FITNESS CLASS BOOKING PROGRAM

**ABSTRACT:**

The Fitness Class Booking Platform is a comprehensive system developed using a Database Management System (DBMS) to streamline the management of fitness classes for gyms and fitness centers. It features user registration, class scheduling, booking, instructor management, secure payment processing, location information, membership options, attendance tracking, and a review system. This platform enhances user experience by providing a convenient interface for booking and managing fitness activities. It improves operational efficiency for administrators through simplified schedule and instructor management. By leveraging a relational DBMS, the platform ensures data integrity, security, and efficient data retrieval, promoting a more organized and engaging fitness environment.

**INTRODUCTION**

In the digital era, fitness centers and gyms need innovative solutions to meet the demands of their clients and streamline operations. The Fitness Class Booking Platform is designed to revolutionize fitness management by leveraging a robust Database Management System (DBMS). This platform integrates user registration, class scheduling, booking, instructor management, secure payments, and attendance tracking into one seamless system.

Users can easily browse and book classes, while instructors manage their schedules effortlessly. Administrators benefit from simplified operations and real-time updates, ensuring data integrity and security. The Fitness Class Booking Platform not only enhances user convenience but also optimizes administrative efficiency, making it a game-changer in the fitness industry.

FUNCTIONAL REQUIREMENTS

There are some functional requirements for a Fitness Class Booking Platform entities:

* **User Management:**

Users should be able to create accounts, log in, and update their profiles.

Different user roles might exist, such as regular users and administrators.

* **Class Management:**

Ability for administrators to add, edit, and delete classes.

Users should be able to view available classes, including details like time, duration, instructor, and location.

Classes may have different categories (e.g., yoga, cardio, strength training) that users can browse.

* **Booking Management**:

Users should be able to book classes based on availability.

The system should prevent overbooking and notify users if a class is already full.

Users should be able to cancel bookings within a certain time frame.

* **Instructor Management**:

Administrators should be able to add, edit, and remove instructors.

Instructors should have profiles displaying their qualifications, specialties, and schedule.

* **Payment Management:**

Integration with a payment gateway to facilitate secure online payments for booking classes or purchasing memberships.

Ability to handle different payment methods like credit/debit cards, digital wallets, or membership credits.

* **Schedule Management:**

Users should be able to view a schedule of upcoming classes.

The schedule should be filterable by date, time, class type, and instructor.

* **Location Management**:

Ability to add, edit, and remove locations where classes are held.

Users should be able to view location details including address, contact information, and amenities.

* **Membership Management:**

Users should have the option to purchase memberships with different pricing tiers and benefits.

Administrators should be able to manage membership plans, including setting prices, durations, and access levels.

* **Attendance Tracking:**

Ability to track and record attendance for each class.

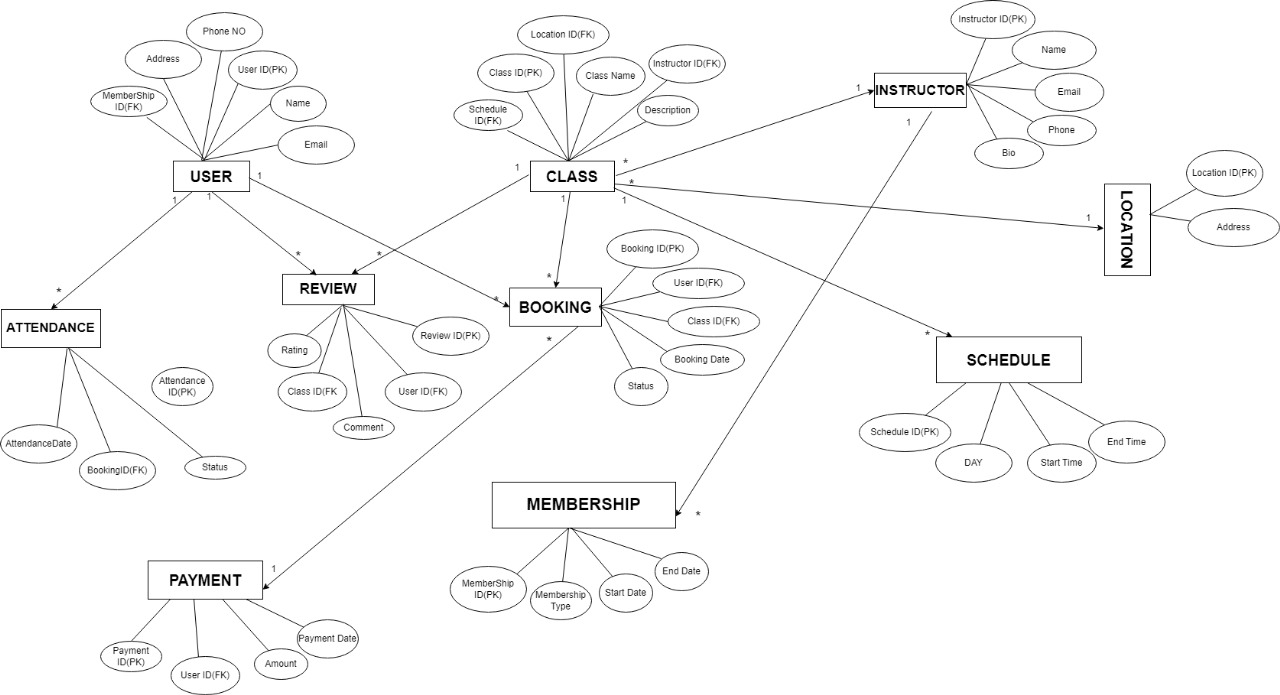
Administrators should have access to attendance reports and statistics.

* **Review System:**

Users should be able to leave reviews and ratings for classes and instructors.

Reviews should be visible to other users to help them make informed decisions.

ER-DIAGRAM



QUERIES TO CREATE DB

create database fitness\_class;

use fitness\_class;

show databases;

CREATE TABLE Membership (

MembershipID INT PRIMARY KEY,

MembershipType VARCHAR(50),

StartDate DATE,

EndDate DATE

);

CREATE TABLE Instructor (

InstructorID INT PRIMARY KEY,

Name VARCHAR(100),

Email VARCHAR(100),

Phone VARCHAR(15),

Bio TEXT

);

CREATE TABLE Location (

LocationID INT PRIMARY KEY,

Address VARCHAR(255)

);

CREATE TABLE Schedule (

ScheduleID INT PRIMARY KEY,

DayOfWeek VARCHAR(20),

StartTime TIME,

EndTime TIME

);

CREATE TABLE Class (

ClassID INT PRIMARY KEY,

ClassName VARCHAR(100),

Description VARCHAR(255),

ScheduleID INT,

LocationID INT,

InstructorID INT

);

CREATE TABLE User (

UserID INT PRIMARY KEY,

Name VARCHAR(100),

Email VARCHAR(100),

Address VARCHAR(255),

PhoneNo VARCHAR(15),

MembershipID INT

);

CREATE TABLE Booking (

BookingID INT PRIMARY KEY,

UserID INT,

ClassID INT,

BookingDate DATE,

Status VARCHAR(50));

CREATE TABLE Payment (

PaymentID INT PRIMARY KEY,

UserID INT,

Amount DECIMAL(10, 2),

PaymentDate DATE

);

CREATE TABLE Attendance (

AttendanceID INT PRIMARY KEY,

BookingID INT,

AttendanceDate DATE,

Status VARCHAR(50)

);

CREATE TABLE Review (

ReviewID INT PRIMARY KEY,

UserID INT,

ClassID INT,

Rating INT,

Comment TEXT,

ReviewDate DATE

);

ALTER TABLE Class

ADD FOREIGN KEY (ScheduleID) REFERENCES Schedule(ScheduleID),

ADD FOREIGN KEY (LocationID) REFERENCES Location(LocationID),

ADD FOREIGN KEY (InstructorID) REFERENCES Instructor(InstructorID);

ALTER TABLE User

ADD FOREIGN KEY (MembershipID) REFERENCES Membership(MembershipID);

ALTER TABLE Booking

ADD FOREIGN KEY (UserID) REFERENCES User(UserID),

ADD FOREIGN KEY (ClassID) REFERENCES Class(ClassID);

ALTER TABLE Payment

ADD FOREIGN KEY (UserID) REFERENCES User(UserID);

ALTER TABLE Attendance

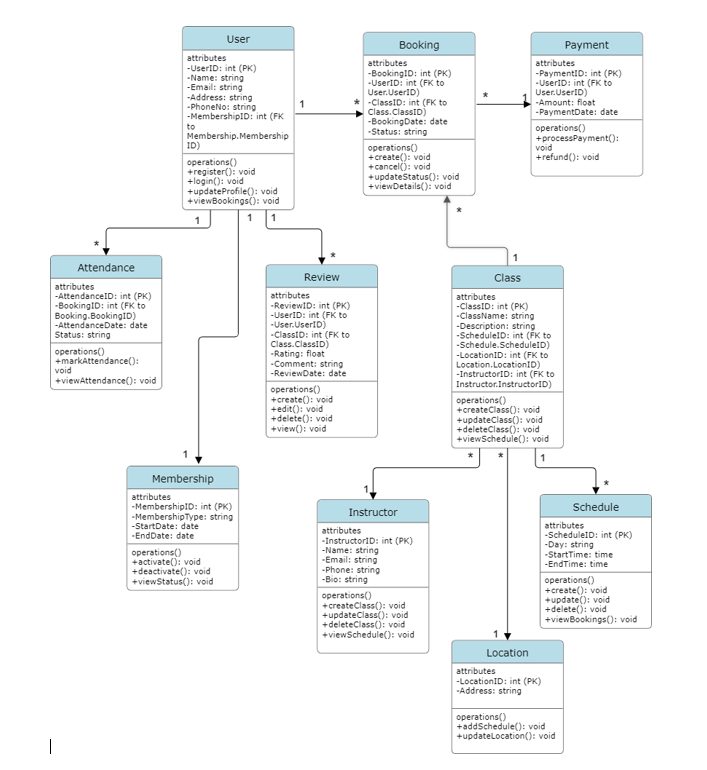
ADD FOREIGN KEY (BookingID) REFERENCES Booking(BookingID);

ALTER TABLE Review

ADD FOREIGN KEY (UserID) REFERENCES User(UserID),

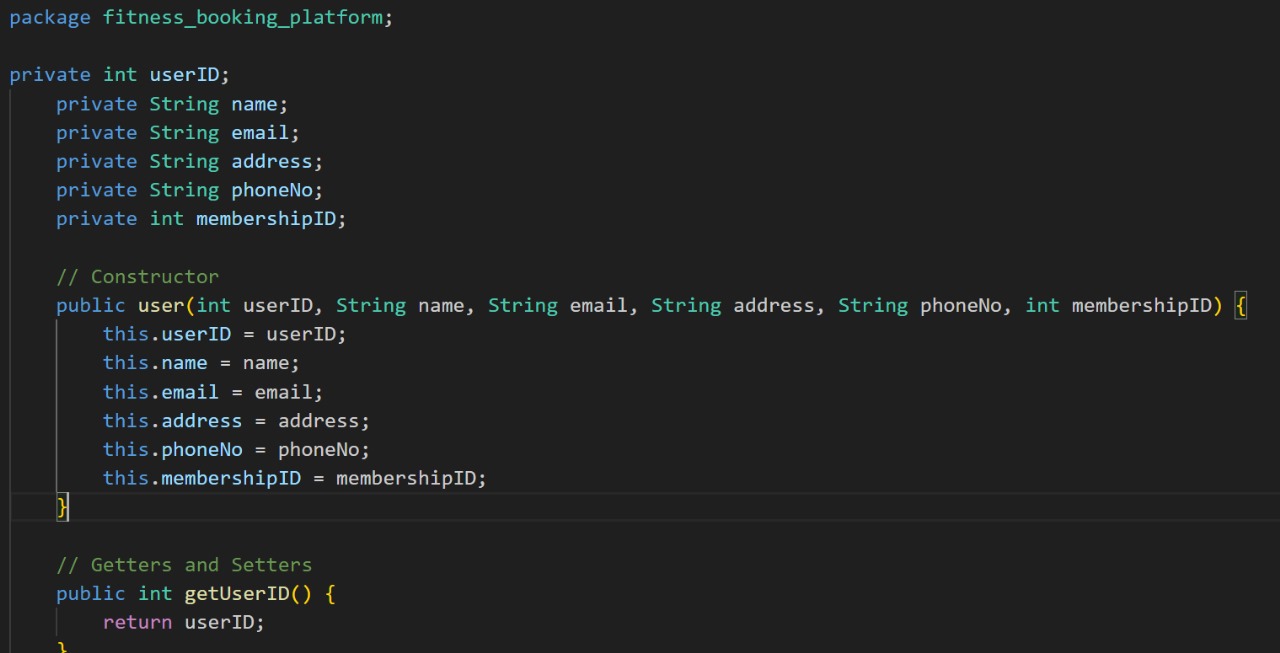
ADD FOREIGN KEY (ClassID) REFERENCES Class(ClassID);

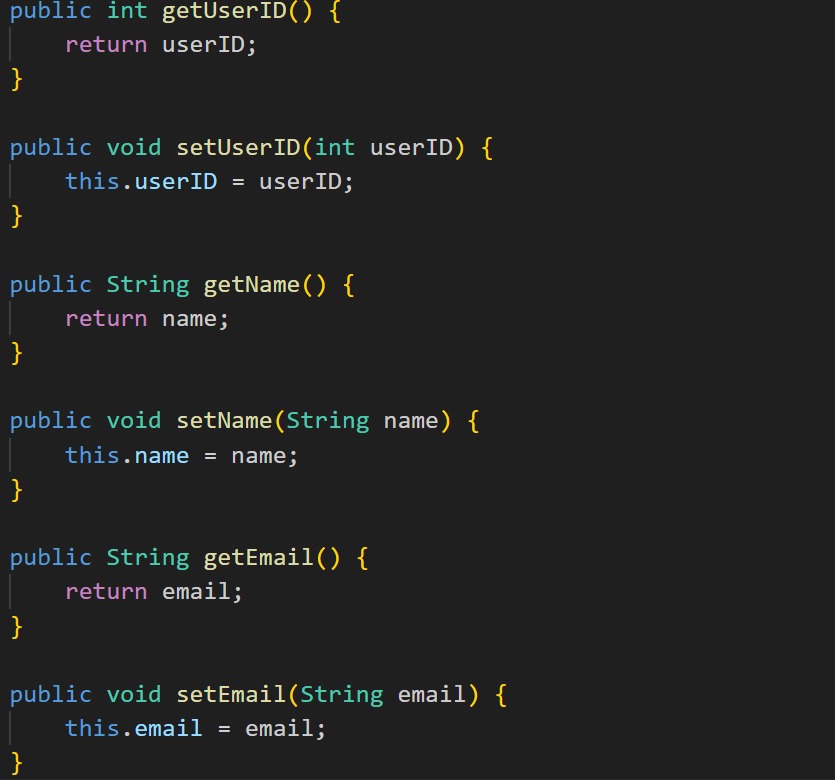
UML-DIAGRAM

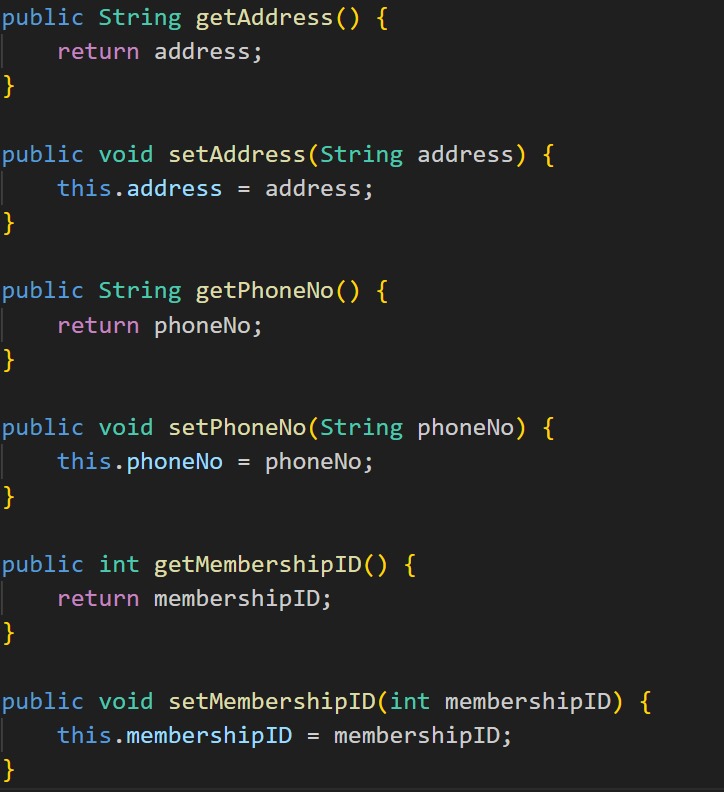


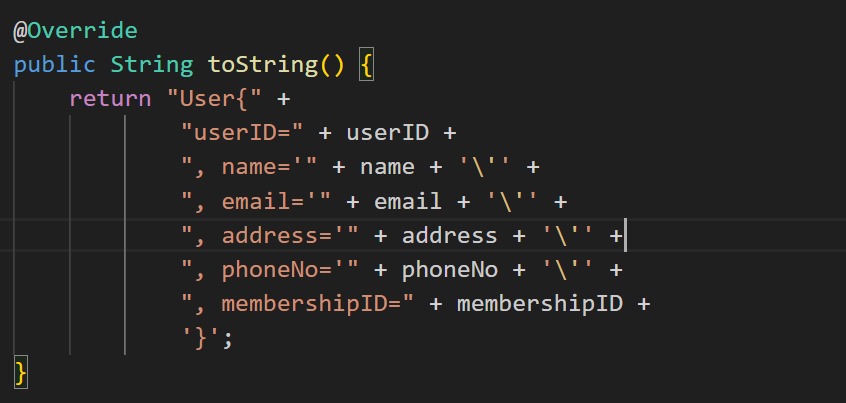
ALL CLASS JAVA FILE CODE

User.java

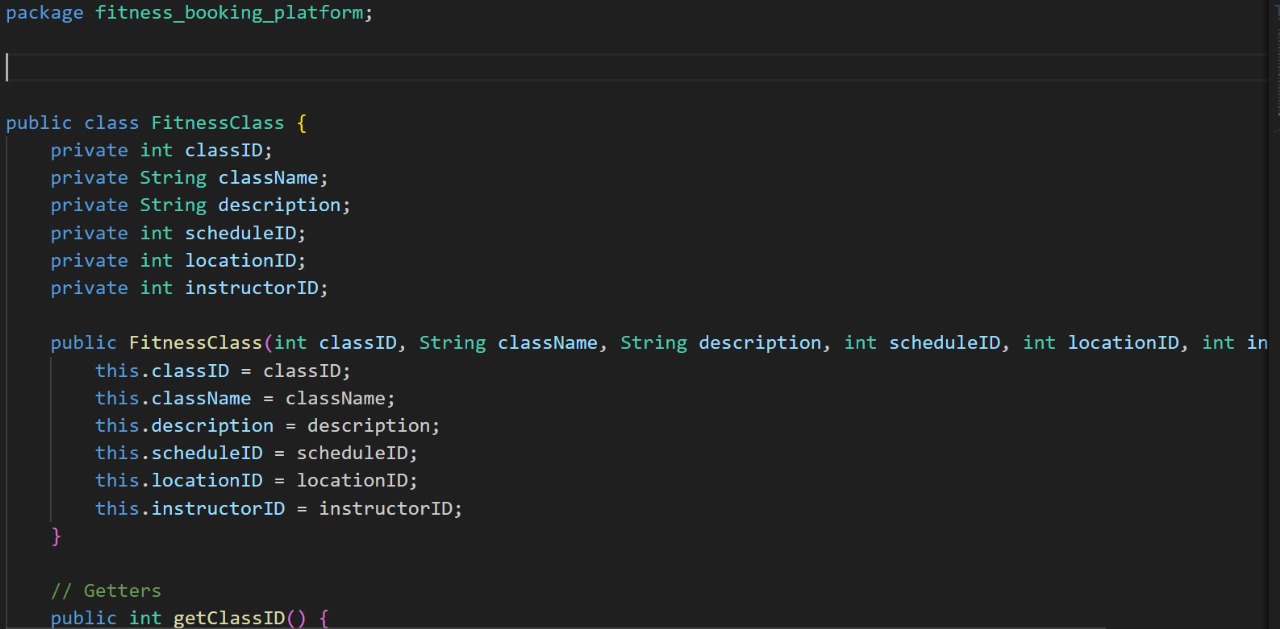


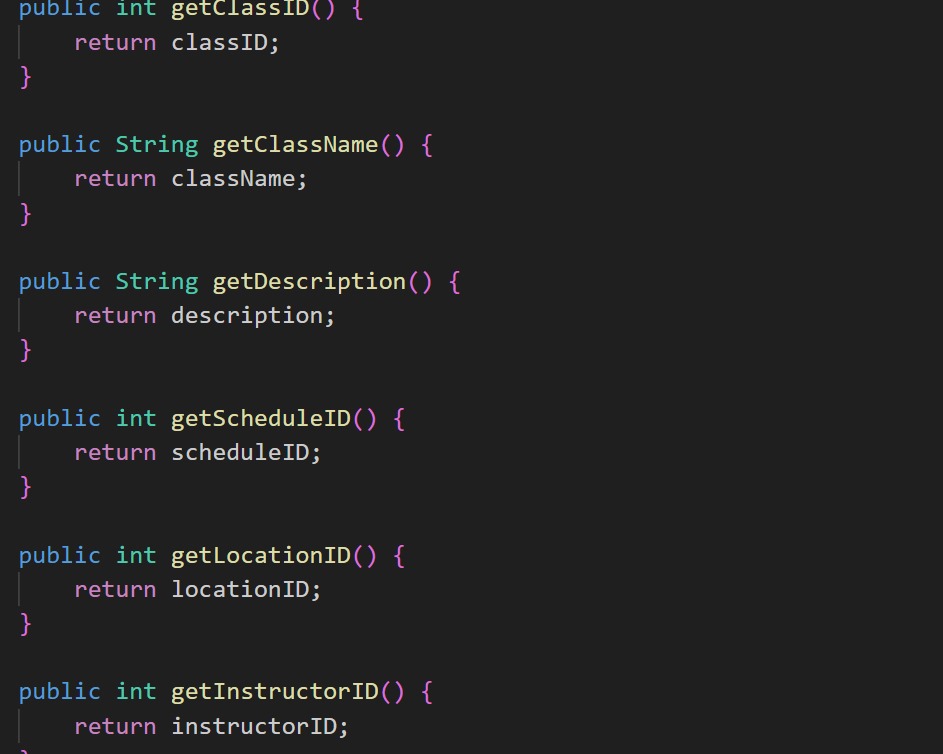


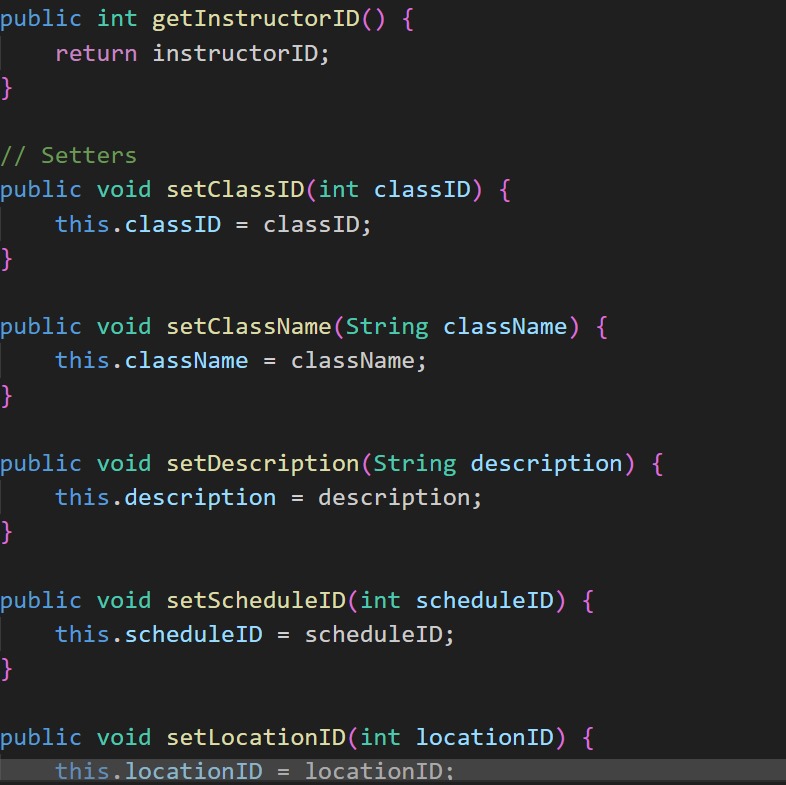


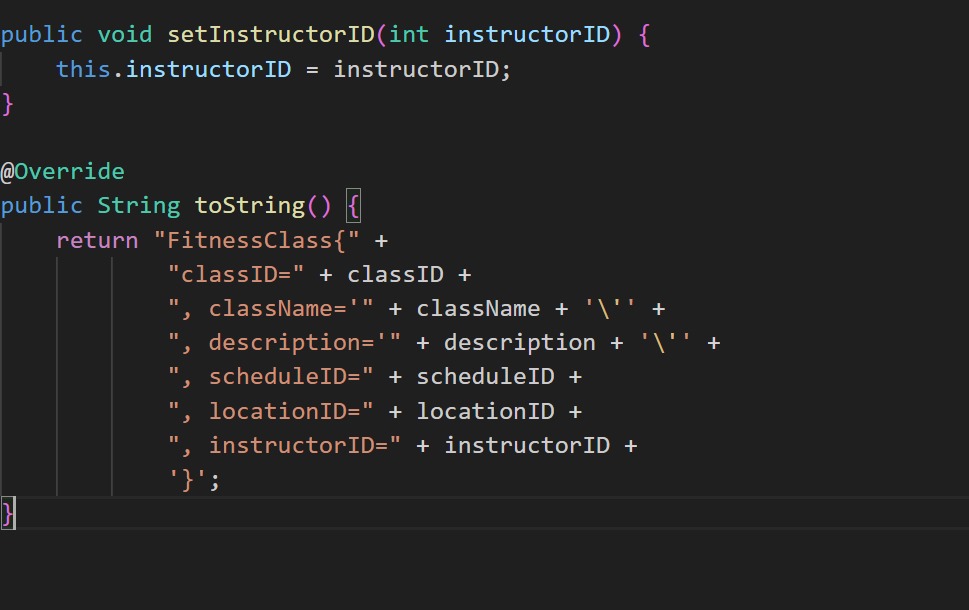


Class.java

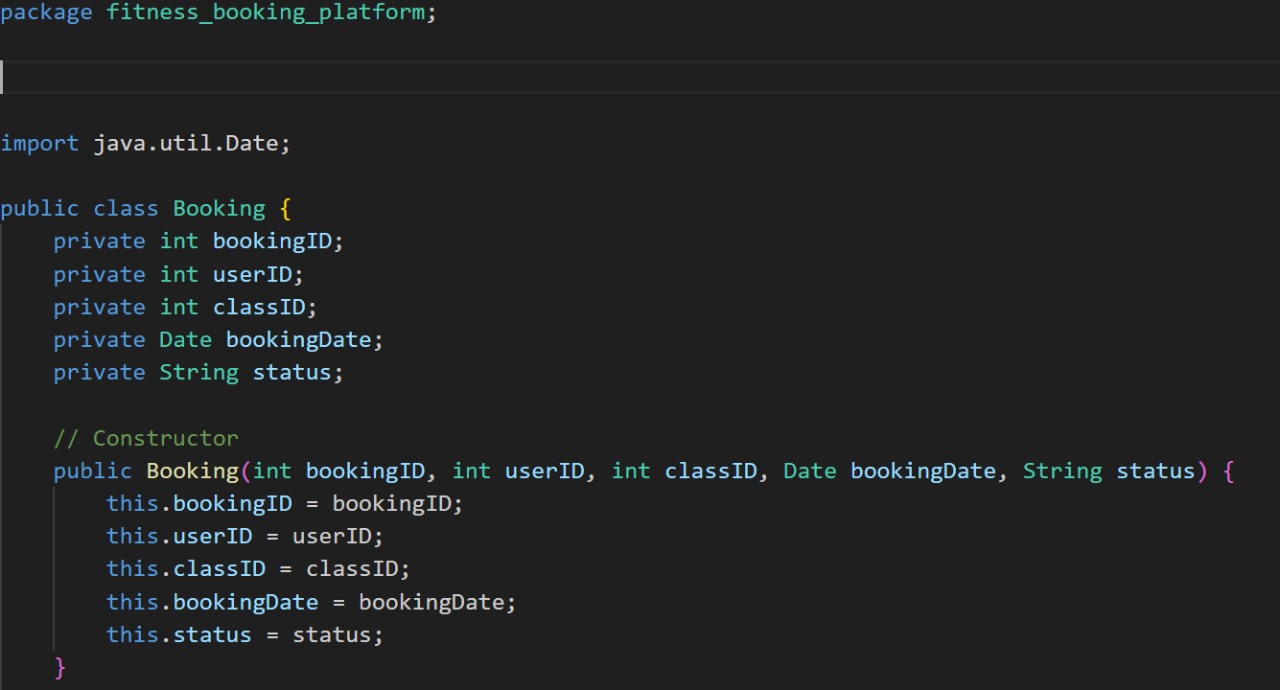


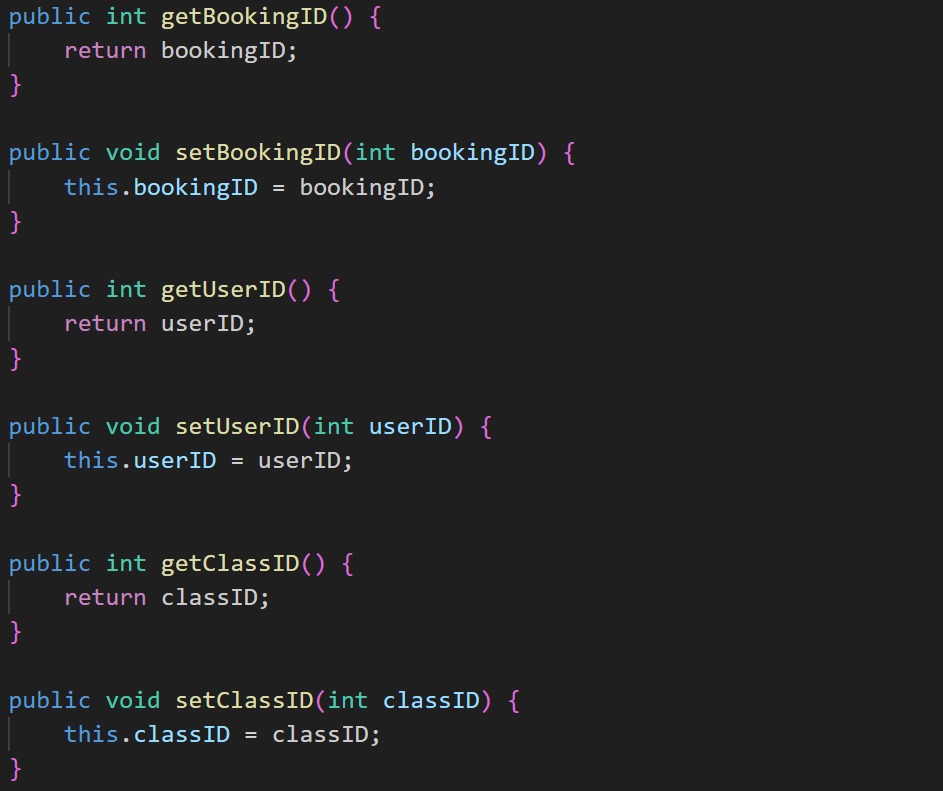






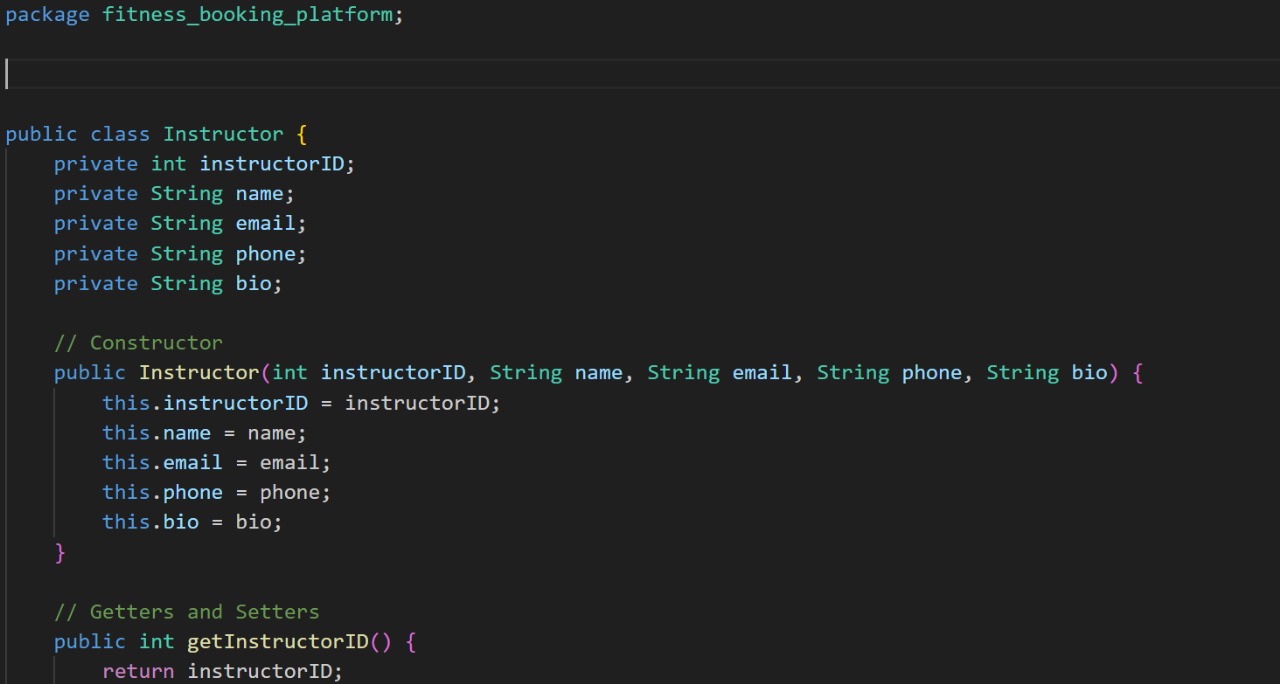
Booking.java

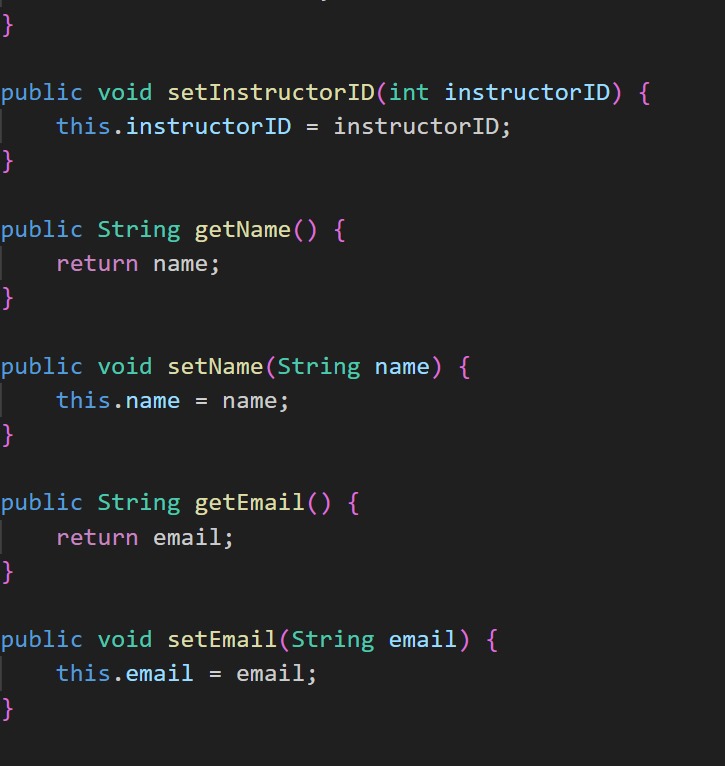


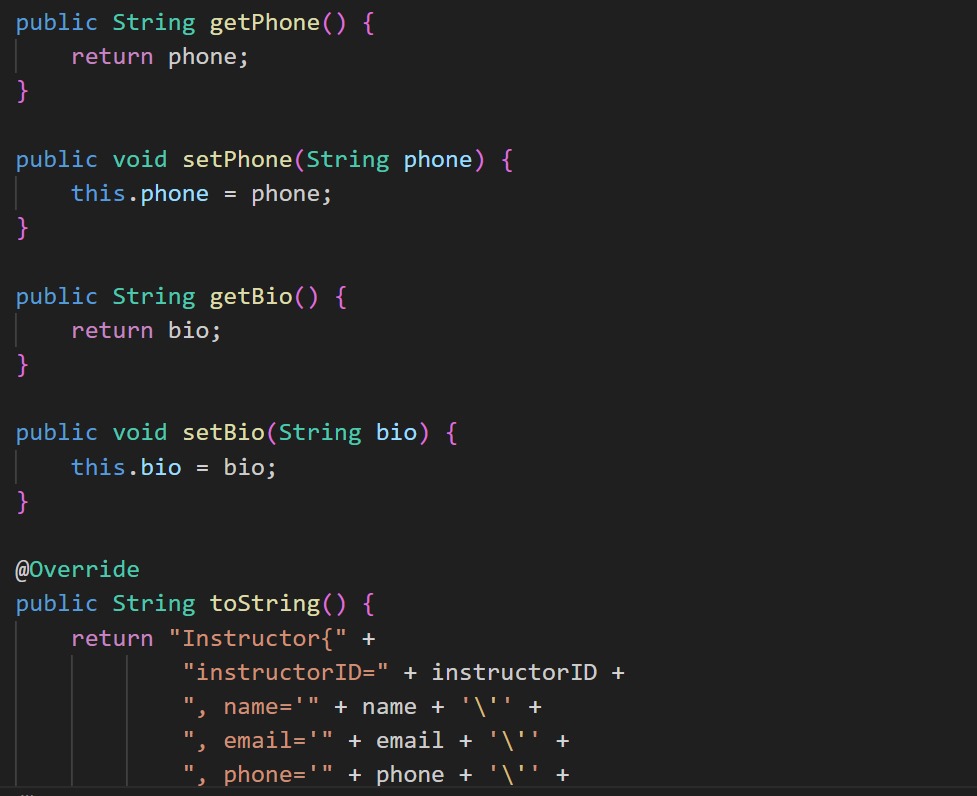




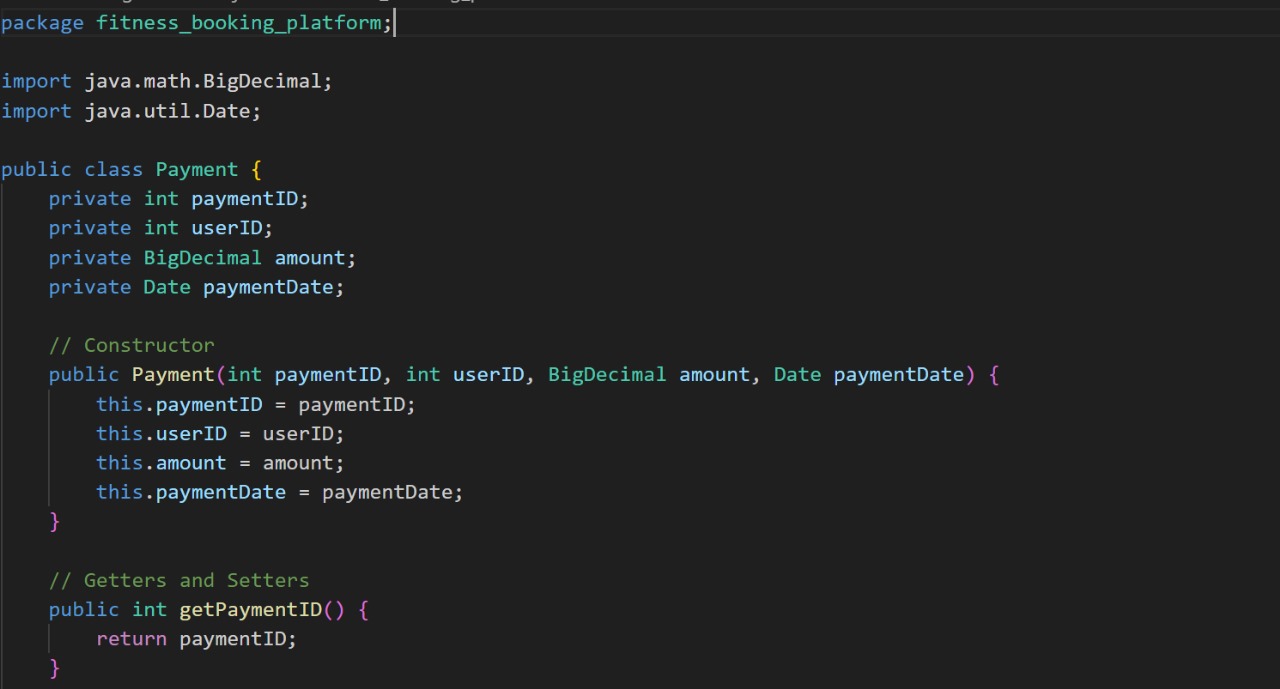
Instructor.java

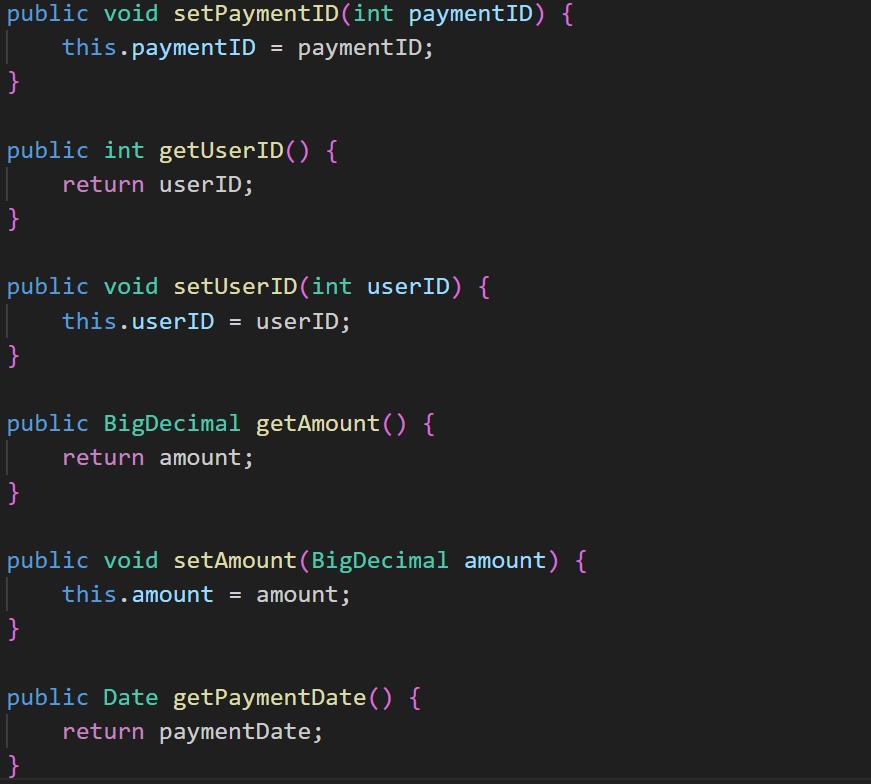


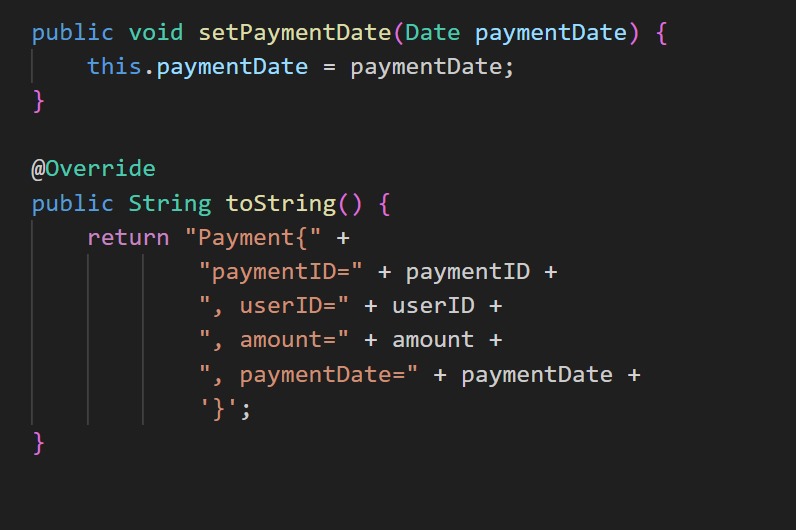


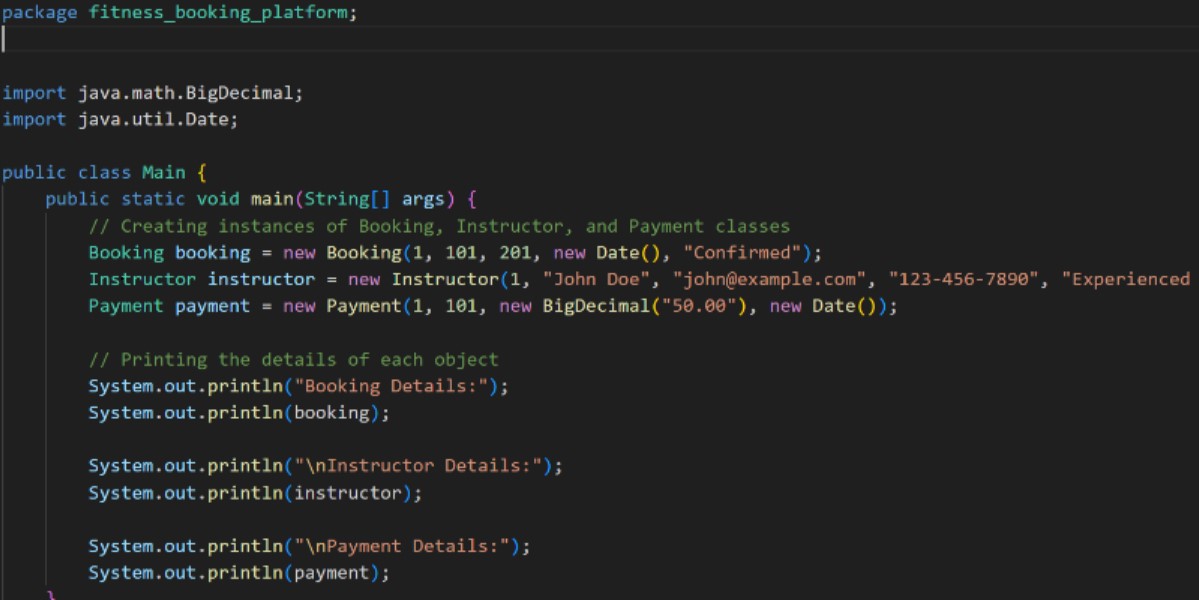


Payment.java









CHALLENGES LIST:

1)Problem faced when accurately capturing all user requirements and understanding scope.

2)Ensuring Data Integrity and consistency across multiple transactions and concurrent access.

3)Ensuring data is backed-up regularly and can be recovered in case of failure.

4)Keeping the database system updated and well-maintained overtime.