

PROGRAMMING PROJECT

COURSES CODE & NAME

PROGRAMMING (K_BCS_004)

FACULTY

SCHOOL OF TECHNOLOGY AND ARC

STUDENTS NAME & MATRIC NUMBER

1.NUR ALEESA RIZAL (100007381)
2.CHARVI SHARMA (100006734)
3.DEV CHAUDHARY (100006200)
4.PALAKSHA BORA (100007676)
5.DAKSH PATEL (100007931)

LECTURERS' NAME

PROFESSOR PAUL
TANZER PROFESSOR
PETER DILLINGER



Parking Management System

Desktop Client + REST Backend

Java Swing • Spring Boot • MySQL • JPA/Hibernate



Project Overview

Search Feature

Admin searches for specific car plates using the search bar. Table filters to show only matching results after clicking Apply.

Filter & Sort

Filter by vehicle type (CAR, BIKE, TRUCK, OTHER). Sort by Exit Time, Entry Time newest/oldest, or Car Plate A-Z / Z-A.

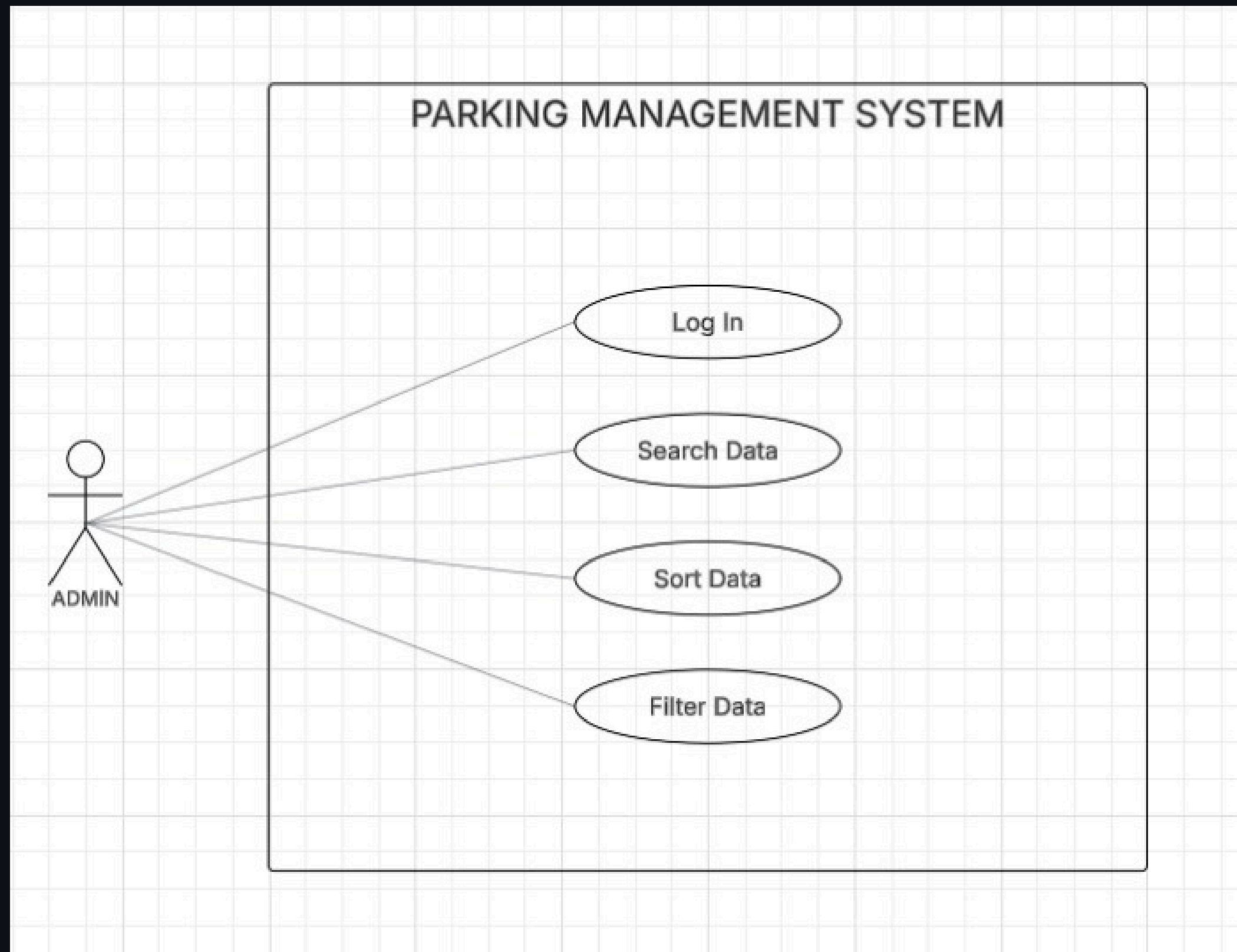
History Data View

Full table of all CLOSED tickets — ID, Plate, Type, Status, Entry Time, Exit Time, Amount. Shows the complete parking record.

Persistent MySQL Storage

All ticket data stored in MySQL database — survives restarts. Amount auto-calculated via a GENERATED column formula.

System Architecture — Use Case Diagram



Log In

Admin must authenticate before accessing the system.

Search Data

Admin searches for specific plate numbers in the table.

Sort Data

Admin sorts by Exit/Entry time or Car Plate A-Z.

Filter Data

Admin filters by vehicle type: CAR, BIKE, TRUCK, OTHER.

Tech Stack

Java + Swing

Programming Language & Desktop UI

Core language. Swing provides JFrame, JTable, JButton, ActionListeners, HttpClient, SwingWorker.

Spring Boot

REST Backend Framework

@RestController, @Service, @Transactional, Maven build tool.

IntelliJ IDEA

Development IDE

Main IDE used to develop the system. Used as backbone — allows development in Java with Spring Boot plugin.

MySQL Workbench

Database

Stores all ticket data. Allows adding, removing, editing and reading parking records.

GitHub — Version control. Holds all members' code contributions in one place.

MySQL — Tickets Table

Column	Type	Key Feature
id	BIGINT	PRIMARY KEY · AUTO_INCREMENT starts at 10001
plate	VARCHAR(20)	License plate — NOT NULL. Stored in uppercase.
vehicleType	VARCHAR(255)	CAR / BIKE / TRUCK / OTHER — stored as String via @Enumerated
status	VARCHAR(225)	OPEN when parked · CLOSED when exited
entryTime	DATETIME	DEFAULT CURRENT_TIMESTAMP — auto set on insert
exitTime	DATETIME	DEFAULT NULL · Set when car exits via manualExit()
amount	DECIMAL(10,2)	GENERATED: ceiling(minutes ÷ 60) — read-only, insertable=false
ticketID	VARCHAR(225)	Customer-facing ID e.g. TKT-10001, set by @PrePersist

Spring Boot Backend — Layer Architecture

Controller Layer

`@RestController`

1

TicketApiController — Receives HTTP GET requests to /api/tickets, builds dynamic Specification, returns JSON list of tickets

Service Layer

`@Service`

2

TicketService — manualEntry() checks if plate already has OPEN ticket. manualExit() finds OPEN ticket and closes it. Both use @Transactional

Repository Layer

`@Repository`

3

TicketRepository extends JpaRepository + JpaSpecificationExecutor. Spring auto-generates SQL — no manual SQL written

Entity Layer

`@Entity`

4

Ticket.java maps to MySQL 'tickets' table. @PrePersist lifecycle hook auto-sets ticketID and default status = OPEN before INSERT

Spring Boot — REST API Endpoints

POST

/api/tickets/entry

Create new ticket — car enters parking

```
{ "plate": "ABC123", "vehicleType": "CAR" }
```

POST

/api/tickets/exit

Close ticket — car exits, fee auto-calculated by MySQL

```
{ "plate": "ABC123" }
```

GET

/api/tickets?status=OPEN

Get all currently parked vehicles (Live tab)

```
Returns: [ { id, plate, status, entryTime... } ]
```

GET

/api/tickets?status=CLOSED&vehicleType=CAR&sort=exitTime,desc

History with filters + sort (used by ParkingHistoryUI)

```
Returns: [ { id, plate, exitTime, amount... } ]
```

GET

/health

Server health check — used by Ping button in UI

```
Returns: { "status": "UP" }
```


GUI Highlights — Actual System Screenshots

Management System

Plate

Plate contains... (e.g., MH12)

Status

CLOSED

Type

ALL

Sort

Exit time (newest)

Apply

Clear

	Plate	Type	Status	Entry Time	Exit Time	Am
	B OF 4176	CAR	CLOSED	2026-02-16T00:12:05	2026-02-16T12:12:05	12.0
	M DM 1033	TRUCK	CLOSED	2026-02-15T23:56:05	2026-02-16T11:56:05	12.0
	HH AI 5329	CAR	CLOSED	2026-02-15T23:55:05	2026-02-16T11:55:05	12.0
	HH ZX 5634	TRUCK	CLOSED	2026-02-15T23:41:05	2026-02-16T11:41:05	12.0
	F AQ 634	BIKE	CLOSED	2026-02-15T23:17:05	2026-02-16T11:17:05	12.0
	B FS 9057	TRUCK	CLOSED	2026-02-16T00:05:05	2026-02-16T11:05:05	11.0
	HH WM 7139	BIKE	CLOSED	2026-02-16T00:05:05	2026-02-16T11:05:05	11.0
	S BA 8687	CAR	CLOSED	2026-02-15T22:55:05	2026-02-16T10:55:05	12.0
	B TF 8452	CAR	CLOSED	2026-02-15T22:31:05	2026-02-16T10:31:05	12.0
	F IT 7813	CAR	CLOSED	2026-02-15T22:27:05	2026-02-16T10:27:05	12.0
	F VP 5876	CAR	CLOSED	2026-02-15T23:23:05	2026-02-16T10:23:05	11.0
	M WE 3383	BIKE	CLOSED	2026-02-16T00:16:05	2026-02-16T10:16:05	10.0
	M JU 9572	TRUCK	CLOSED	2026-02-15T22:13:05	2026-02-16T10:13:05	12.0
	S QV 2866	BIKE	CLOSED	2026-02-15T22:11:05	2026-02-16T10:11:05	12.0
	F SX 3630	TRUCK	CLOSED	2026-02-15T23:10:05	2026-02-16T10:10:05	11.0
	HH GH 7382	CAR	CLOSED	2026-02-15T23:10:05	2026-02-16T10:10:05	11.0

Management System

Plate

B OF 4176

Status

CLOSED

Type

ALL

Sort

Exit time (newest)

Apply

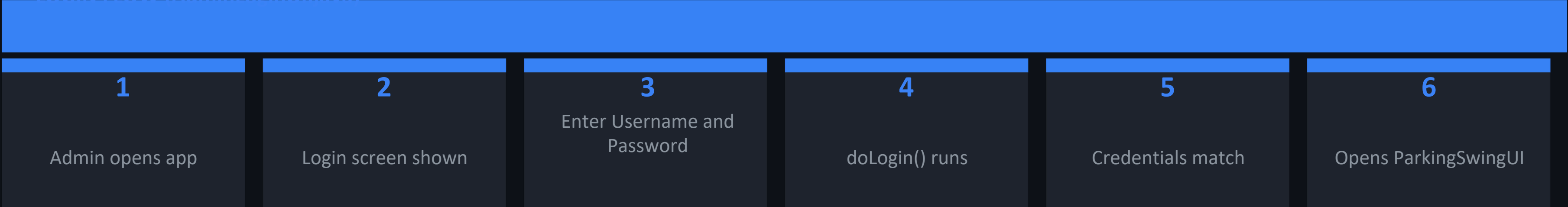
Clear

	Plate	Type	Status	Entry Time	Exit Time	A
	B OF 4176	CAR	CLOSED	2026-02-16T00:12:05	2026-02-16T12:12:05	12.0

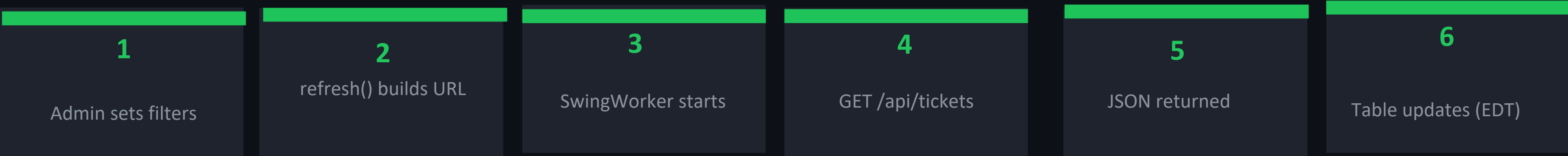
History section — columns: ID · Plate · Type · Status · Entry Time · Exit Time · Amount | Filter: Status + Type dropdowns | Sort: dropdown

End-to-End Flow — Login to History View

LOGIN FLOW (AdminLoginUI.java)

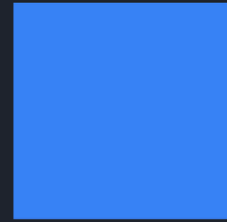


SEARCH / FILTER / SORT FLOW (ParkingHistoryUI.java)



OOP Concepts in This Project

Encapsulation



All fields in Ticket.java are private — accessed only through getters/setters. The backend hides the MySQL database entirely from the Swing frontend.

Inheritance



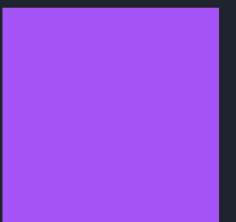
AdminLoginUI extends JFrame (login window). ParkingSwingUI extends JFrame (dashboard). TicketRepository extends JpaRepository — inherits all CRUD methods.

Polymorphism



VehicleType enum (CAR, BIKE, TRUCK, OTHER) — each is a distinct polymorphic value. ThrowingSupplier functional interface overrides the standard Supplier behavior.

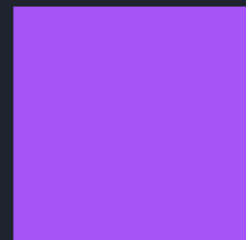
Abstraction



JpaRepository hides all SQL. HttpClient hides HTTP socket details. runAsync() hides threading complexity. Each layer exposes only what is needed.

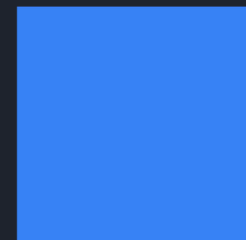
Thank You

Parking Management System



Swing Client

Java 11 Desktop UI



Spring Boot

REST API Backend



MySQL + JPA

Persistent Storage



REST / JSON

Client-Server Protocol