

Nama : Muh. Nur Afrizal

NIM : 222212738

Kelas / No : 3SI1 / 22

LAPORAN PRAKTIKUM PEMROGRAMAN PLATFORM KHUSUS

MODUL 3

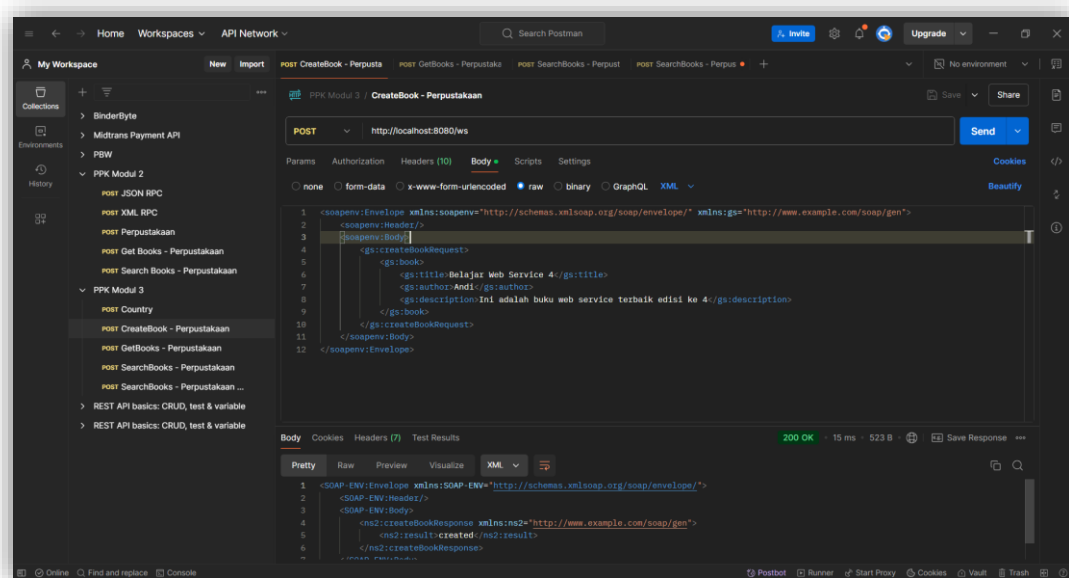
SIMPLE OBJECT ACCESS PROTOCOL (SOAP)

Ubahlah layanan perpustakaan pada modul praktikum 2 ke versi SOAP Web Service. Layanan yang dibuat adalah menambahkan koleksi buku dan mendapatkan semua koleksi buku. Lakukan pengujian untuk setiap layanan yang dibuat. Selanjutnya, buat laporan pekerjaan Anda dengan penjelasan lengkap dalam format pdf.

Link Repository Git: <https://github.com/afrzl/Pemrograman-Platform-Khusus/tree/master/Pertemuan%203/perpustakaan>

POSTMAN

a. CreateBook

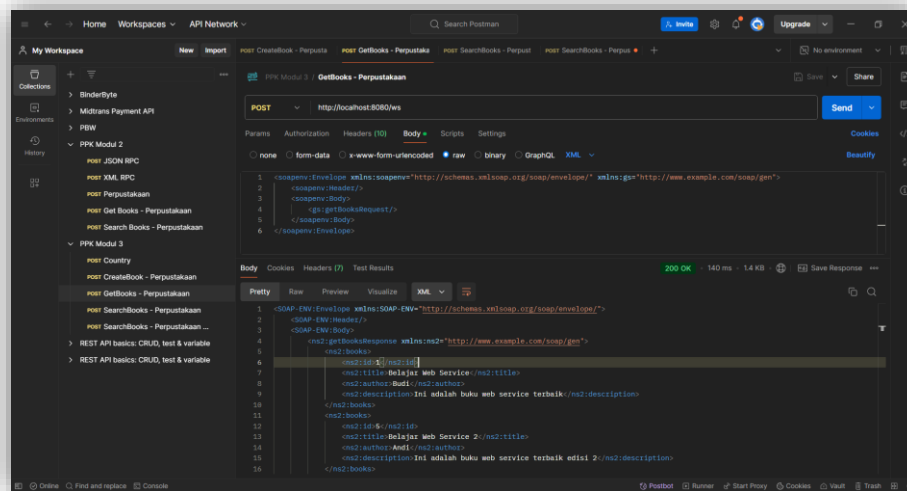


Nama : Muh. Nur Afrizal

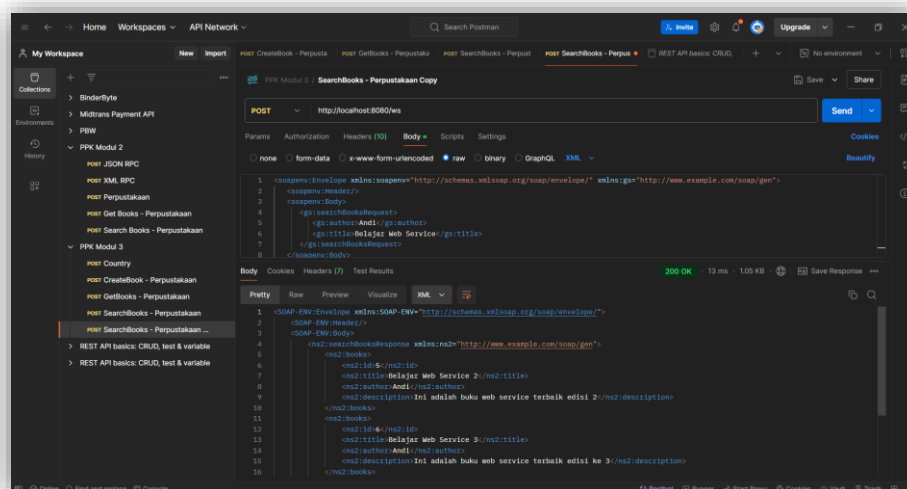
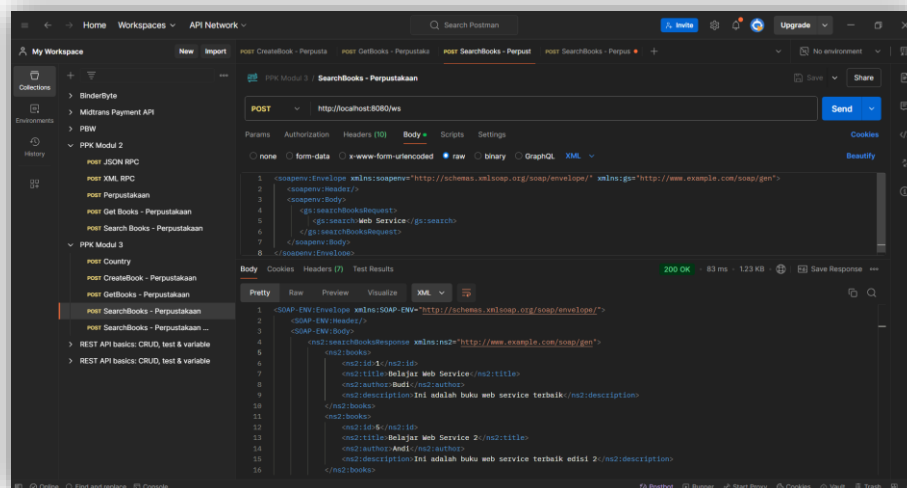
NIM : 222212738

Kelas / No : 3SI1 / 22

b. GetBooks



c. SearchBooks



Nama : Muh. Nur Afrizal

NIM : 222212738

Kelas / No : 3SI1 / 22

SOURCE CODE

a. books.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:tns="http://www.example.com/soap/gen"
  targetNamespace="http://www.example.com/soap/gen"
  elementFormDefault="qualified">

  <xs:element name="createBookRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="book" type="tns:book"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="createBookResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="result" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="getBooksRequest">
    <xs:complexType/>
  </xs:element>

  <xs:element name="getBooksResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="books" type="tns:book" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="searchBooksRequest">
    <xs:complexType>
      <xs:choice>
        <xs:sequence>
          <xs:element name="author" type="xs:string"/>
          <xs:element name="title" type="xs:string"/>
        </xs:sequence>
        <xs:element name="search" type="xs:string"/>
      </xs:choice>
    </xs:complexType>
  </xs:element>

  <xs:element name="searchBooksResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="books" type="tns:book" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="book">
    <xs:sequence>
      <xs:element name="id" type="xs:long" minOccurs="0"/>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="author" type="xs:string"/>
      <xs:element name="description" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

File XSD ini mendefinisikan struktur untuk layanan web SOAP terkait manajemen buku. Skema mencakup operasi untuk membuat, mengambil, dan mencari buku, serta mendefinisikan struktur data buku. Menggunakan namespace khusus, skema ini menyediakan blueprint yang jelas untuk implementasi dan penggunaan layanan web buku, memastikan konsistensi dalam pertukaran pesan SOAP.

Nama : Muh. Nur Afrizal

NIM : 222212738

Kelas / No : 3SI1 / 22

b. BookEntity (entity)

```
package org.example.entity;

import jakarta.persistence.*;

@Entity
@Table(name = "books")
public class BookEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String title;
    private String author;
    private String description;

    // Getter

    // Setter
}
```

Kelas BookEntity ini adalah entitas JPA yang merepresentasikan tabel 'books' dalam database. Memiliki empat atribut: id (kunci utama yang di-generate secara otomatis), title, author, dan description. Anotasi `@Entity` menandai kelas sebagai entitas JPA, sementara `@Table` menentukan nama tabel. Kelas ini menyediakan getter dan setter untuk semua atribut, memungkinkan akses dan modifikasi data buku dalam aplikasi.

c. BookMapper (mapper)

```
package org.example.repository;

import org.example.entity.BookEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;

import java.util.List;

public interface BookRepository extends JpaRepository<BookEntity, Long> {
    @Query("SELECT b FROM BookEntity b WHERE LOWER(b.title) LIKE LOWER(CONCAT('%', :searchTerm, '%')) OR LOWER(b.author) LIKE LOWER(CONCAT('%', :searchTerm, '%'))")
    List<BookEntity> searchBooks(@Param("searchTerm") String searchTerm);

    @Query("SELECT b FROM BookEntity b WHERE LOWER(b.author) LIKE LOWER(CONCAT('%', :author, '%')) AND LOWER(b.title) LIKE LOWER(CONCAT('%', :title, '%'))")
    List<BookEntity> searchBooksByAuthorAndTitle(@Param("author") String author, @Param("title") String title);
}
```

Interface BookRepository ini memperluas JpaRepository untuk entitas BookEntity, menyediakan operasi CRUD standar dan fungsionalitas pencarian khusus:

1. Mewarisi metode CRUD dasar dari JpaRepository untuk BookEntity.

Nama : Muh. Nur Afrizal

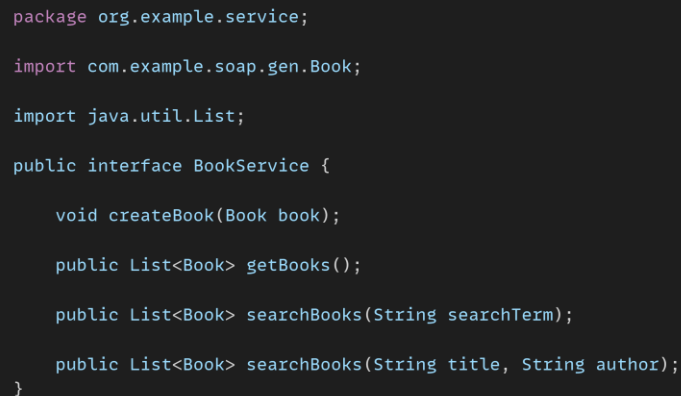
NIM : 222212738

Kelas / No : 3SI1 / 22

2. searchBooks: Mencari buku berdasarkan kata kunci dalam judul atau penulis, menggunakan query JPQL kustom.
3. searchBooksByAuthorAndTitle: Mencari buku berdasarkan penulis dan judul secara bersamaan.

Kedua metode pencarian menggunakan anotasi `@Query` dengan JPQL untuk mendefinisikan logika pencarian yang fleksibel, memungkinkan pencarian case-insensitive dengan operator LIKE.

d. BookService (service)



```
package org.example.service;

import com.example.soap.gen.Book;
import java.util.List;

public interface BookService {

    void createBook(Book book);

    public List<Book> getBooks();

    public List<Book> searchBooks(String searchTerm);

    public List<Book> searchBooks(String title, String author);
}
```

Nama : Muh. Nur Afrizal

NIM : 222212738

Kelas / No : 3SI1 / 22

e. **BookServiceImpl (service)**

```
package org.example.service;

import com.example.soap.gen.Book;
import org.example.entity.BookEntity;
import org.example.mapper.BookMapper;
import org.example.repository.BookRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.stream.Collectors;

@Service
public class BookServiceImpl implements BookService {
    @Autowired
    private BookRepository bookRepository;

    @Override
    public void createBook(Book book) {
        BookEntity entity = BookMapper.mapToEntity(book);
        bookRepository.save(entity);
    }

    @Override
    public List<Book> getBooks() {
        List<BookEntity> entities = bookRepository.findAll();
        return entities.stream()
            .map(BookMapper::mapToBook)
            .collect(Collectors.toList());
    }

    @Override
    public List<Book> searchBooks(String searchTerm) {
        return bookRepository.searchBooks(searchTerm).stream()
            .map(BookMapper::mapToBook)
            .collect(Collectors.toList());
    }

    @Override
    public List<Book> searchBooks(String author, String title) {
        return bookRepository.searchBooksByAuthorAndTitle(author, title).stream()
            .map(BookMapper::mapToBook)
            .collect(Collectors.toList());
    }
}
```

Kelas BookServiceImpl ini mengimplementasikan BookService, menyediakan logika bisnis untuk operasi buku:

1. Menggunakan BookRepository untuk akses data.
2. createBook: Menyimpan buku baru ke database.
3. getBooks: Mengambil semua buku dari database.
4. searchBooks(String): Mencari buku berdasarkan kata kunci.
5. searchBooks(String, String): Mencari buku berdasarkan penulis dan judul.

Nama : Muh. Nur Afrizal

NIM : 222212738

Kelas / No : 3SI1 / 22

Semua metode menggunakan BookMapper untuk konversi antara entitas database (BookEntity) dan objek SOAP (Book). Kelas ini menggunakan Stream API untuk transformasi data.

f. BookEndpoint

```
package org.example;

import com.example.soap.gen.*;
import org.example.service.BookService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.ws.server.endpoint.annotation.Endpoint;
import org.springframework.ws.server.endpoint.annotation.PayloadRoot;
import org.springframework.ws.server.endpoint.annotation.RequestPayload;
import org.springframework.ws.server.endpoint.annotation.ResponsePayload;

import java.util.List;

@Endpoint
public class BookEndpoint {
    private static final String NAMESPACE_URI = "http://www.example.com/soap/gen";

    @Autowired
    private BookService bookService;

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "createBookRequest")
    @ResponsePayload
    public CreateBookResponse createBook(@RequestPayload CreateBookRequest request) {
        bookService.createBook(request.getBook());

        CreateBookResponse response = new CreateBookResponse();
        response.setResult("created");
        return response;
    }

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "getBooksRequest")
    @ResponsePayload
    public GetBooksResponse getBooks(@RequestPayload GetBooksRequest request) {
        List<Book> books = bookService.getBooks();
        GetBooksResponse response = new GetBooksResponse();
        response.getBooks().addAll(books);
        return response;
    }

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "searchBooksRequest")
    @ResponsePayload
    public SearchBooksResponse searchBooks(@RequestPayload SearchBooksRequest request) {
        List<Book> searchResults;
        if (request.getSearch() != null) {
            searchResults = bookService.searchBooks(request.getSearch());
        } else {
            searchResults = bookService.searchBooks(request.getAuthor(), request.getTitle());
        }
        SearchBooksResponse response = new SearchBooksResponse();
        response.getBooks().addAll(searchResults);
        return response;
    }
}
```

Kelas BookEndpoint ini adalah endpoint SOAP yang menangani permintaan terkait buku:

1. Dianotasi dengan @Endpoint untuk dikenali sebagai endpoint SOAP.
2. Menggunakan BookService untuk operasi bisnis.
3. Memiliki tiga metode yang masing-masing menangani operasi berbeda:
 - createBook: Membuat buku baru.
 - getBooks: Mengambil semua buku.

Nama : Muh. Nur Afrizal

NIM : 222212738

Kelas / No : 3SI1 / 22

- searchBooks: Mencari buku berdasarkan kriteria.
4. Setiap metode dianotasi dengan `@PayloadRoot` untuk menentukan namespace dan elemen lokal yang ditangani.
 5. Menggunakan `@RequestPayload` untuk menerima data permintaan dan `@ResponsePayload` untuk mengirim respons.
 6. Metode `searchBooks` menangani dua jenis pencarian: dengan kata kunci tunggal atau dengan penulis dan judul.

Kelas ini berfungsi sebagai penghubung antara permintaan SOAP dan logika bisnis, mengonversi data antara format SOAP dan objek Java.

g. **WebServiceConfig**

```
package org.example;

import org.springframework.boot.web.servlet.ServletRegistrationBean;
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.core.io.ClassPathResource;
import org.springframework.ws.config.annotation.EnableWs;
import org.springframework.ws.transport.http.MessageDispatcherServlet;
import org.springframework.ws.wsdl.wsdl11.DefaultWsdl11Definition;
import org.springframework.xml.xsd.SimpleXsdSchema;
import org.springframework.xml.xsd.XsdSchema;

@EnableWs
@Configuration
public class WebServiceConfig {
    @Bean
    public ServletRegistrationBean<MessageDispatcherServlet> messageDispatcherServlet(ApplicationContext applicationContext) {
        MessageDispatcherServlet servlet = new MessageDispatcherServlet();
        servlet.setApplicationContext(applicationContext);
        servlet.setTransformWsdlLocations(true);
        return new ServletRegistrationBean<>(servlet, "/ws/");
    }

    @Bean(name = "books")
    public DefaultWsdl11Definition defaultWsdl11Definition(XsdSchema booksSchema) {
        DefaultWsdl11Definition wsdl11Definition = new DefaultWsdl11Definition();
        wsdl11Definition.setPortTypeName("BooksPort");
        wsdl11Definition.setLocationUri("/ws");
        wsdl11Definition.setTargetNamespace("http://www.example.com/soap/gen");
        wsdl11Definition.setSchema(booksSchema);
        return wsdl11Definition;
    }

    @Bean
    public XsdSchema booksSchema() {
        return new SimpleXsdSchema(new ClassPathResource("books.xsd"));
    }
}
```

Kelas `WebServiceConfig` ini mengkonfigurasi layanan web SOAP:

1. Dianotasi dengan `@EnableWs` dan `@Configuration` untuk mengaktifkan dukungan Spring WS.
2. Mendefinisikan `MessageDispatcherServlet` untuk menangani permintaan SOAP.
3. Mengkonfigurasi WSDL dengan `DefaultWsdl11Definition`, termasuk nama port, URI lokasi, dan namespace target.
4. Menyediakan skema XSD untuk layanan web melalui metode `booksSchema()`.

Nama : Muh. Nur Afrizal

NIM : 222212738

Kelas / No : 3SI1 / 22

Konfigurasi ini memungkinkan Spring Boot untuk mengekspos endpoint SOAP dan menghasilkan WSDL berdasarkan skema XSD yang didefinisikan, memudahkan integrasi dengan klien SOAP.