**Software Requirements Specification**

**for**

Delivery Company

**Version 1.0 approved**

**Prepared by Strimbeanu Mihai**

**Gavrila Sebastian**

**Sbaroi Ionut-Alexandru**

**MSA SRL**

**12/3/2024**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Sbaroi Ionut-Alexandru | 12/3/2024 | - added content to Intended Audience and Reading Suggestions - added content to Product functions  - added team members, creation date, and name of the project on the cover | 1.1 |
| Sbaroi Ionut-Alexandru | 15/3/2024 | -added content to Product Scope | 1.2 |
| Sbaroi Ionut-Alexandru | 15/3/2024 | -added content Operating Environment | 1.3 |

# **Introduction**

## **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.>*

## **Document Conventions**

*<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>*

## **Intended Audience and Reading Suggestions**

Types of Readers:

*-Developers: Developers will be interested in the detailed technical specifications, including system architecture, database design, and API documentation.*

*-Marketing Staff: Marketing staff may be interested in the high-level features and functionalities of the software to better grasp its market positioning and potential customer benefits.*

*-Users: Users will want to know how to interact with the system, including user authentication, order placement, and tracking functionalities.*

*-Testers: Testers will look for detailed functional and non-functional requirements to create test cases and ensure the software meets the specified criteria.*

*The SRS provides a detailed look into the Courier Management System. It covers everything from its scope, functional and non-functional requirements, system architecture, user roles, and security considerations. Additionally, you'll find in-depth descriptions of system behaviors, interfaces, and data requirements within the SRS.*

## **Product Scope**

*The software we're talking about here is a web application for a delivery company. Its main purpose is to make order management easier and more efficient. With this app, users can create, delete, and track orders, as well as automatically assign them to delivery employees.*

*Here are the key objectives and benefits of using this application:*

*-Efficient Order Management: This app will make it seamless for users to create and delete orders, reducing the need for manual effort and minimizing errors that can happen with traditional methods.*

*-Order Tracking: Users will be able to easily track the status and history of their orders, giving them transparency and improving customer satisfaction.*

*- Automated Assignment: The application will automate the process of assigning orders to delivery employees based on factors like proximity, workload, and availability. This feature aims to optimize delivery routes and ensure orders are fulfilled on time.*

*- Enhanced Productivity: By automating repetitive tasks and providing user-friendly interfaces for order management, this application will boost the productivity of both users and delivery employees.*

*-Improved Customer Experience: The app's efficient order handling and timely delivery will greatly enhance the overall customer experience, leading to customer loyalty and retention.*

*- Alignment with Business Goals: This software aligns perfectly with the company's goals of improving operational efficiency, optimizing resource utilization, and providing exceptional customer service.*

*By offering a centralized platform for order management and utilizing automation for order assignment, this web application aims to revolutionize the way orders are processed and fulfilled. Ultimately, it will contribute to the company's success in the competitive delivery services market.*

## **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.>*

# **Overall Description**

## **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.>*

## **Product Functions**

*User Authentication and Access Control:*

*-Users can log in with a username and password.*

*-Different user roles (Administrator, Client, Courier) have distinct access privileges.*

*Order Management:*

*-Users can place, cancel, and confirm orders.*

*-System automatically assigns couriers based on various criteria*

*-Users can monitor order status and view order history.*

*User Management:*

*-Administrators can add, edit, and manage employees.*

*-Couriers can mark the start and end of their workday/delivery and update order status.*

*Vehicle Management:*

*-Administrators can allocate and manage vehicles for courier operations.*

*Data Viewing:*

*-Users can view real-time information about orders, deliveries, and courier assignments.*

*Security:*

*-Ensure the protection of personal and sensitive data.*

## **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.>*

## **Operating Environment**

*Hardware Platform:*

*The servers hosting the application should have the capability to support the .NET framework and SQL Server. It is important to ensure that the hardware meets the minimum requirements in terms of CPU, RAM, and disk space to handle the expected workload and data storage needs.*

*Operating System and Versions:*

*During development and testing, the application will primarily be focused on Windows-based operating systems. For server deployment, it is recommended to use Windows 10 or 11, every version after version 1507 of Windows 10 is suitable .For development environments, Windows 10 or 11 is recommended.*

*Software Components:*

*The application will be developed using the MVC architecture with .NET 8, which ensures compatibility with the latest features and enhancements provided by the framework. SQL Server will be used as the relational database management system to store and manage data related to orders, users, delivery employees, and other relevant entities. Compatibility with SQL Server versions compatible with .NET 8 need to be minimum SQL Server 2005. Visual Studio 2022 or later will serve as the primary IDE for designing, coding, debugging, and testing the application. When it comes to web servers, options like Internet Information Services (IIS), Apache HTTP Server, Azure Devops WebApp and AWS can be chosen, as long as they are compatible with ASP.NET MVC applications and have the necessary configurations to support .NET applications.*

*Other Considerations:*

*To provide a consistent user experience across different platforms, the web application will be designed to be compatible with modern web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari. Security measures will also be integrated into the application, user authentication, authorization mechanisms, and data encryption, to protect sensitive information, everything built using EntityFramework.Identity Nuget package from .NET.*

## **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).>*

## **User Documentation**

*<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>*

## **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).>*

# **External Interface Requirements**

## **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.>*

## **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.>*

## **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.>*

## **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.>*

# **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.>*

## **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:

## **System Feature 2 (and so on)**

# **Other Nonfunctional Requirements**

## **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.>*

## **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.>*

## **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.>*

## **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.>*

## **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.>*

# **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.>*