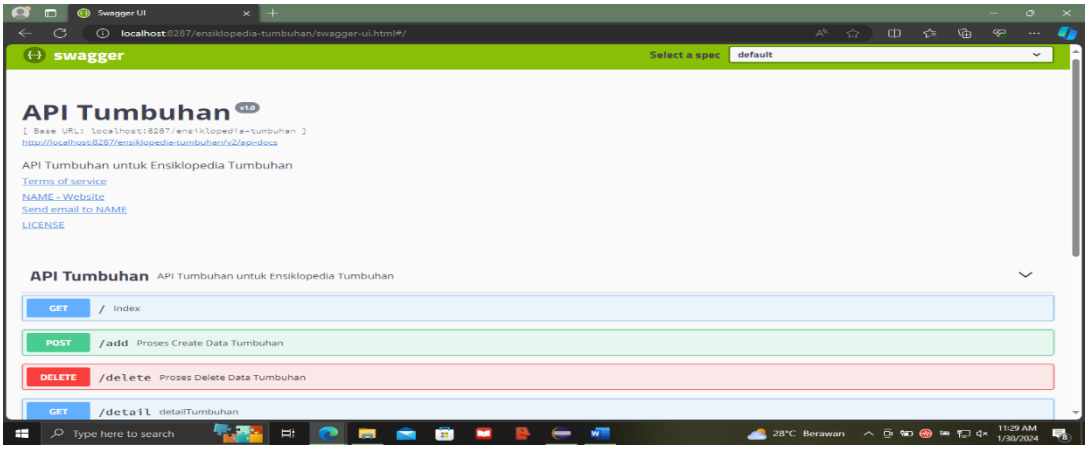
```
<generatorConfiguration>

	<classPathEntry
location="C:\Users\Michael\.m2\repository\org\postgresql\postgresql\9.4.1212\postgresql-
9.4.1212.jar"/>
    <context id="default" targetRuntime="MyBatis3">

	 <jdbcConnection	connectionURL="jdbc:postgresql://localhost:5432/testingdb"
driverClass="org.postgresql.Driver" password="******" userId="******" />
    <javaModelGenerator		targetPackage="com.example.tumbuhan.postgresql.model"
targetProject="api-tumbuhan/src/main/java" >
	<property name="enableSubPackages" value="false"/>
	 <property name="trimStrings" value="true"/>
    </javaModelGenerator>
    <sqlMapGenerator		targetPackage="com.example.tumbuhan.postgresql.mapper"
targetProject="api-tumbuhan/src/main/resources" >
	<property name="enableSubPackages" value="false"/>
    </sqlMapGenerator>
    <javaClientGenerator		targetPackage="com.example.tumbuhan.postgresql.mapper"
targetProject="api-tumbuhan/src/main/java" type="XMLMAPPER" >
	<property name="enableSubPackages" value="false"/>
    </javaClientGenerator>
```

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

```
    <table schema="public" tableName="tumbuhan" enableCountByExample="true"
enableSelectByExample="true">
    <generatedKey column="id" sqlStatement="JDBC" identity="true" />
    </table>
    </context>
</generatorConfiguration>
```

Kode Program 3 digunakan untuk *generate* model, model *example*, *mapper Java*, dan *mapper* xml sesuai dengan *database* pada *project* yang dituju. *Project* yang dituju dan *database* yang dipilih akan mengikuti sesuai dengan apa yang diperintahkan pada file *generator* xml tersebut.

3.2 Pembuatan API Tumbuhan



Gambar 3. Tampilan Halaman *Swagger* API Tumbuhan

API Tumbuhan adalah *software* untuk melakukan *Create*, *Read*, *Update*, dan *Delete* data lalu melanjutkannya agar akses dari data tersebut dapat digunakan untuk berbagai macam. Halaman *swagger* dari API Tumbuhan ini memiliki akses untuk melakukan proses berupa *create*, *read*, *update*, dan *delete*. Proses *read* tersebut dibagi menjadi dua bagian, yaitu *read* semua data dan *read* data berdasarkan id. Selain itu, terlihat yang lainnya berupa *index*, *encyclopedia*, dan detail merupakan akses poin untuk menampilkan halaman *website*.

Kode Program 4. *Create* Data Tumbuhan

```
public String saveTumbuhan(DataTumbuhan dataTumbuhan) throws ServiceException{
                Tumbuhan tumbuhan = new Tumbuhan();
            Tumbuhan tumbuhanCheck =
tumbuhanMapper.selectByPrimaryKey(dataTumbuhan.getId());
            if(!ValidatorUtil.isNullOrEmpty(tumbuhanCheck)) {
                        throw new ServiceException("ID Tumbuhan Sudah ada!");
                }
                tumbuhan.setId(dataTumbuhan.getId());
                tumbuhan.setDeskripsi(dataTumbuhan.getDeskripsi());
                tumbuhan.setJenis(dataTumbuhan.getJenis());
                tumbuhan.setLatin(dataTumbuhan.getLatin());
                tumbuhan.setManfaat(dataTumbuhan.getManfaat());
                tumbuhan.setNama(dataTumbuhan.getNama());
                tumbuhan.setImage(dataTumbuhan.getImageUrl());
                tumbuhanMapper.insert(tumbuhan);
                return "Create Berhasil";
        }
```

Kode Program 4 tersebut untuk dapat melakukan inputan data tumbuhan ke *database*. Proses dilakuan dengan melakukan *setter* pada model tumbuhan hasil dari *generator* dan mengambil data yang diinputkan ke model DataTumbuhan. Setelah itu, melakukan inputan dengan memanggil *mapper*

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

tumbuhan dan melakukan fungsi *insert*. Selain itu, *update* data tumbuhan juga mirip dengan fungsi *create*. Perbedaannya terlihat pada paramnya memerlukan adanya id, proses yang dilakukan dimulai dengan melakukan fungsi *read* untuk membaca seluruh data dari id tersebut. Setelah itu, mengatur nilainya dengan *setter* dan menjalankan fungsi *updateByPrimaryKey*.

Kode Program 5. *Read* semua Data Tumbuhan

```
public List<Tumbuhan> findAll() throws ServiceException {
        List<Tumbuhan> listTumbuhan = new ArrayList<Tumbuhan>();

                listTumbuhan = tumbuhanMapper.findAll();

                if (ValidatorUtil.isNullOrEmpty(listTumbuhan) ) {

                        throw new ServiceException("List Tumbuhan Tidak Ditemukan");
                    }
                return listTumbuhan;
        }
```

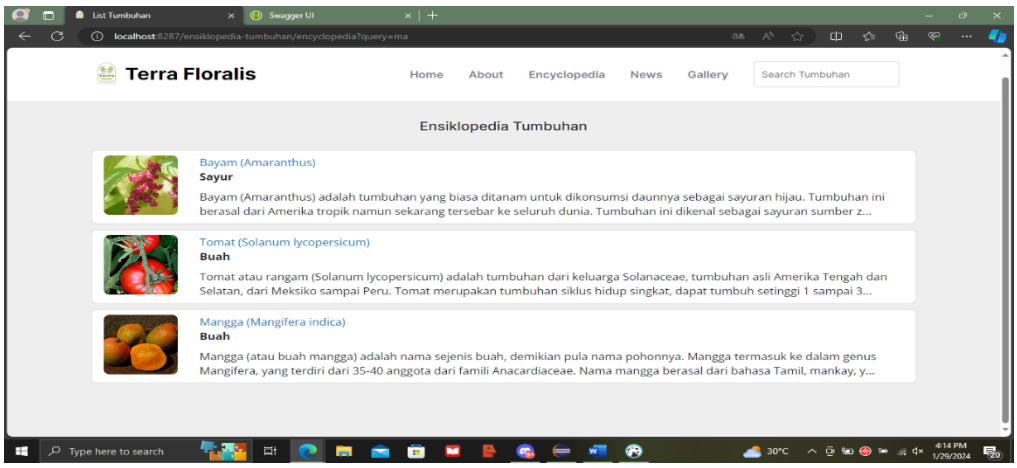
Kode Program 5 berfungsi untuk mendapatkan semua data tumbuhan yang ada di *database*. Proses dilakukan dengan memberikan nilai pada listTumbuhan berupa fungsi *findAll* pada mapper tumbuhan. Setelah itu akan ada validasi jika listTumbuhan tidak ada data atau berupa *null* maka akan menghasilkan *error exception* berupa "*List* Tumbuhan Tidak Ditemukan", tetapi jika datanya tidak *null* atau ada maka akan mengembalikan nilai listTumbuhan tersebut. Selain itu, kode program untuk *read* data berdasarkan id hampir sama. Perbedaannya terlihat pada memerlukan adanya param id dan menggunakan model Tumbuhan bukan sebagai *list*, lalu diberikan nilai berupa fungsi *selectByPrimaryKey*(id). Kode program untuk fungsi *delete* juga sama cara kerjanya dengan melakukan fungsi *read* berdasarkan id, perbedaannya hanyalah pemanggilan fungsinya saja.

3.3 Pembuatan *Website* Terra Floralis dan menyambungkan ke API Tumbuhan

Gambar 4. Tampilan Halaman Home

Terra Floralis adalah *website* yang memanfaatkan data dari API Tumbuhan untuk dijadikan sebagai Ensiklopedia Tumbuhan. Pada halaman *website* ini terdapat akses untuk *user* yang tidak perlu mendaftarkan akun dan dapat secara langsung mengakses halaman *website*. Halaman *home* berisi informasi mengenai *website*, penjelasan mengenai ensiklopedia tumbuhan serta link menuju halaman ensiklopedia, gambaran berita mengenai tumbuhan, dan galeri tumbuhan.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712



Gambar 5. Tampilan Halaman *Encyclopedia* (Hasil *Searching*)

Ketika *user* memasuki halaman ensiklopedia, *user* dapat melihat berbagai macam tumbuhan yang ada dari *list* tersebut maupun menggunakan fungsi *searching*. Fungsi *searching* didasarkan dengan menggunakan nama tumbuhan dan nama latinnya. Setelah *user* melakukan *searching* akan menampilkan hasil tumbuhan yang memiliki atau mendekati dari hasil yang dimasukkan pada inputan *search* tersebut. Contohnya terlihat pada url terdapat *query* yang dimasukkan yaitu "ma" yang menampilkan hasil yang memiliki kata kunci "ma" tersebut.

Kode Program 6. Mengirim *list* data dan data hasil *searching* ke HTML

```
@GetMapping("/encyclopedia")
private String searchTumbuhan(
                @RequestParam(name = "page", defaultValue = "1")int page,
                @RequestParam(name = "size", defaultValue = "5") int size,
                @RequestParam(name = "query", required = false) String query, Model
model) {
        List<Tumbuhan> tumbuhanList;
        int totalTumbuhan = tumbuhanMapper.findAll().size();
        int totalPages = (int) Math.ceil((double) totalTumbuhan / size);
    page = validatePage(page, totalPages);
    int startIndex = (page - 1) * size;
    int endIndex = Math.min(startIndex + size, totalTumbuhan);
    if (query != null && !query.isEmpty()) {
            tumbuhanList = tumbuhanMapper.search(query);
        } else {
            tumbuhanList = tumbuhanMapper.findAllByPage(startIndex, endIndex);
        }
        model.addAttribute("tumbuhan", tumbuhanList);
        model.addAttribute("currentPage", page);
        model.addAttribute("totalPages", totalPages);
        return "encyclopedia";
}
```

Kode Program 6 digunakan sebagai *controller* untuk menentukan *path* menuju tampilan *website enyclopedia* dengan mengirimkan *attribute* ke file HTML *encyclopedia* yang berfungsi untuk menampilkan data tumbuhan yang berbentuk *pagination* dan hasil dari fungsi *search*.

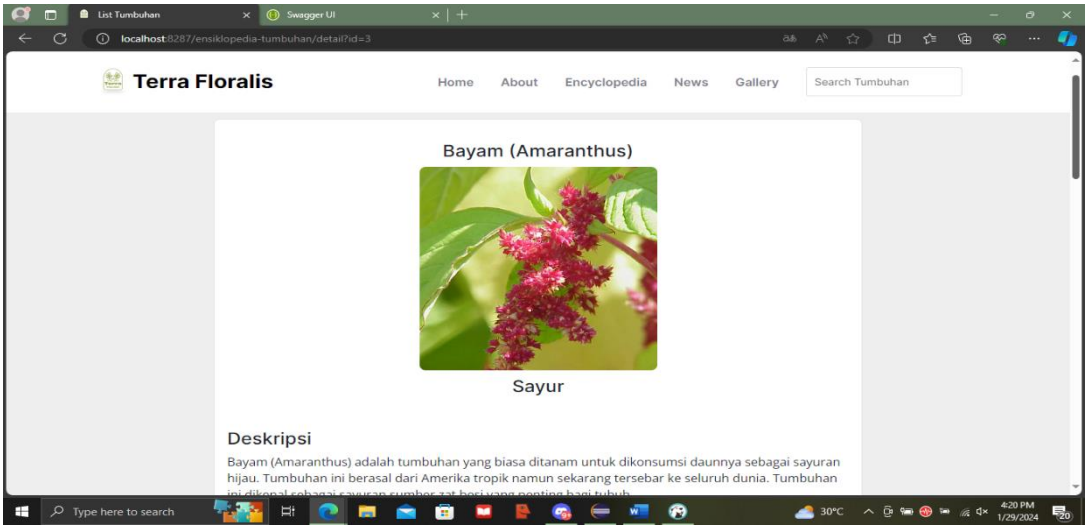
Kode Program 7. Menampilkan *list* data dan data hasil *searching* pada HTML

```
<ul class="list-group" th:each="tumbuhan : ${tumbuhan}">
        <li class="list-group-item">
            <a              class="glightbox"              data-gallery="images-gallery"
th:href="${tumbuhan.image}"><img th:src="${tumbuhan.image}" th:alt="'Gambar Tumbuhan ' +
${tumbuhan.id}" class="img-fluid"></a>
            <div class="content">
```

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

```
                <a class="nama" th:text="${tumbuhan.nama} + ' '+ '(' + ${tumbuhan.latin} +
')'" th:href="'/ensiklopedia-tumbuhan/detail?id=' + ${tumbuhan.id}"></a>
                <span th:text="${tumbuhan.jenis}"></span>
                <ul class="sublist">
                   <li>
                      <p th:inline="text">
                         [[${tumbuhan.deskripsi.length()          >          230          ?
tumbuhan.deskripsi.substring(0, 230) + '...' : tumbuhan.deskripsi}]]
                      </p>
                   </li>
                </ul>
```

Kode Program 7 digunakan untuk menampilkan *list* data maupun data hasil *searching* dengan menggunakan Thymeleaf melalui metode *add attribute* dari Kode Program 6. Setelah itu, diberikan ketentuan jika deskripsi melebihi 230 karakter maka hanya akan ditampilkan 230 karakter tersebut ditambah dengan titik tiga kali. *List* data yang ditampilkan tersebut dipisah melalui *pagination* yang berada di bagian bawah *website*.



Gambar 6. Tampilan Halaman *Encyclopedia*

Pada *list* yang ditampilkan pada halaman *encyclopedia* tumbuhan maupun *list* hasil *searching* jika ditekan nama dari tumbuhan yang ada akan mengalihkan ke bagian detailnya. Untuk bagian detail berisi nama tumbuhan, gambar tumbuhan, jenis tumbuhan, deskripsi tumbuhan, dan manfaat tumbuhan seperti terlihat pada Gambar 8. Kode program untuk mengirimkan data tumbuhan ke HTML dapat dilakukan seperti Kode Program 6 dengan tambahan menggunakan param id dan model. Model tumbuhan tidak berbentuk *list* dan dilakukan fungsi *select* berdasarkan *primary key* adalah id dan diberikan fungsi *if else* jika tumbuhan bukan *null* maka akan melakukan *addAttribute*. Sedangkan kode program untuk menampilkan data pada HTML detail hampir sama dengan Kode Program 7, hanya disesuaikan bagian tampilan agar menyerupai bentuk detail pada Gambar 8 dan menampilkan data tambahan lainnya.

3.4 Pengujian Sistem *Blackbox*

Setelah aplikasi *website* selesai dibuat, tahap selanjutnya yang akan dilakukan adalah pengujian sistem dengan metode *blackbox*. Pengujian sistem dilakukan untuk menguji fungsi dari aplikasi yang telah dibuat dan mencari kesalahan atau *bug* pada sistem. Pengujian tersebut bertujuan untuk memastikan bahwa sistem berjalan sesuai dengan yang diharapkan dan dapat memenuhi kebutuhan pengguna. Tabel 1 dan Tabel 2 adalah hasil pengujian sistem dari aplikasi yang telah dibuat.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E-ISSN: 2723-7079 | P-ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

Tabel 1. Hasil Pengujian Sistem *Blackbox User*

Fungsi yang diuji	Kondisi	Output yang diharapkan	Output yang dihasilkan sistem	Status Pengujian
Search	Mengisi *form search* dan ada kata kunci yang terkait	Berhasil menampilkan data hasil searching	Sukses menampilkan data	*Valid*
	Mengisi *form search* dan tidak ada kata kunci yang terkait	Gagal untuk menampilkan data	Gagal untuk menampilkan data	*Valid*
Ensiklopedia Tumbuhan	Menekan tombol navigasi ke bagian ensiklopedia	Berhasil menampilkan seluruh *list* data	Sukses menampilkan seluruh *list* data	*Valid*
Detail Tumbuhan	Menekan nama tumbuhan	Berhasil menampilkan detail data	Sukses menampilkan detail data	*Valid*

Tabel 2. Hasil Pengujian Sistem *Blackbox Admin*

Fungsi yang diuji	Kondisi	Output yang diharapkan	Output yang dihasilkan sistem	Status Pengujian
Menampilkan data tumbuhan	Menekan tombol *execute* pada *swagger find all*	Berhasil menampilkan seluruh data	Sukses menampilkan seluruh data	*Valid*
	Mengisi id dengan benar	Berhasil menampilkan data	Sukses menampilkan data	*Valid*
	Mengisi id dengan salah	Gagal menampilkan data	Gagal menampilkan data	*Valid*
Tambah data tumbuhan	Mengisi *form* JSON dengan benar	Berhasil menambahkan data	Sukses menambahkan data	*Valid*
	Mengisi *form* JSON dengan salah	Gagal menambahkan data	Gagal menambahkan data	*Valid*
Update data tumbuhan	Mengisi id dan *form* JSON dengan benar	Berhasil mengubah data	Sukses mengubah data	*Valid*
	Mengisi id dan *form* JSON dengan salah	Gagal mengubah data	Gagal mengubah data	*Valid*
Hapus data tumbuhan	Mengisi id dengan benar	Berhasil menghapus data	Sukses menghapus data	*Valid*
	Mengisi id dengan salah	Gagal menghapus data	Gagal menghapus data	*Valid*

Berdasarkan dari pengujian sistem yang dilakukan dengan menggunakan *blackbox* pada sistem *website* dapat dilihat status pengujian setiap fungsi bernilai *valid*. Dari hal tersebut, maka dapat disimpulkan bahwa sistem ini berjalan dengan baik sesuai dengan harapan.

Jurnal Indonesia : Manajemen Informatika dan Komunikasi
https://journal.stmiki.ac.id
E - ISSN: 2723-7079 | P - ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

4. Kesimpulan

Berdasarkan penelitian yang telah dilakukan dapat disimpulkan bahwa masalah umum mengenai kurangnya pengetahuan akan tumbuhan dan pentingnya tumbuhan dalam kehidupan manusia dapat teratasi dengan menggunakan *website* ensiklopedia tumbuhan Terra Floralis. Dengan menggunakan sistem ini, *admin* dapat melakukan pengolahan data tumbuhan. *User* akan terbantu karena dapat mengakses informasi mengenai tumbuhan. Penggunaan *Framework Spring Boot* bermanfaat untuk mengelola *Java Project* yang berfungsi memudahkan pengelolaan dependensi dan mengurangi kesulitan dalam konfigurasi serta kelebihan lannya dari *Spring Boot* yang berupa kemampuannya untuk berintegrasi dengan berbagai bahasa pemrograman lainnya, sehingga memungkinkan pembuatan aplikasi berbasis layanan *website* REST API. Sedangkan, *MyBatis Generator* memudahkan dalam sinkronisasi dengan *database* karena hasil *generate code* yang berbentuk model dan *mapper* telah disesuaikan dengan *database* yang dituju. Saran untuk pengembangan *website* ini adalah untuk ditingkatkan tampilan dari *website* agar dapat menarik perhatian pengguna dan membuat mereka nyaman dalam menjelajahi *website* Terra Floralis, serta dirapikan dan dikembangkan tampilan *website* tersebut pada *device handphone* ataupun *tablet*.

5. Ucapan Terima Kasih

Terima kasih sebesar-besarnya kepada Tuhan Yang Maha Esa atas anugerahnya setiap saat serta telah memberikan kesempatan dan kemudahan untuk dapat menyelesaikan penelitian ini dengan baik. Selain itu, penulis juga berterima kasih kepada seluruh pihak yang berkaitan dalam membantu penulis untuk menyelesaikan penelitian ini.

6. Daftar Pustaka

Asgar, H., & Hartono, B. (2022). Rancang Bangun Rest Api Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan. *Jurnal Informatika Teknologi dan Sains (Jinteks)*, *4*(1), 8-14. DOI: https://doi.org/10.51401/jinteks.v4i1.1474.

Chatterjee, S., & Mamatha, T. (2020). A comparative study on SOAP and RESTful web services. *Int. Res. J. Eng. Technol., no. May*, 2881-2885.

Dagha, W. C. U. (2021). Web Event, Spring Boot, Java Pembangunan Aplikasi Web Event menggunakan Framework Spring Boot di PT XYZ. *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, *8*(3), 1457-1469. DOI: https://doi.org/10.35957/jatisi.v8i3.1052.

De Bortoli, L., Underwood, C., & Thomson, S. (2023). PISA 2022. Reporting Australia's results. Volume I: Student performance and equity in education. DOI: https://doi.org/10.37517/978-1-74286-725-0.

DWI RIYANTO, N. R., ALFIAN, M., & YUHANA, U. L. (2023). Evaluasi Efisiensi Kinerja Object Relational Mapping pada Web API Point of Sale Menggunakan ISO/IEC 25010. *Jurnal Ilmu Komputer dan Agri-Informatika*, *10*(1). DOI: https://doi.org/10.29244/jika.10.1.57-69.

E - ISSN: 2723-7079 | P - ISSN: 2776-8074
Vol. 5 No. 2 (2024) | May
OPEN ACCESS
https://doi.org/10.35870/jimik.v5i2.712

Ginanjar, A., Sari, W. P., & Dwipriyoko, E. (2021). Perbandingan Kehandalan Operasi CRUD Menggunakan Perpaduan Spring dan MyBatis Framework serta Algoritma Cache Engine. *Jurnal Tiarsie*, *18*(1), 11-18. DOI: https://doi.org/10.32816/tiarsie.v18i1.90.

Ilman, B., & Ginanjar, A. (2019). Rancang Bangun Web Service-JSON Menggunakan Kombinasi Spring dan MyBatis Framework dalam lingkungan Java Platform. *Jurnal Teknologika*, *9*(1).

Liantoni, F. (2015). Klasifikasi Daun Dengan Perbaikan Fitur Citra Menggunakan Metode K-Nearest Neighbor. *Ultimatics: Jurnal Teknik Informatika*, *7*(2), 98-104. DOI: https://doi.org/10.31937/ti.v7i2.356.

Pranata, B. A. (2017). Perancangan Application Programming Interface (Api) Berbasis Web Menggunakan Gaya Arsitektur Representational State Transfer (Rest) Untuk Pengembangan Sistem Informasi Administrasi Pasien Klinik Perawatan Kulit.

Putra, B. P., & Susetyo, Y. A. (2020). Implementasi Api Master Store Menggunakan Flask, Rest Dan Orm Di Pt Xyz. *SISTEMASI*, *9*(3), 543-556. DOI: https://doi.org/10.32520/stmsi.v9i3.899.

Riyanto, N. R. D. (2018). Implementasi Object Relational Mapping (ORM) Pada Aplikasi Point Of Sale Berbasis Android (Studi Kasus: Toko Hard Ground).

Taiz, L., Zeiger, E., Møller, I. M., & Murphy, A. (2015). Plant physiology and Development.