FINAL PROJECT - 25 Winter CST8288

ProjectOverview

The **Public Transit Fleet Management System (PTFMS)** is a web-based application designed for public transit agencies to monitor, track, and optimize operations of buses, electric light rail, and diesel-electric trains. The system integrates real-time GPS tracking, fuel/energy monitoring, predictive maintenance alerts, and route optimization to improve transit efficiency and reliability.

Design & Technology

This project is to be implement using a **3-tier architecture** (i.e. layers: *Presentation*, *Business* and *Data*). In addition, it must incorporate the following **GoF Design Patterns** and software patterns:

- DAO,
- Builder,
- plus an additional 4 selected from:
  - Strategy,
  - Simple Factory,
  - Adapter,
  - Observer, and
  - Command.

The project must implement a "controller class" to manage communication and processing between the *Presentation*, *Business*, and *Data* management layers.

The project must be implemented using:

- Java 21 and Java Servlets
- MySQL 8.0.40
- Apache Tomcat 9.0.90 (or later)

All *Business* layer requirements must be implemented using Java Servlets (not JavaScript)
Code must be delivered in the form of a NetBeans Maven project exported as `.zip` file.

Functional Requirements:

FR – 01: User Registration & Authentication:

- Transit Managers and Operators can register by providing name, email, password, and user type (Manager or Operator). Note: A Transit Manager is also an Operator.
- The system includes login/logout functionality with role-based access control.

FR – 02: Vehicle Management

- Transit Managers can register vehicles with details like:
  - Vehicle Type (Diesel Bus, Electric Light Rail, Diesel-Electric Train)
  - Vehicle Number (Unique identifier)
  - Fuel/Energy Type & Consumption Rate
  - Maximum # of passengers
  - Current Assigned Route

FR – 03: GPS Tracking

- Vehicles transmit real-time location to the system.
- Transit managers can view a report listing each vehicle along with its arrival and departure time from each transit station.
- Operators manually log times when they are taking a break or otherwise out-of-service.

FR-04: Monitoring Energy/Fuel Consumption

- Fuel/Energy usage reports for each vehicle type:
  - Buses: Diesel or CNG fuel efficiency tracking
  - Electric Light Rail: Energy consumption per mile
  - Diesel-Electric Train: Fuel usage tracking
- Transit Mopeaanagers get alerts for excessive fuel/energy consumption.

FR- 05: Alerts for Predictive Maintenance

- The system monitors:
  - Number of hours of use (wear) of key components (e.g. brakes, wheels/tires and axle bearings)
  - for electric light rail, key electrical components (e.g. catenary, pantograph, circuit breakers)
  - Engine diagnostics (for buses & trains)
- Maintenance alerts are triggered when vehicles require servicing.
- Transit Managers can schedule maintenance tasks.

FR-06: Reporting & Analytics

- Transit maintenance dashboards.
- Operator performance dashboards (on-time arrival rates, efficiency).
- Cost reports on fuel/energy usage and maintenance expenses.

Deliverables

The deliverables are divided into 2 parts:

Part 1 - High-Level Design Checkpoint

This is an interim deliverable that documents the High-Level Design. Its purpose is to provide an overview of the solution and to guide further refinement of the solution.
The High Level Design must include as a minimum:

- Application Architecture
- Business Architecture
  - Including UML Use Case Diagram(s) with text descriptions,
- Detailed Design
  - Including UML Class diagrams, Component diagrams
- Data Architecture
  - Including ERD
- Testing Model
  - Description of test plan

A sample table of contents will be provided.

Part 2 – Final Project Deliverable

As minimum, the Final Project deliverables must be uploaded to BrightSpace:

- Peer Review document
  - A template will be provided.
- Final revision of the High Level Design document
- Database
  - Final version of the ERD
  - You are required to come up with a sample data based on your database structure in order to support your demo and for your testing activities
  - .sql file that creates the database schema, its tables and populates the tables with data.
  - .properties with the MySQL connection string along with userid: *cst8288* and password: *cst8288*

FINAL PROJECT - 25 Winter CST8288

- Final version of UML Diagrams
    - List of all Design Patterns and software patterns used
        - Each pattern must be documented with an UML class diagram
- A table listing each file name organized by:
    - Presentation Layer
    - Business Layer
    - Data Layer
- Use Case Diagram complete with text descriptions.
- Sequence Diagram describing key elements of your solution
- HTML, Java code and Junit tests
- Code for all classes, interfaces, HTML
    - To be delivered as a NetBeans Maven project that has been exported as a `.zip`
    - The name of the `.zip` file must adhere to the naming convention: **Group#_Final_Project.zip**
    - The NetBeans project name must adhere to the convention: **Group#_Final_Project**
- All Java code must include JavaDoc comments for all **public** elements.
    - Be sure to include a separate **@author** tag for each Group member who contributed to that code.
- `.zip` export of your Group GitHub repo
    - See **Additional Instructions** below for more details on the use of GitHub

ALGONQUIN COLLEGE ©

FINAL PROJECT - 25 Winter CST8288

Additional Instructions

Demo

- In-class demonstration of the solution is mandatory.
- **Group members absent from the demo will be given an automatic 0 for the Final Project. No demo means 0 marks for all Group members.**
- You are required to create a PowerPoint presentation (Max 4/5 slides) and present it during the demo.
- A sample PowerPoint will be provided.

Use of GitHub

- The use of **GitHub** is mandatory for the project.
- Delegate one group member to take the responsibility of *Team Lead*. The *Team Lead* will create the **GitHub** repository.
  - It must be a **private** GitHub repository and only accessible by Group members and the Lab Professor (see below).
- Each team member will create their own branch from the main repository in order to implement the feature for which they are responsible.
- Code should be submitted to the main repository using Pull Requests that the Team Lead will be responsible for approving or rejecting.
- All team members should be creating pull requests and sharing work.
- If team members are not contributing work equally (as indicated by their GitHub activity) it may be reflected in their individual final mark for the Group project.
- **Each Group must provide their Lab Professor access to their GitHub. If they do not, all Group members will receive an automatic zero.**
- Your Lab Professor will provide you with their GitHub id (e.g. via BrightSpace).

Deadlines:

- The deadline for submitting the High Level Design Checkpoint will be published on BrightSpace.
- The deadline for submitting the Final Project deliverables will also be published on BrightSpace.
- No extensions for either will be permitted.
- Demo sessions for each group will take place during your regularly scheduled Lab period in Week14.

Project Weight and Marks Distributions:

The project contributes **20 marks** to your Final Grade.

- High-Level Design – **5 marks**
- Solution – **15 marks.**