**Software Requirements Specification**

**for**

**<Project>**

**Version 1.0 approved**

**Prepared by <author>**

**<organization>**

**<date created>**

**Table of Contents**

**Table of Contents**

**Revision History**

**1. Introduction**

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Product Scope

1.5 References

**2. Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

**3. External Interface Requirements**

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

**4. System Features**

4.1 System Feature 1

4.2 System Feature 2 (and so on)

**5. Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

**6. Other Requirements**

**Appendix A: Glossary**

**Appendix B: Analysis Models**

**Appendix C: To Be Determined List**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

1. **Introduction**
   1. **Purpose**

*<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>*

This document represents the Requirement Analysis and Specification Document (RASD) for the software system myTaxiService, and the main goal of this document is to completely describe the system in terms of functional and non-functional requirements, analyse the needs of the customer, and represent the constraints and the limitations of the software. Also typical use cases and scenarios will be presented.

This document is intended to be used by developers, programmer and testers, who will implement and test the system, system analysts and requirement analysts for inter related systems, project managers, customers and users of the system.

* 1. **Scope**

*<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>*

Software system myTaxiService is a project meant to optimize the taxi services of the city, by providing the mobile and web application that will allow users to request a taxi, or book the ride in advance and cancel the ride. It will also provide a fair management of taxi queues for the taxi drivers, and maximize the profit for the city.

In particular, city is divided into taxi zones, and each zone is assigned to a taxi queue (the system automatically calculates the distribution of taxis in zones based on the GPS information it receives from taxis). If the taxi is available (this information is provided by the taxi driver, who informs the system about his/hers availability through a mobile application), its identifier is stored in a queue of taxis in the corresponding zone. When a user requests a taxi from a zone (through a mobile or web application), the system forwards it to the first taxi queuing in that zone. Taxi driver can then confirm or deny the request through the mobile application. If the request is denied system will forward the request to a next taxi in the queue (and move the first taxi to the end of the queue), or if the queue is empty it will make sure that another taxi gets a request. When the taxi driver confirms the request, the passenger will be informed by the system about the taxi identifier and the waiting time. The passenger can also book a taxi, specifying the starting point and the destination, and the system then allocates a taxi 10 minutes before the meeting time with the passenger.

The system will also provide programmatic interface to enable development of additional services on top of the basic one.

1.3 **Actors**

* Visitor - A person that is not registered for the service, or a registered user who is not logged in to the application. Visitor can only see the home page, registration from and log in form.

* Registered users - A person that has an account for myTaxiService.

* User - A logged in registered user.

* Passenger - A user that uses myTaxiService for the means of requesting and reserving a taxi ride.

* Driver - A user of the mobile application, that drives a taxi, and uses the application for receiving the requests for the ride.

* Administrator - Supervisor, manages databases, and also takes care of drivers responsibility.

1.5 **Goals**

List of the goals for myTaxiService:

* [G1] Allow users to access myTaxiService through web or mobile application.

* [G2] Allow visitors to register to the application.

* [G3] Allow registered users to log in to the application, with same credentials for both the web and the mobile application.

* [G4] Passenger can request a taxi ride by specifying his starting point and his destination.

* + [G4.1] When requesting a taxi ride, passenger can specify either his starting point or destination or both by selecting the points on the map for the locations.

* + [G4.2] When requesting a taxi ride, passenger can specify either his starting point or destination or both by typing in the address of the locations.

* + [G4.3] When requesting a taxi ride, passenger can specify his starting point by revealing his/hers GPS location.

* + [G4.4] The system will notify the passenger on request about the taxi code, waiting time and the fare amount.

* + [G4.5] The passenger can either accept or cancel the taxi proposal.

* + [G4.6] The passenger has to pay for the accepted ride.

* [G5] Passenger can reserve a taxi ride specifying his starting point, his destination and the desired time, at least two hours before the ride

* + [G5.1] When reserving a taxi ride, passenger can specify either his starting point or destination or both by selecting the points on the map for the locations.

* + [G5.2] When requesting a taxi ride, passenger can specify either his starting point or destination or both by typing in the address of the locations.

* [G6] When the passenger requested a taxi ride, he will be informed with a code of the incoming taxi and the waiting time.

* [G7] When the passenger reserves a taxi ride, he will be informed with a code of the incoming taxi 10 minutes before the specified time of the ride.

* [G8] Passengers can pay for the ride through the application.

* [G9] Driver can inform the machine that he is available for the rides.

* [G10] Available driver can inform the machine that he is no longer available for the rides.

* [G11] Available driver can confirm or reject the certain request or reservation for the ride.

1.4 **Definitions, Acronyms, Abbreviations**

1.4.1 **Definitions**

* Available driver - is a driver who informed the system that he is available for the rides.

* Proposed taxi - is a taxi, that is not yet allocated by the system, but accepted the request.

1.4.2 **Acronyms**

* RASD - Requirements Analysis and Specification Document

* DBMS - Data Base management system

* DB - Data Base

* API - Application Programming Interface

* OS - Operating System

1.4.3 **Abbreviations**

* [Gn] - n-th Goal

* [Rn] - n-th Functional Requirement

* [Dn] - n-th Domain Assumption

* [Mn] - n-th Mockup
  1. **References**

*<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>*

* Specification Document: myTaxiService Project, AA 2015-2016

.

* IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

1.6 **Document Overview**

This document is structured in four parts:

* Section 1 - Introduction. It gives a description of document and some basic information about software.
* Section 2 - Overall Description. Gives general information about the software product with more focus about constraints and assumptions.
* Section 3 - Specific Requirements. This part lists requirements, typical scenarios and use cases. To give an easy way to understand all functionality of this software, in this section UML diagrams are given.
* Section 4 - Appendix. This part contains some information about the attached .als file and some described screenshot of software used to generate it.

1. **Overall Description**
   1. **Product Perspective**

*<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>*

The system myTaxiService is a project being developed for the government of city X. It will be released as a new, self-contained product, not integrated with other existing systems. Users will be able to access the service through a web and mobile application. The application will have a internal interface for administration, and it will provide API to enable the development of additional services on top of the basic one.

* 1. **Product Functions**

*<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>*

* 1. **User Classes and Characteristics**

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

* 1. **Operating Environment**

*<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>*

* 1. **Design and Implementation Constraints**

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>*

2.5.1 **Regulatory Policies**

* Privacy Policy - It fulfills a legal requirement to protect a customer or client's privacy.
* Online Payment Policy - satisfies a safe online transactions.

2.5.2 **Hardware Limitations**

myTaxiService doesn’t have any hardware limitations.

2.5.3 **Interfaces to other applications**

* Online transactions -
* Maps -

2.5.4 **Parallel operations**

myTaxiService must support parallel operations from different users when working with database and with all operation done by the user after connection.

2.5.5 **Documents related**

* Requirements and Analysis Specification Document (RASD).
* Design Document (DD).
* User’s Manual.
* Testing report.
  1. **Assumptions and Dependencies**

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

* This system will only be used only for government taxi service.
* Taxi service owns the vehicles, and the drivers work for the government.
* Taxi service is available 24 hours a day.
* Each taxi has a mobile phone, GPS system and a driver.
* Passenger has to pay if he accepts the taxi proposal.
* If a passenger cancels a taxi ride, he will be refunded 50% of the amount he paid.
* If a passenger cancels a taxi reservation before 48 hours, he will be refunded 70% of the amount he paid.
* If the passenger is not available at the pick up point, taxi waits up to 15 minutes, and after that the reservation is canceled, and the passenger is refunded 30% of the amount he paid.
* Passenger can reserve a taxi at least 2 hours before the desired time, and at most 1 week before the desired time.

1. **External Interface Requirements**
   1. **User Interfaces**

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

* 1. **Hardware Interfaces**

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>*

* 1. **Software Interfaces**

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

* 1. **Communications Interfaces**

*<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>*

1. **System Features**

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

* 1. **System Feature 1**

*<Don’t really say “System Feature 1.” State the feature name in just a few words.>*

4.1.1 Description and Priority

*<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>*

4.1.2 Stimulus/Response Sequences

*<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>*

4.1.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1:

REQ-2:

* 1. **System Feature 2 (and so on)**

1. **Other Nonfunctional Requirements**
   1. **Performance Requirements**

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>*

* 1. **Safety Requirements**

*<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>*

* 1. **Security Requirements**

*<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>*

* 1. **Software Quality Attributes**

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

* 1. **Business Rules**

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

1. **Other Requirements**

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

**Appendix A: Glossary**

*<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>*

**Appendix B: Analysis Models**

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

**Appendix C: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*