

SerVICE

Name:Nicoara Tomas

Group: VI

Table of Contents

[Deliverable 1 3](#_Toc64843130)

[Project Specification 3](#_Toc64843131)

[Functional Requirements 3](#_Toc64843132)

[Use Case Model 3](#_Toc64843133)

[Use Cases Identification 3](#_Toc64843134)

[UML Use Case Diagrams 3](#_Toc64843135)

[Supplementary Specification 3](#_Toc64843136)

[Non-functional Requirements 3](#_Toc64843137)

[Design Constraints 3](#_Toc64843138)

[Glossary 3](#_Toc64843139)

[Deliverable 2 3](#_Toc64843140)

[Domain Model 3](#_Toc64843141)

[Architectural Design 4](#_Toc64843142)

[Conceptual Architecture 4](#_Toc64843143)

[Package Design 4](#_Toc64843144)

[Component and Deployment Diagram 4](#_Toc64843145)

[Deliverable 3 4](#_Toc64843146)

[Design Model 4](#_Toc64843147)

[Dynamic Behavior 4](#_Toc64843148)

[Class Diagram 4](#_Toc64843149)

[Data Model 4](#_Toc64843150)

[System Testing 4](#_Toc64843151)

[Future Improvements 4](#_Toc64843152)

[Conclusion 4](#_Toc64843153)

[Bibliography 4](#_Toc64843154)

# Deliverable 1

## Project Specification

This project is a web application that allows users to sell services. It consists of four main entities: User, Service, Category, and Order.

## Functional Requirements

1.User registration and login: Users must be able to register for an account with the web app using their email address and create a username and password. They should be able to log in and out of the app using their credentials.

2.Service creation and editing: Users should be able to create new services to sell on the web app, including adding a title, description, price, and category. They should also be able to edit and delete their existing services.

3.Service ordering and payment: Buyers should be able to browse and search for services on the web app, and place orders for the services they want to purchase. They should be able to pay for services using a secure payment gateway.

4.Service categorization and filtering: Services should be organized into categories, and buyers should be able to filter services by category when browsing and searching.

5.Order management: Sellers should be able to manage their orders, including viewing order details and marking orders as complete. Buyers should be able to view their order history and track the status of their current orders.

## Use Case Model 1

### Use Cases Identification

Use-Case: *Create Account*

Level: User Goal

Primary Actor: User

Main success scenario: The user creates an account by providing a valid email address, a unique username, and a password.

Extensions: If the email address or username is already in use, the system displays an error message.

Use-Case: *Create Service*

Level: User Goal

Primary Actor: User

Main success scenario: The user creates a service by providing a title, a description, a price, and one or more categories.

Extensions: If the user is not logged in, the system redirects them to the login page.

Use-Case: *Searching for Services*

Level: User Goal

Primary Actor: Buyer

Main success scenario:

The buyer enters a keyword or category to search for services.

The system displays a list of services matching the search criteria.

The buyer selects a service from the list.

The system displays detailed information about the service.

Extensions:

If no services match the search criteria, the system displays a message indicating that no results were found.

Use-Case: *Buying Service*

Level: User Goal

Primary Actor: Buyer

Main success scenario:

The buyer selects a service they wish to purchase.

The system displays the details of the selected service, including the price.

The buyer confirms that they wish to purchase the service.

The system prompts the buyer to enter their payment details.

The buyer enters their payment details and confirms the purchase.

The system confirms that the purchase was successful and updates the buyer's order history.

Extensions:

If the buyer's payment is declined, the system displays an error message and prompts the buyer to enter a different payment method.

If the service is no longer available (e.g. it has been deleted by the seller), the system displays an error message and prompts the buyer to select a different service.  
  
Use-Case: *Update Service*

Level: User Goal

Primary Actor: Service Provider

Main Success Scenario:

The Service Provider selects a Service they wish to update.

The Service Provider makes the desired updates to the Service's information (e.g., title, description, price, categories).

The Service Provider saves the changes.

The Service information is updated in the system and is now displayed correctly on the Service Provider's profile and on search results.

Extensions:

If the Service Provider makes a mistake while updating the Service information, they can choose to cancel the update without saving the changes.

Use-Case: *Delete Service*

Level: User Goal

Primary Actor: Service Provider

Main Success Scenario:

The Service Provider selects a Service they wish to delete.

The Service Provider confirms that they wish to delete the Service.

The Service information is removed from the system and is no longer displayed on the Service Provider's profile or on search results.

Extensions:

If the Service Provider changes their mind about deleting the Service, they can choose to cancel the deletion before confirming it.

Use-Case: *Delete Account*

Level: User Goal

Primary Actor: User

Main Success Scenario:

The User selects the "Delete Account" option.

The User is prompted to confirm that they wish to permanently delete their account.

If the User confirms, all of their information (including Services and Orders) is deleted from the system.

The User is logged out and redirected to the homepage.

Extensions:

If the User changes their mind about deleting their account, they can choose to cancel the deletion before confirming it.

If the User has pending orders or services, they will be unable to delete their account until those are completed or transferred to another user. In this case, they will receive an appropriate error message.

### UML Use Case Diagrams

Diagram

Description automatically generated

## Supplementary Specification

### Non-functional Requirements

1.Performance: The application should be able to handle a large number of users and services without significant delays or downtime. This is important because the success of the application depends on its ability to provide a fast and reliable service to its users.

2.Security: The application should be designed with security in mind, to protect user data and prevent unauthorized access or malicious attacks. This is important because users will be providing sensitive information such as their email address and payment details.

3.Usability: The application should be easy to use and navigate, with clear and intuitive interfaces. This is important because users will not use an application that is difficult or confusing to use.

4.Compatibility: The application should be compatible with a wide range of devices and browsers, to ensure that users can access it from anywhere. This is important because users may access the application from different devices and browsers.

### Design Constraints

1.ASP.NET Core 7: The web application must be built using ASP.NET Core 7 framework. This framework is a modern, cross-platform framework for building web applications and is suitable for this project's requirements.

2.Microsoft SQL Server Express: The database for the web application must use Microsoft SQL Server Express. This is a free and lightweight version of the popular SQL Server database management system and is suitable for this project's requirements.

3.Front-end Technologies: The front-end of the web application must be developed using Blazor, CSS, HTML, and Razor. Blazor is a modern and open-source web framework for building web apps using C# and HTML. CSS and HTML are industry-standard web technologies for styling and structuring web pages, respectively. Razor is a markup syntax used to create dynamic web content in .NET.

4.Coding standards: The code developed for this project must follow the established coding standards and best practices for the ASP.NET Core and Blazor frameworks. These standards will ensure that the code is maintainable, scalable, and easily understandable by other developers who may work on the project in the future.

5.Testing: The web application must be thoroughly tested to ensure that it is free from defects and meets the functional and non-functional requirements. The testing process should include unit testing, integration testing, and end-to-end testing.

6.Deployment: The web application must be deployed on a suitable hosting platform that supports ASP.NET Core 7 and Microsoft SQL Server Express. This may include cloud-based hosting platforms such as Microsoft Azure, Amazon Web Services, or Google Cloud Platform, or on-premises hosting solutions.

## Glossary

User: A person who uses the application to sell or order services.

Service: A task or job that a user offers to perform for a fee.

Category: A classification of services based on their type or subject.

Order: A request by a buyer to purchase a service from a seller.

## Deliverable 2

## Domain Model

[Define the domain model and create the conceptual class diagrams]

## Architectural Design

### Conceptual Architecture

[Define the system’s conceptual architecture; use an architectural style and pattern - highlight its use and motivate your choice.]

### Package Design

[Create a package diagram]

### Component and Deployment Diagram

[Create the component and deployment diagrams.]

# Deliverable 3

## Design Model

### Dynamic Behavior

[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]

### Class Diagram

[Create the UML class diagram; apply GoF patterns and motivate your choice]

## Data Model

[Create the data model for the system.]

# System Testing

[Describe the testing methides and some test cases.]

# Future Improvements

[Present some features that apply to the application scope.]

# Conclusion

# Bibliography