Turcsa Alexandru

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

2. Non-functional Requirements 4

2.1 Availability 4

2.2 Performance 4

2.3 Security 4

2.4 Testability 4

2.5 Usability 4

3. Design Constraints 4

1. Introduction

The client-server application for the supermarket is designed to provide a secure and efficient way for employees to sell products, manage user accounts and products, and generate reports. This Supplementary Specification captures the non-functional requirements and design constraints of the system.

1. Non-functional Requirements

2.1 Availability

* Quality attribute definition: The system should be available 24/7.
* Source of stimulus: All users of the system.
* Stimulus: Accessing the system.
* Environment: Any time of day, any day of the week.
* Artifact: The entire system.
* Response: The system should respond to requests in a timely manner and be accessible at all times.
* Response measure: The system should have a uptime of at least 99.9%.
* Tactics: Implementing redundancy, backup, and failover mechanisms to ensure high availability.

2.2 Performance

* Quality attribute definition: The system should respond quickly to user requests.
* Source of stimulus: All users of the system.
* Stimulus: Accessing the system, adding products to a sale, computing the total price, decreasing product stocks, and generating reports.
* Environment: Any time of day, any day of the week.
* Artifact: The entire system.
* Response: The system should respond to requests quickly and efficiently.
* Response measure: The system should have an average response time of less than 1 second.
* Tactics: Implementing caching, database indexing, and optimizing algorithms to improve performance.

2.3 Security

* Quality attribute definition: The system should be secure and protect sensitive information.
* Source of stimulus: Unauthorized users.
* Stimulus: Attempting to access sensitive information or perform unauthorized actions.
* Environment: Any time of day, any day of the week.
* Artifact: The entire system.
* Response: The system should prevent unauthorized access and protect sensitive information.
* Response measure: The system should meet industry-standard security practices and guidelines.
* Tactics: Implementing secure authentication mechanisms, data encryption, and access control mechanisms.

2.4 Testability

* Quality attribute definition: The system should be testable and maintainable.
* Source of stimulus: Testing and maintenance teams.
* Stimulus: Conducting tests and performing maintenance.
* Environment: Any time of day, any day of the week.
* Artifact: The entire system.
* Response: The system should be easy to test and maintain.
* Response measure: The system should have high test coverage and be easy to modify and update.
* Tactics: Implementing automated testing, version control, and documentation.

2.5 Usability

* Quality attribute definition: The system should be easy to use and intuitive.
* Source of stimulus: All users of the system.
* Stimulus: Accessing the system, adding products to a sale, computing the total price, decreasing product stocks, and generating reports.
* Environment: Any time of day, any day of the week.
* Artifact: The entire system.
* Response: The system should be easy to navigate and use, with clear instructions and feedback.
* Response measure: The system should have a low error rate and a high user satisfaction rate.
* Tactics: Implementing a user-friendly interface, clear instructions, and helpful feedback.

1. Design Constraints

* The application should be developed using Python as the programming language and Flask as the web framework.
* The database used should be MySQL.
* The application should be deployed on a Linux server.
* The application should adhere to industry-standard security practices and guidelines.