FLEET OF TAXI SERVICE MANAGEMENT SYSTEM - LABORATORY EXERCISE 2: SOFTWARE REQUIREMENTS SPECIFICATION

Danielle Winter (563795), Frederick Nieuwoudt (386372), Stephen Friedman (360938) & Sello Molele (0604606X)

School of Electrical & Information Engineering, University of the Witwatersrand, Private Bag 3, 2050, Johannesburg, South Africa

1. Introduction

A software database system for a Taxi Fleet Management is proposed in this Software Requirements Specification. The objective of this document is to describe in detail the user requirements as well as the specifics of what the system requires and how it will be developed.

The aim of the system is to respond to a customer request for transport to a location. It will then find an available driver for the job and deploy their taxi for service. It will also allow for users to manage profiles and administrators to manage diver profiles.

2. User Requirements

A brief description of the different users and their basic requirements is given. Furthermore, potential users were surveyed for additional requirements. The user requirements are specified based on the MOSCOW and FURPS methodologies [1].

2.1 Basic Customer Requirements

- Passengers
  - Registration
  - Account Maintenance
  - Lift Requests
- Drivers
  - Receive Fare
  - Complete Fare
- Administrators
  - Add drivers
  - Maintain driver details
  - View logs

2.2 Detailed User Requirements

Passenger Registration

- Passenger login details set up
- Enter personal details (name, age, gender)
- Enter contact details (email address, cellphone number)
- Enter banking details (Credit Card number and type, expiry date of card, CVV number)

Passenger Account Maintenance

- Ability to update details (Change in name or contact details and updated payment details)
- Account cancellation (optional)

Passenger Lift Requests

- Log a destination
- Log a current location
- Specify number of passengers to be transported
- Passenger payment confirmation
- Order a lift in advance (e.g. collection from the airport at a specified time)
- Passenger feedback (ride comfort, satisfaction)
- Job update (passenger is able to say if they have been collected)

Driver Fare Received

- Driver receives an order for a nearby customer
- Driver accepts order
- Passenger updated about taxi arrival time
- Driver status changed to unavailable

Driver Fare Completed

- Driver reports drop-off
- Driver status changed to free

Administrator Driver Addition

- Driver personal details added to database (Name, age, Identity number, gender, medical aid details, tax number)
- Driver contact details (residential and postal address, email address, cellphone number)
- Driver vehicle details (number plate, car model, colour and insurance details)

Administrator Driver Maintenance

- Update driver details for personal, contact and vehicle categories

Administrator Log Viewing

- View current job data
- View historical job data

3. System Specifications

3.1 Scope

The taxi fleet management system is a tool that provides a service where a user can request a taxi and the system will dispatch the closest available driver.

The system back-end will be required to store information about drivers, customers, and the current pricing scheme. The system will keep historical data regarding trips that have been taken for auditing and dispute resolution.

The system front end will be presented in the form of a web interface. This will provide a means for customers to input their current location and destination. The web interface will also provide a quoted price and allow for the customer to either accept or reject the quote.

3.2 Testing Plan

To achieve a quality product the test driven development(TDD) paradigm will be adopted. TDD calls for the writing of tests before writing the code to be used in the implementation. This encourages simple modular code.

The TDD workflow follows the sequence of writing a test, validating that the new test fails, writing the application code, running all the tests together, and then refactoring. This sequence allows for the project to move from getting code working to refactoring the code into simple understandable modules.

3.3 Implementation Plan

Following the Scrum SDLC methodology, will provide shippable incrementally improved products. A minimum of required features will be needed for a product that can meet the project scope as defined above. Additional functionality can be incrementally provided over the life span of the product.

3.4 Implementation Plan

Following the Scrum SDLC methodology, will provide shippable incrementally improved products. Though a minimum of required features will be needed for a product that can meet the project scope. Additional functionality can be incrementally provided over the life span of the product.

4. Scrum Product Backlog

4.1 Product Owner List of Needs

- Passenger Registration
- Passenger Account Maintenance
- Passenger lift requests
- Driver fare received
- Driver fare completed
- Administrator Driver Addition
- Administrator Driver Maintenance
- Administrator Driver Log Viewing

4.2 Sprint 1

Basic Website that can allow few passengers to register and to request for the available driver, but the passengers/ customers do not see or have access to the list of drivers in this application

- Passenger Registration
- Passenger lift requests
- Driver fare received
- Driver fare completed
- Administrator Driver Addition

4.3 Sprint 2

Additional features to the website that allows the drivers and passengers to have an ability of updating their personal details

- Passenger Account Maintenance
- Administrator Driver Maintenance

4.4 Sprint 3

Additional feature which is the driver log view storage which kept details of all the drivers in the system

- Administrator Driver Log Viewing
- Software Testing of the product.

4.5 Sprint Timelines

- Sprint 1 has a timeline of 2 weeks
- Sprints 2 and 3 have a timeline of 1 week
- Sprint 4 has a timeline of 1 week

4.6 Scrum Roles

- Product Owner
  - Gave the Taxi Fleet Management Specifications and He/ She will continue to update the requirements and interact with the customers and the scrum team.
- Project Developer
  - Implement the specified website with the scrum team.
- Scrum Master
  - Make sure that the team have meetings atleast twice or three times a week in order to woek on the project and ensures they work in a conducive enviroment and that they also give feedback once the work is done. The on time software tool may be used for creating clear time lines for the project. She/ he also is responsible for drawing the burn down graph ones the team began to work to check their progress every week.
- Scrum Team
  - Developing the website, testing it and also improving the website as soon as there are special needs and specifications that the product owner needs.

4.7 Product Release Date

- After 4 weeks

At this stage the scrum retrospective stage may take place and the team will have a scrum review meeting with the product owner and hear of the improvements that could be made, after the new list of sprints have been developed, the team can improve the second product or service and release the second product.

REFERENCES

[1] R. Stephens. *Beginning Software Engineering*. John Wiley and Sons, Inc, 2015.