4.6 Penugasan

Modifikasi aplikasi perpustakaan dengan menambahkan RESTful API untuk layanan peminjaman dan pengembalian buku oleh member. Atribut peminjaman buku antara lain: id member yang meminjam, id buku yang dipinjam, tanggal pinjam, tanggal kembali, status peminjaman (sedang dipinjam, sudah dikembalikan), dan jumlah hari telat dikembalikan.

Jawab :

Loan.java

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

import jakarta.persistence.*;
import lombok.*;
import java.time.LocalDate;

@Setter
@Getter
@AllArgsConstructor
@NoArgsConstructor
@Builder
@Entity
@Table(name = "loans")
public class Loan {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @ManyToOne
    @JoinColumn(name = "member_id", nullable = false)
    private Member member;

    @ManyToOne
    @JoinColumn(name = "book_id", nullable = false)
    private Book book;

    @Column(nullable = false)
    private LocalDate borrowDate;

    @Column(nullable = true)
    private LocalDate returnDate;
```

```
    @Column(nullable = false)
    @Enumerated(EnumType.STRING)
    private LoanStatus loanStatus;

    @Column(nullable = true)
    private Integer overdueDays;

    public enum LoanStatus {
        BORROWED,
        RETURNED
    }
}
```

Kelas ini menyediakan cara yang terstruktur untuk merepresentasikan dan mengelola peminjaman dalam sistem perpustakaan, memungkinkan pelacakan anggota mana yang meminjam buku yang mana, tanggal peminjaman dan status terkini dari peminjaman tersebut.

LoanController.java

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

import com.polstat.perpustakaan.dto.LoanDto;
import com.polstat.perpustakaan.service.LoanService;
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/loans")
public class LoanController {

    @Autowired
    private LoanService loanService;

    @PostMapping("/loan")
    public ResponseEntity<String> loanBook(@RequestBody LoanDto loanDto) {
        loanService.loanBook(loanDto);
        return ResponseEntity.ok("Buku sukses terpinjam!");
    }

    @PostMapping("/return/{id}")
    public ResponseEntity<String> returnBook(@PathVariable Long id) {
```

```
        loanService.returnBook(id);
        return ResponseEntity.ok("Buku sukses dikembalikan!");
    }

    @GetMapping("/getall")
    public ResponseEntity<List<LoanDto>> getAllLoans() {
        return ResponseEntity.ok(loanService.getAllLoans());
    }
}
```

Kelas ini bertanggung jawab dalam mengelola proses peminjaman serta pengembalian dari buku di perpustakaan. Kelas ini mengatur tiga metode utama, diantaranya loanBook, returnBook, dan getAllLoans. loanBook merupakan metode untuk memproses peminjaman buku, returnBook untuk memproses pengembalian buku berdasarkan id peminjaman, dan getAllLoans untuk mengambil dan mengembalikan semua catatan peminjaman yang ada.

LoanDto.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 LoanDto {
    private Long id;
    private Long memberId;
    private Long bookId;
    private LocalDate borrowDate;
    private LocalDate returnDate;
    private String loanStatus;
    private Integer overdueDays;
}
```

Kelas ini berfungsi sebagai wadah untuk menyimpan dan mentransfer data terkait peminjaman buku.

LoanMapper.java

```
package com.polstat.perpustakaan.mapper;

import com.polstat.perpustakaan.dto.LoanDto;
import com.polstat.perpustakaan.entity.Loan;
import com.polstat.perpustakaan.entity.Book;
import com.polstat.perpustakaan.entity.Member;

public class LoanMapper {

    public static Loan mapToLoan(LoanDto loanDto, Member member, Book book) {
        return Loan.builder()
                .id(loanDto.getId())
                .member(member)
                .book(book)
                .borrowDate(loanDto.getBorrowDate())
                .returnDate(loanDto.getReturnDate())
                .loanStatus(Loan.LoanStatus.valueOf(loanDto.getLoanStatus().toUpp
erCase())) // Convert String to Enum
                .overdueDays(loanDto.getOverdueDays())
                .build();
    }

    public static LoanDto mapToLoanDto(Loan loan) {
        return LoanDto.builder()
                .id(loan.getId())
                .memberId(loan.getMember().getId())
                .bookId(loan.getBook().getId())
                .borrowDate(loan.getBorrowDate())
                .returnDate(loan.getReturnDate())
                .loanStatus(loan.getLoanStatus().name()) // Convert Enum to
String
                .overdueDays(loan.getOverdueDays())
                .build();
    }
}
```

Kelas ini berfungsi untuk memudahkan konversi antara dua jenis objek yang berhubungan dengan peminjaman buku. Ini membantu memisahkan logika antara penyimpanan data dan pengiriman data, yang penting untuk menjaga kebersihan dan ketetaruran kode dalam aplikasi.

LoanRepository.java

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

import com.polstat.perpustakaan.entity.Loan;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface LoanRepository extends JpaRepository<Loan, Long> {

}
```

Kelas ini berfungsi untuk melakukan operasi database pada entitas peminjaman.

LoanService.java

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

import com.polstat.perpustakaan.dto.LoanDto;

import java.util.List;

public interface LoanService {
    void loanBook(LoanDto LoanDto);
    void returnBook(Long LoanId);
    List<LoanDto> getAllLoans();
}
```

Kelas ini mendefinisikan metode yang terkait dengan layanan peminjaman buku, diantaranya yaitu untuk meminjam, mengembalikan, dan mendapatkan list dari semua peminjaman yang sudah terjadi.

LoanServiceImpl.java

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

import com.polstat.perpustakaan.dto.LoanDto;
import com.polstat.perpustakaan.entity.Loan;
import com.polstat.perpustakaan.entity.Book;
import com.polstat.perpustakaan.entity.Member;
import com.polstat.perpustakaan.mapper.LoanMapper;
import com.polstat.perpustakaan.repository.LoanRepository;
import com.polstat.perpustakaan.repository.BookRepository;
```

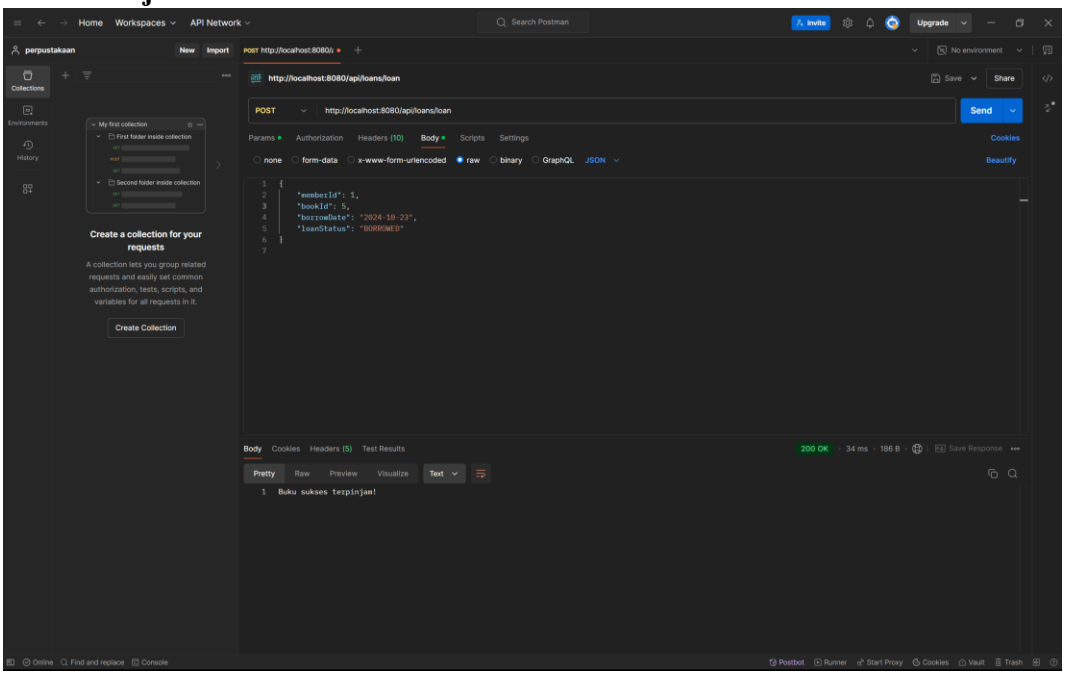
```java
import com.polstat.perpustakaan.repository.MemberRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

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

@Service
public class LoanServiceImpl implements LoanService {

    @Autowired
    private LoanRepository loanRepository;

    @Autowired
    private MemberRepository memberRepository;

    @Autowired
    private BookRepository bookRepository;

    @Override
    public void loanBook(LoanDto loanDto) {
        Member member = memberRepository.findById(loanDto.getMemberId())
                .orElseThrow(() -> new IllegalArgumentException("Member tidak
ditemukan"));

        Book book = bookRepository.findById(loanDto.getBookId())
                .orElseThrow(() -> new IllegalArgumentException("Buku tidak
ditemukan"));

        Loan loan = LoanMapper.mapToLoan(loanDto, member, book);
        loan.setBorrowDate(LocalDate.now());
        loan.setLoanStatus(Loan.LoanStatus.BORROWED);
        loanRepository.save(loan);
    }

    @Override
    public void returnBook(Long borrowId) {
        Loan loan = loanRepository.findById(borrowId)
                .orElseThrow(() -> new IllegalArgumentException("Loan tidak
ditemukan"));

        loan.setReturnDate(LocalDate.now());
        loan.setLoanStatus(Loan.LoanStatus.RETURNED);
```

```
        LocalDate dueDate = loan.getBorrowDate().plusDays(14);
        if (loan.getReturnDate().isAfter(dueDate)) {
            loan.setOverdueDays((int) loan.getReturnDate().toEpochDay() - (int)
dueDate.toEpochDay());
        } else {
            loan.setOverdueDays(0);
        }

        loanRepository.save(loan);
    }

    @Override
    public List<LoanDto> getAllLoans() {
        List<Loan> loans = loanRepository.findAll();
        return loans.stream()
                .map(LoanMapper::mapToLoanDto)
                .collect(Collectors.toList());
    }
}
```

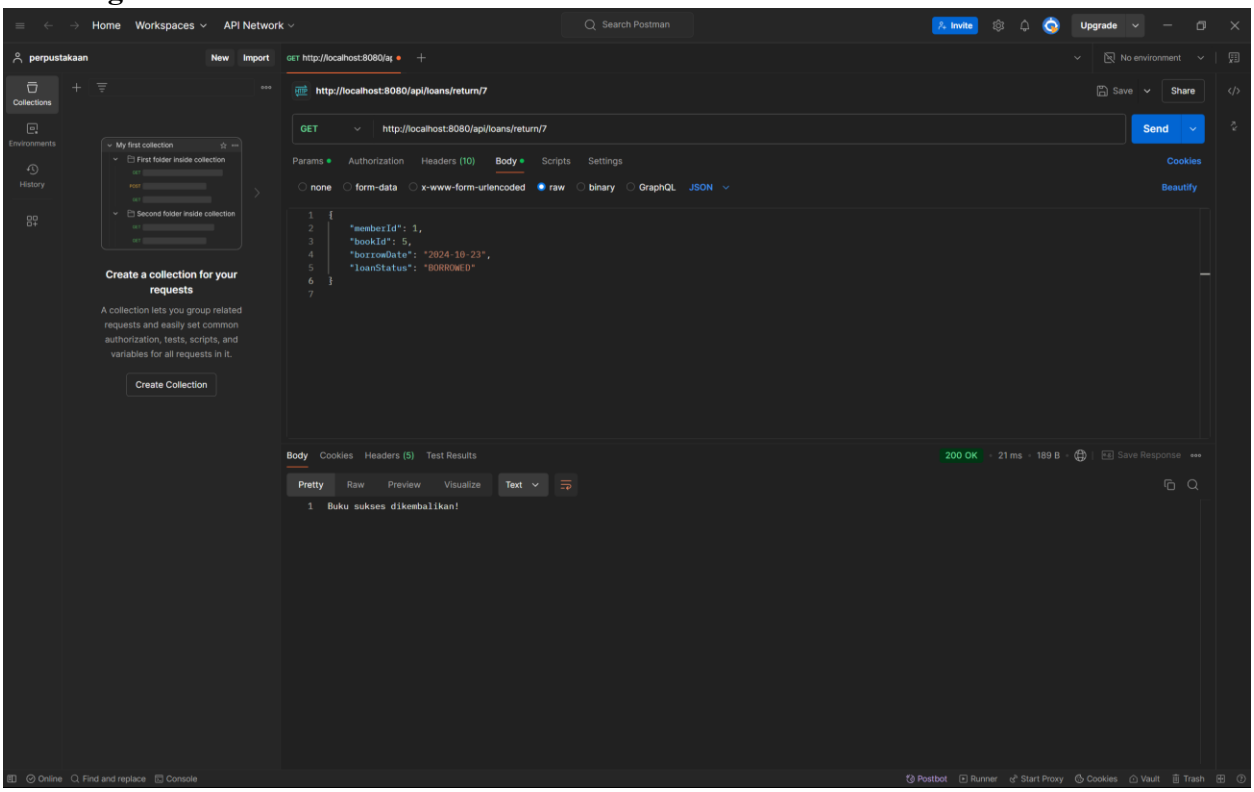
Kelas ini mengimplementasikan logika bisnis untuk mengelola peminjaman buku. Ini mencakup peminjaman dan pengembalian buku, serta pengambilan daftar semua peminjaman yang ada.

Uji Coba Postman

- **Tes Peminjaman**



- **Tes Pengembalian**



- **Tes Melihat Semua Buku yang Sudah Dipinjam**