Low Level Design (LLD)

E-Bus Management Based Current Location System

Revision Number: 2
Last date of revision: 06/09/2023

Abutalha D Maniyar

Document Version Control

Date Issued	Version	Description	Author
05th September 2023	1.1	First Draft	Abutalha Maniyar
06th September 2023	1.2	Second Draft	Abutalha Maniyar

Contents

Abstract

Advanced traveller information systems are one component of intelligent transportation systems, and a major component is travelling time information. It is a very important subject to improve the precision and reliability of the bus management model which can attract additional ridership, reduce passenger's anxiety and waiting times at bus stop, and increase their satisfaction. Furthermore, it can promote the development of city public transportation. This paper presents an improved approach to predict the public bus arrival time based on location. After analysing the components of bus arrival time systematically, the bus arrival time and dwell time at previous stops are chosen as the main input variables of the prediction model. At first, the algorithm of data interpolation and processing is designed to get the last know time of bus arrival as the input variables of the prediction models.

1 Introduction

1.1 Why this Low-Level Design Document?

The purpose of this document is to present a detailed description of the E-Bus Management Based Current Location System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system and will be proposed to the higher management for its approval.

The main objective of the project is to create a system that enables effective management of the buses and to provide information of the bus, In order to help both the travel agencies and travellers.

The are certain problems with existing system:

- There is no system based on the current system that provides information on buses and their expected arrival time and expected waiting time.
- Currently the management of public transport buses is the main problem.
- There is no generic platform for private agencies to manage their service online.
- A common platform for all users to get bus information.

1.2 Scope

- This software system will be a Web application, this system will be designed to overcome the drawbacks of the present system and to make the travel experience better and easy.
- It aims to provide comprehensive bus information, including expected arrival times and waiting times, while also facilitating efficient management of public transport buses.
- Additionally, it will serve as a platform for private agencies to manage their services online and offer a common platform for all users to access bus-related information.

1.3 Constraints

- Security measures will be in place to protect user data and system integrity.
- Compliance with relevant regulations and standards in the public transportation sector.

1.4 Risks

Document specific risks that have been identified or that should be considered.

- Potential risks, such as technical challenges and data accuracy issues, will be identified and addressed through risk management strategies defined in the project plan.

1.5 Out of Scope

Delineate specific activities, capabilities, and items that are out of scope for the project.

- Ticketing and fare collection systems are not within the scope of this project.
- Physical bus infrastructure, such as bus stops and shelters, is not included.

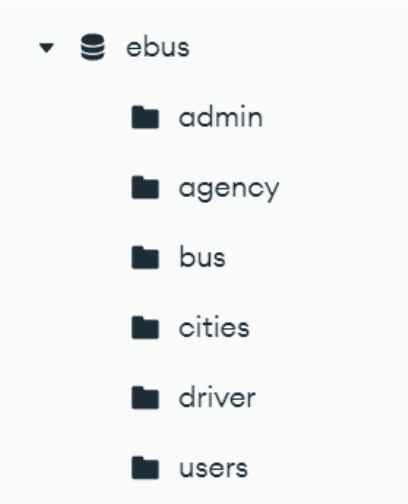
2 Technical specifications

2.1 Web-based specs

- Client-side technologies: HTML5, CSS, Bootstrap
- Server-side technologies: Flask, Python
- Database: MongoDB for storing bus and user data.

2.1.1 Database overview

The application uses ebus database and it consists of 6 different collections.



2.2 Logging

We should be able to log every activity done by the user.

- The System identifies at what step logging required
- The System should be able to log each and every system flow.
- Developers can choose logging methods. You can choose database logging/ File logging as well.
- System should not be hung even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.

2.3 Database

System needs to store every request into the database and we need to store it in such a way that it is easy to retrain the model as well.

- MongoDB database design to store bus-related data efficiently.
- Indexing and query optimization for fast data retrieval.

2.5Deployment

- Deployment on cloud infrastructure (i.e., AWS) for scalability and availability.
- Continuous integration and continuous deployment (CI/CD) pipelines for automated updates.

AWS

EBS

CODEPIPELINE

3 Technology stack

Front End	HTML/CSS/Bootstrap
Backend	Flask/Python
Database	MongoDB
Deployment	AWS

4 Proposed Solution

The Ebus Management Based Location System aims to transform the public transportation sector by providing passengers with a seamless and efficient bus service experience while enabling public transport authorities and private agencies to manage and optimize their services effectively. This proposed solution outlines the key features, technical aspects, and benefits of BIMS, paving the way for a more convenient and reliable transportation system.

4.1 Key Features

4.1.1 Passenger Features:

- User Accounts: Passengers can create accounts, save favourite routes, and receive personalized alerts.
- Expected Waiting Times: Passengers can check the expected waiting times at bus stops, making it easier to plan their journeys.
- Route Information: Access to comprehensive route details, including stops and schedules.

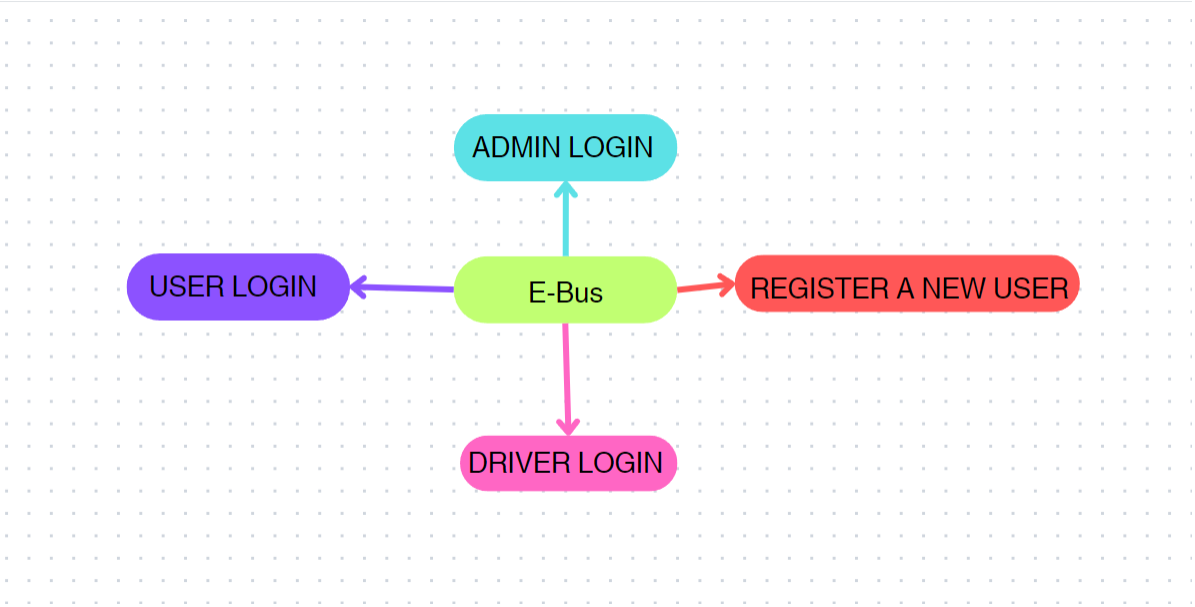
4.1.2 Administrator Features:

- Bus Management: Tools to monitor buses in real-time, track performance, and optimize deployment.
- Scheduling and Dispatching: Streamline bus schedules and dispatching for efficient service.
- Maintenance Tracking: Schedule and log maintenance tasks for each bus to ensure safety and reliability.

4.1.3 Private Agency Features:

- Service Management: Private agencies can manage routes, schedules, and bus information.
- Collaboration: Seamless integration with the ebus ecosystem for a unified bus service experience.

5 Model



There are four end points

- User login
- User registration
- Driver login
- Admin Login

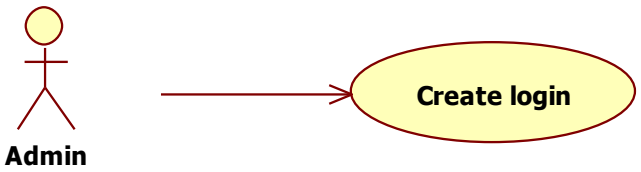
All the login pages will have input field. Respective IDs and passwords to give further access to the respective person. Once the access is given the person will be able to execute all the respective function at the interface. Using the OOPS concept user module, driver module and admin module must be implemented.

And the registration page will take first name, last name, mail id and password as input. By providing the information a new user will be able to create his/her account.

6 User I/O workflow

Admin Module:

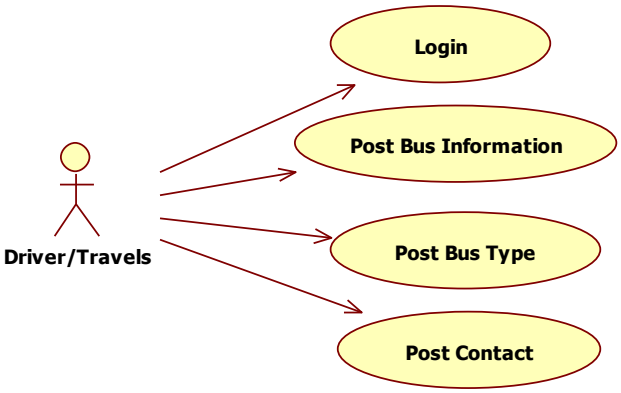
- Admin can Log in using his admin id and password
- Admin can create a driver



- A new admin can be added to ebus with help of developer

Driver Module:

- Can log in using driver id and password provided by admin/agency
- Can post bus information like source, destination
- Can post bus type
- Can Post his contact along with the bus info



Below is the sample of the bus info posted by the driver


```
_id: ObjectId('64f699d6b837938fdf320fc2')
driver id: "Mohan"
▾ info: Array (4)
    0: "Achalpur"
    1: "Achhnera"
    2: "Every Day"
    3: "09:30"
type: "General"
contact: "6366288222"
```


User Module:

- User can log in with registered email id
- Can search bus based on location
- Can view bus information and contact information
- User has read only access

7 Test cases

Test case	Steps to perform test case	Module	Pass/Fail
01	Hit the URL	Application module, index function is executed because '/' URL is GET method and it return landing page of application	Pass
02	Click on Button Register as User	Application module, register function decorated with route	Pass
03	Click Button User Login (No need to do this if you run test case 2, it will automatically redirect you)	User module, user object creation and calling login method of user	Pass
04	Logged in as user Select city names and search bus	User module, search bus function	Pass
05	(Go to Landing Page) and click Admin Login	Application module a get method will be executed which return an Admin login page	Pass
06	Enter admin details (id = admin) (password = 123123)	Admin module, admin object creation and calling login method on admin	Pass
07	Create driver	Admin module, createdriver method	Pass
08	Click logout	Application module, index function will return landing page	Pass
09	Driver Login Page	Application module a get method will be executed which return a Driver login page	Pass
10	Driver Login	Driver module, login method is executed	Pass
11	Post bus info	Driver module, postbusinfo method is executed	Pass