Laporan Modifikasi Aplikasi Perpustakaan

1. Pendahuluan

Aplikasi perpustakaan yang telah dikembangkan sebelumnya kini dimodifikasi dengan menambahkan RESTful API untuk mengelola layanan peminjaman dan pengembalian buku oleh anggota. Fitur-fitur baru ini meliputi: - Meminjam buku - Mengembalikan buku - Mendapatkan daftar peminjaman buku

2. Struktur Proyek

2.1. Struktur Kode

Berikut adalah struktur utama dari kode yang telah ditambahkan:

```
com.polstat.perpustakaan
│
├── controller
│   └── BorrowController.java
├── dto
│   └── BorrowDto.java
├── entity
│   └── Borrow.java
├── repository
│   └── BorrowRepository.java
└── service
    ├── BorrowService.java
    └── BorrowServiceImpl.java
```

3. Penjelasan Kode Program

3.1. Kelas Entity

Borrowing.java

```java
package com.polstat.perpustakaan.entity;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import java.time.LocalDate;

@Setter
@Getter
```

```java
@AllArgsConstructor
@NoArgsConstructor
@Builder
@Entity
@Table(name = "borrow")
public class Borrow {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false)
    private Long memberId;

    @Column(nullable = false)
    private Long bookId;

    @Column(nullable = false)
    private LocalDate borrowDate;

    @Column
    private LocalDate returnDate;

    @Column(nullable = false)
    private String borrowStatus;

    @Column
    private Integer overdueDays;
}
```

Kelas `Borrowing` merepresentasikan tabel peminjaman dalam database. Atribut yang terdapat dalam kelas ini mencakup ID peminjaman, ID anggota, ID buku, tanggal pinjam, tanggal kembali, status peminjaman, dan jumlah hari telat.

3.2. Kelas DTO

BorrowingDto.java

```java
package com.polstat.perpustakaan.dto;

import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;

import java.time.LocalDate;

@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
```

```
public class BorrowDto {
    private Long id;
    private Long memberId;
    private Long bookId;
    private LocalDate borrowDate;
    private LocalDate returnDate;
    private String borrowStatus;
    private Integer overdueDays;
}
```

Kelas BorrowingDto digunakan untuk mengemas data peminjaman yang akan dikirimkan melalui API.

3.3. Kelas Repository

BorrowingRepository.java

```
package com.polstat.perpustakaan.repository;

import com.polstat.perpustakaan.entity.Borrow;
import org.springframework.data.jpa.repository.JpaRepository;

public interface BorrowRepository extends JpaRepository<Borrow, Long> {
    // interface kosong karena sudah ada method bawaan dari JpaRepository
}
```

Kelas BorrowingRepository bertanggung jawab untuk melakukan operasi database pada entitas peminjaman.

3.4. Kelas Service

BorrowingService.java

```
package com.polstat.perpustakaan.service;

import com.polstat.perpustakaan.dto.BorrowDto;

import java.util.List;

public interface BorrowService {
    void borrowBook(BorrowDto borrowDto);
    void returnBook(Long borrowId);
    List<BorrowDto> getAllBorrowings();
}
```

Kelas BorrowingService mendefinisikan metode untuk meminjam buku, mengembalikan buku, dan mendapatkan daftar semua peminjaman.

BorrowingServiceImpl.java

```
package com.polstat.perpustakaan.service;

import com.polstat.perpustakaan.dto.BorrowDto;
```

```java
import com.polstat.perpustakaan.entity.Borrow;
import com.polstat.perpustakaan.mapper.BorrowMapper;
import com.polstat.perpustakaan.repository.BorrowRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

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

@Service
public class BorrowServiceImpl implements BorrowService {

    @Autowired
    private BorrowRepository borrowRepository;

    @Override
    public void borrowBook(BorrowDto borrowDto) {
        Borrow borrow = BorrowMapper.mapToBorrow(borrowDto);
        borrow.setBorrowDate(LocalDate.now());
        borrow.setBorrowStatus("Borrowed");
        borrowRepository.save(borrow);
    }

    @Override
    public void returnBook(Long borrowId) {
        Borrow borrow = borrowRepository.findById(borrowId).orElseThrow();
        borrow.setReturnDate(LocalDate.now());
        borrow.setBorrowStatus("Returned");

        // Logika kalkulasi hari keterlambatan
        LocalDate dueDate = borrow.getBorrowDate().plusDays(14); // 14 hari
batas waktu peminjaman
        if (borrow.getReturnDate().isAfter(dueDate)) {
            borrow.setOverdueDays((int) borrow.getReturnDate().toEpochDay() -
(int) dueDate.toEpochDay());
        } else {
            borrow.setOverdueDays(0);
        }

        borrowRepository.save(borrow);
    }

    @Override
    public List<BorrowDto> getAllBorrowings() {
        List<Borrow> borrowings = borrowRepository.findAll();
        return borrowings.stream()
                .map(BorrowMapper::mapToBorrowDto)
                .collect(Collectors.toList());
```

```
    }
}
```

3.5. Kelas Controller

BorrowingController.java

```
package com.polstat.perpustakaan.controller;

import com.polstat.perpustakaan.dto.BorrowDto;
import com.polstat.perpustakaan.service.BorrowService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/borrowing")
public class BorrowController {

    @Autowired
    private BorrowService borrowService;

    @PostMapping("/borrow")
    public ResponseEntity<String> borrowBook(@RequestBody BorrowDto
borrowDto) {
        borrowService.borrowBook(borrowDto);
        return ResponseEntity.ok("Book borrowed successfully");
    }

    @PostMapping("/return/{id}")
    public ResponseEntity<String> returnBook(@PathVariable Long id) {
        borrowService.returnBook(id);
        return ResponseEntity.ok("Book returned successfully");
    }

    @GetMapping("/all")
    public ResponseEntity<List<BorrowDto>> getAllBorrowings() {
        return ResponseEntity.ok(borrowService.getAllBorrowings());
    }
}
```

Kelas `BorrowingController` bertanggung jawab untuk menangani permintaan dari klien, mengarahkan permintaan ke metode layanan yang sesuai, dan mengembalikan respons.

4. Pengujian Endpoint

4.1. Meminjam Buku

Endpoint: `POST /api/borrowing/borrow`

Request Body:

```
{
    "memberId": 1,
    "bookId": 2
}
```

Response:

```
{
    "message": "Book borrowed successfully"
}
```

4.2. Mengembalikan Buku

Endpoint: `POST /api/borrowing/return/{id}`

URL: `http://localhost:8080/api/borrowing/return/1`

Response:

```
{
    "message": "Book returned successfully"
}
```

4.3. Mendapatkan Semua Peminjaman

Endpoint: `GET /api/borrowing/all`

Response:

```
[
    {
        "id": 1,
        "memberId": 1,
        "bookId": 2,
        "borrowDate": "2024-10-17",
        "returnDate": null,
        "status": "Borrowed",
        "overdueDays": 0
    }
]
```

5. Kesimpulan

Modifikasi aplikasi perpustakaan dengan menambahkan RESTful API untuk peminjaman dan pengembalian buku telah berhasil dilakukan. Dengan fitur ini, pengguna dapat meminjam dan mengembalikan buku dengan mudah melalui API. Pengujian yang dilakukan menggunakan Postman menunjukkan bahwa semua endpoint berfungsi dengan baik dan mengembalikan respons yang sesuai.

Laporan di atas dapat disesuaikan lebih lanjut jika diperlukan. Jika Anda memerlukan format PDF dari laporan ini, beri tahu saya!