Two Wheeler Technician Appointment System

Team:

Pranav Tripathi

Rupak

Shreya Rastogi

Soumyajit Das

Yash Todwal

Table of Contents

1. Introduction

1.1. Purpose of the Document

The purpose of this document is to provide an overview of the Online Two Wheeler Technician Appointment System project developed using Core Java. It outlines the system's requirements, architecture and features. This report aims to explain the key components, functionalities, and interactions within the system, as well as highlight the benefits and potential challenges of such an architecture.

1.2. Project Overview

The Online Two Wheeler Technician Appointment System is an application that allows customers & two wheeler technicians to perform various Two Wheeler Technician Appointment activities online, such as registration, booking an appointment, viewing different services, receiving appointments, modifying appointments, cancelling it, payment. The system aims to provide a secure and user-friendly interface for customers to access their accounts conveniently.

This is done by distributing the functions across three different classes: Customer, Technician and Service. The functionalities are unified by another “Appointment” class.

1.3. Scope

The scope of the Online Two Wheeler Technician Appointment System project includes the following functionalities:

- Account management (create, update, delete accounts).

- Booking appointments, Modifying, cancelling it.

- Viewing history of appointments.

2. System Requirements

2.1. Functional Requirements

1. User Registration: Users should be able to create new accounts by providing their details.

2. Account Management: Users should be able to create, update, and delete their accounts.

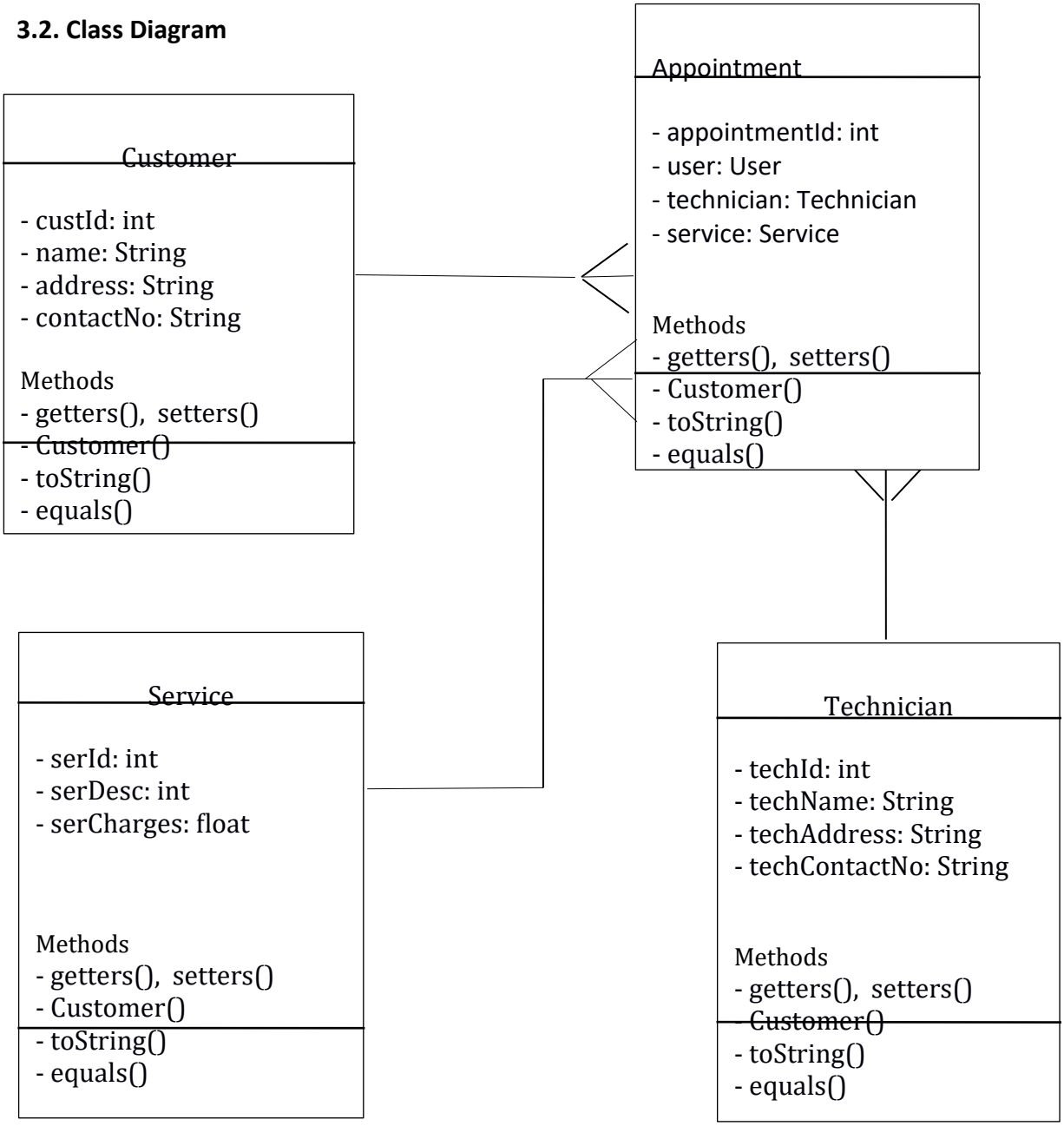
3. Viewing appointment history: Users should be able to view their appointment history.

3. Architecture

3.1. High-Level Architecture

The Online Two Wheeler Technician Appointment System will follow a stand-alone PC architecture. The client-side will consist of a user interface, while the attending side will handle the application logic, data processing, and database interactions.

3.2. Class Diagram



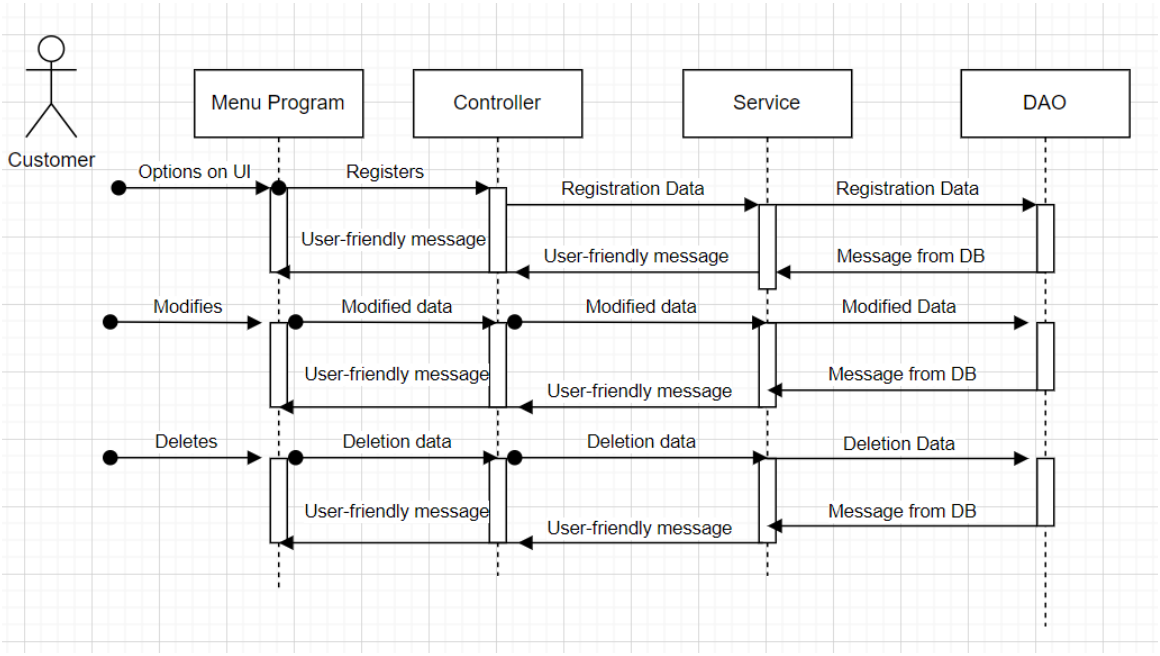
Other Specifications:

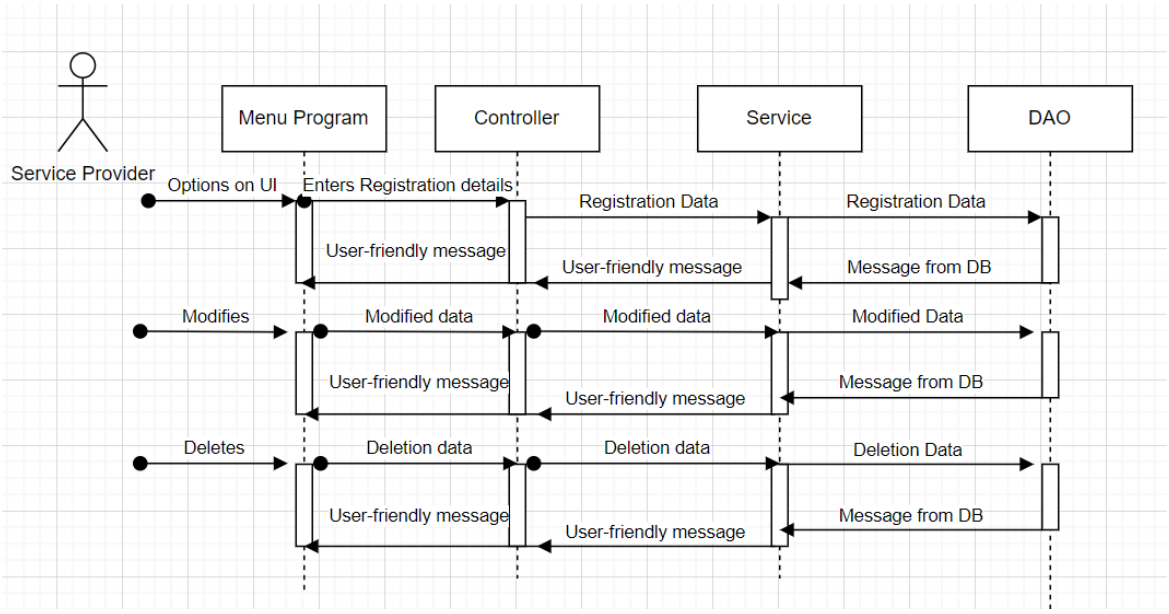
- Per entity one controller, service, DAO class expected.
- Insert, update, delete methods of DAO classes should return no. of records affected.
- Interfaces for service, DAO class expected.
- There should be layered architecture that follows MVC pattern.
- Common utility class for communicating to DB & closing connection mandatory.
- Assume appropriate method names, which should be descriptive, user-friendly.

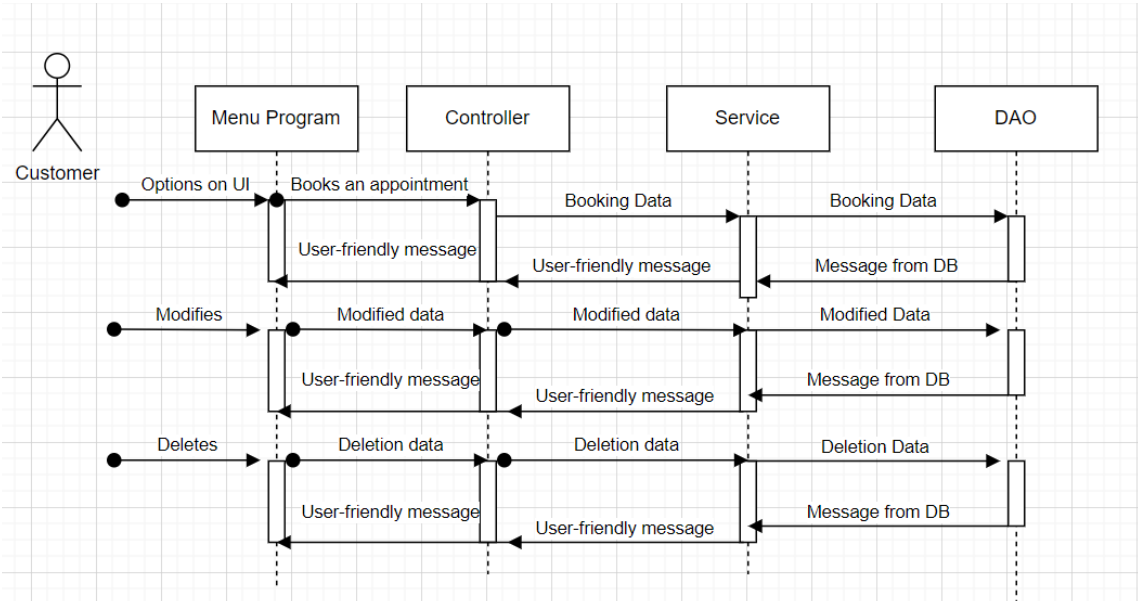
Assumptions:

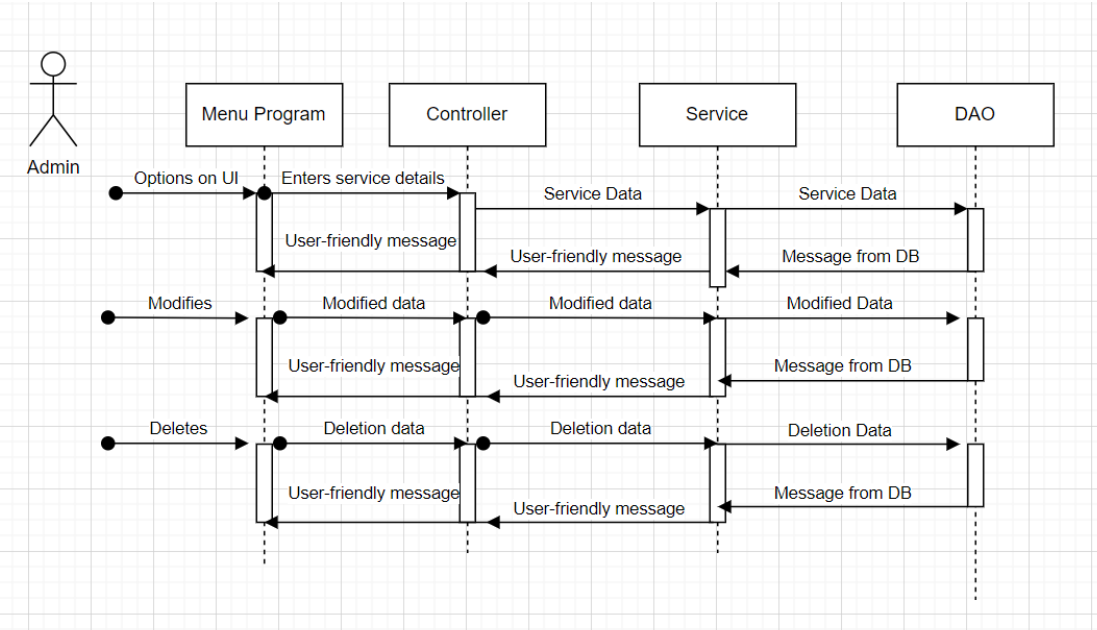
- All Two Wheeler technicians registered can provide all services.
- Only one technician per appointment.
- Only one service per appointment.

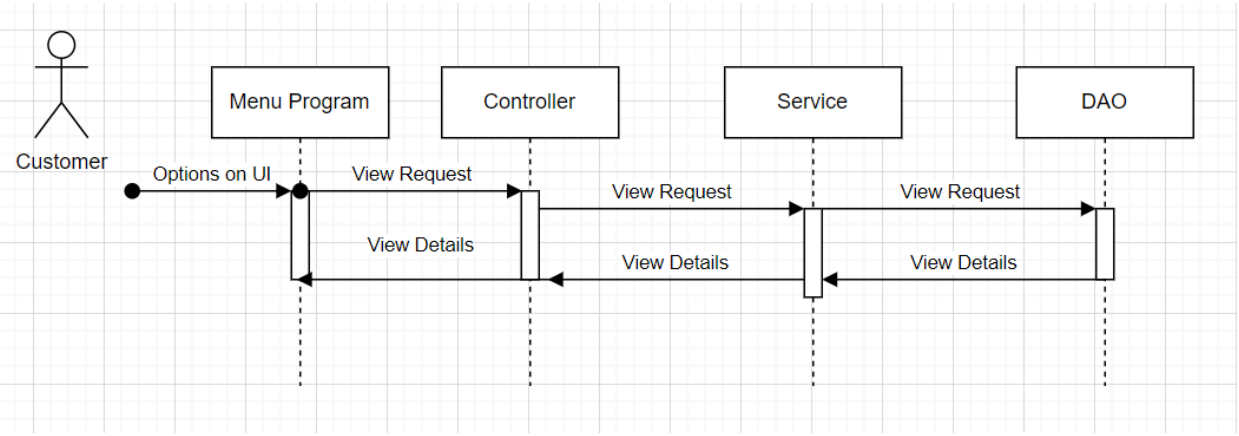
3.3. Sequence Diagrams




Service Provider
Menu Program
Controller
Service
DAO
Options on UI
Enters Registration details
Registration Data
Registration Data
User-friendly message
User-friendly message
Message from DB
Modifies
Modified data
Modified data
Modified Data
User-friendly message
User-friendly message
Message from DB
Deletes
Deletion data
Deletion data
Deletion Data
User-friendly message
User-friendly message
Message from DB


Customer
Menu Program
Controller
Service
DAO
Options on UI
Books an appointment
Booking Data
Booking Data
User-friendly message
User-friendly message
Message from DB
Modifies
Modified data
Modified data
Modified Data
User-friendly message
User-friendly message
Message from DB
Deletes
Deletion data
Deletion data
Deletion Data
User-friendly message
User-friendly message
Message from DB





4. User Interface

As this is a stand-alone PC program, user interface will be menu driven as follows:

1. Customer
2. Technician
3. Appointment
4. Service
0. Exit

Upon choosing 1:

1. Register Customer
2. Modify Customer details
3. Delete Customer record
4. View single record
5. View all records
0. Exit

Upon choosing 2:

1. Book a Technician
2. Modify Technician details
3. Delete Technician record
4. View single record
5. View all records
0. Exit

Upon choosing 3:

1. Book an appointment
2. Modify appointment details
3. Delete an appointment
4. View single record
5. View all records
0. Exit

Upon choosing 4:

1. Enter a repairing service
2. Modify repairing service details
3. Delete repairing service record
4. View single record
5. View all records
0. Exit

Accept required attributes, create its object & interact with DB.

This menu structure provides users with a logical flow, allowing them to navigate through different aspects of the system and perform actions related to customers, technicians, appointments, and services. The "Exit" option at each level enables users to gracefully exit the application when they're done. This setup seems well-organized and user-friendly for managing the different aspects of the appointment and service system.

5. Technologies Used

Following tools should be used:

- Jdk 1.8 or higher.
- Eclipse 2022-12 or higher
- Oracle (10g or higher).
- JDBC driver compatible with above.

6. Testing

Method	Input	Expected Output	Actual Output
Customer.insert()	1 Enter customer ID : 6 Enter name: bvk Enter address: pune Enter contact no: 1133225544	Record inserted successfully with customer ID 6 Record inserted successfully.	Record inserted successfully with customer ID 6 Record inserted successfully.
Customer.modify()	2 Enter customer ID: 6 Enter name: bvk Enter address: mumbai Enter contact no: 2211335544	Record updated successfully with customer ID 6 Record modified successfully.	Record updated successfully with customer ID 6 Record modified successfully.
Customer.delete()	3 Enter the Customer ID: 6	Record deleted successfully with the customer ID 6 Record successfully deleted.	Record deleted successfully with the customer ID 6 Record successfully deleted.
Customer.view()	4 Enter the Customer ID: 5	Customer [custId = 5, name=rupak, address=jaipur, contactNo=4466778855]	Customer [custId = 5, name=rupak, address=jaipur, contactNo=4466778855
Customer.viewAll()	5	List of Customers: [Customer [custId=2, name=yash, address=jaiput, contactNo=3366775544], Customer [custId=1, name=shreya, address=delhi, contactNo=6677990088], Customer [custId=3, name=pranav, address=Prayagraj, contactNo=667788990	List of Customers: [Customer [custId=2, name=yash, address=jaiput, contactNo=3366775544], Customer [custId=1, name=shreya, address=delhi, contactNo=6677990088], Customer [custId=3, name=pranav, address=Prayagraj, contactNo=667788990

		0], Customer [custId=soumyajit, address=West Bengal, contactNo=7799008866], Customer [custId = 5, name=rupak, address=jaipur, contactNo=4466778855]]	0], Customer [custId=soumyajit, address=West Bengal, contactNo=7799008866], Customer [custId = 5, name=rupak, address=jaipur, contactNo=4466778855]]
Service.insert()	1 Enter Service ID : 6 Enter Service Description: transmission check Enter Services Charges: 1500	Record inserted successfully with service ID 6 Record inserted successfully.	Record inserted successfully with service ID 6 Record inserted successfully.
Services.update	2 Enter Service ID : 6 Enter Service Description: transmission check Enter Services Charges: 2000	Record updated successfully with service ID 6 Record modified successfully.	Record updated successfully with service ID 6 Record modified successfully.
Services.delete()	3 Enter the Services ID: 6	Record deleted successfully with service ID 6 Record successfully deleted.	Record deleted successfully with service ID 6 Record successfully deleted.
Services.view()	4 Enter the Services ID: 2	Services [serId=2, serDesc=cluth repair, serCharges=200.0]	Services [serId=2, serDesc=cluth repair, serCharges=200.0]
Services.viewAll()	5	List of Services: [Service [serId=2, serDesc=cluth repair, serCharges=200.0], Service [serId=3, serDesc=break repair, serCharges=250.0],	List of Services: [Service [serId=2, serDesc=cluth repair, serCharges=200.0], Service [serId=3, serDesc=break repair, serCharges=250.0],

		Service [serId=4, serDesc=aligning, serCharges=100.0], Service [serId=5, serDesc=oiling, serCharges=150.0], Service [serId=1handle aligning, serDesc=, serCharges=1200.0]]	Service [serId=4, serDesc=aligning, serCharges=100.0], Service [serId=5, serDesc=oiling, serCharges=150.0], Service [serId=1handle aligning, serDesc=, serCharges=1200.0]]
Technician.insert()	1 Enter Technician ID: 6 Enter name: bvk Enter address: pune Enter contact no: 9977558866	Record inserted successfully with technician ID 6 Record inserted successfully.	Record inserted successfully with technician ID 6 Record inserted successfully.
Technician.update()	2 Enter Technician ID: 6 Enter name: bvk Enter address: mumbai Enter contact no: 9977558866	Record updated successfully with technician ID 6 Record updated successfully.	Record updated successfully with technician ID 6 Record updated successfully.
Technician.delete()	3 Enter the Technician ID: 6	Record deleted successfully with technician ID 6 Record deleted successfully.	Record deleted successfully with technician ID 6 Record deleted successfully.
Techinican.view()	4 Enter the Technician ID: 2	Technician [techId=2, techName=yash, techAddress=jaipur, techContactNo=8899000000]	Technician [techId=2, techName=yash, techAddress=jaipur, techContactNo=8899000000]
Technician.viewAll()	5	[Technician [techId=1, techName=Pranav, techAddress=Prayagraj, techContactNo=1627389917], Technician [techId=2,	[Technician [techId=1, techName=Pranav, techAddress=Prayagraj, techContactNo=1627389917], Technician [techId=2,

		techName=yash, techAddress=jaipur, techContactNo=1627389917], Technician [techId=3, techName=soumyajit, techAddress=delhi, techContactNo=2244668877], Technician [techId=4, techName=Shreya, techAddress=pune, techContactNo=1133557799], Technician [techId=5, techName=rupak, techAddress=goa, techContactNo=5577443311]]	techName=yash, techAddress=jaipur, techContactNo=1627389917], Technician [techId=3, techName=soumyajit, techAddress=delhi, techContactNo=2244668877], Technician [techId=4, techName=Shreya, techAddress=pune, techContactNo=1133557799], Technician [techId=5, techName=rupak, techAddress=goa, techContactNo=5577443311]]
Appointment.insert()	1 Enter Appointment ID: 6 Enter Customer ID: 2 Enter Technician ID: 4 Enter Service ID: 5	Record inserted successfully with appointment ID 6 Record inserted successfully.	Record inserted successfully with appointment ID 6 Record inserted successfully.
Appointment.update()	2 Enter Appointment ID: 6 Enter Customer ID: 1 Enter Technician ID: 1 Enter Service ID: 1	Record updated successfully with appointment ID 6 Record updated successfully.	Record updated successfully with appointment ID 6 Record updated successfully.
Appointment.delete()	3 Enter the appointment ID: 6	Record deleted successfully with appointment ID 6 Record deleted successfully.	Record deleted successfully with appointment ID 6 Record deleted successfully.
Appointment.view()	4 Enter the appointment ID: 3	Appointment [apptId=3, custId=3, techId=3, serId=3]	Appointment [apptId=3, custId=3, techId=3, serId=3]

Appointment.viewAll()	5	List of Appointments: [Appointment [apptId=3, custId=3, techId=3, serId=3], Appointment [apptId=2, custId=2, techId=2, serId=2], Appointment [apptId=4, custId=4, techId=4, serId=4], Appointment [apptId=5, custId=5, techId=5, serId=5]]	List of Appointments: [Appointment [apptId=3, custId=3, techId=3, serId=3], Appointment [apptId=2, custId=2, techId=2, serId=2], Appointment [apptId=4, custId=4, techId=4, serId=4], Appointment [apptId=5, custId=5, techId=5, serId=5]]

6. Conclusion

Throughout this project, we've looked at the important parts that make up the system. We've talked about how the system handles appointments and manages technicians. Customers can book appointments, and get updates. Technicians can see their jobs and tell customers when they'll be there. In the culmination of this project, we have successfully designed and conceptualized a console-based two-wheeler technician appointment booking system. This project is aimed to provide an insightful exploration into the of development of an efficient and user-friendly system for scheduling and managing appointments between customers and technicians.

7. References

https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/