ElectroStock

Software Requirements Specification

- Lead Software Developer: Tomas Mariano Guell
- Tomasgenari@gmail.com

- Final Version

Instituto Tecnico Salesiano Villada

Integrants:

- Canepa Ignacio
- Cimatti Bautista
- Genari Tomas
- Nacho
- Mariano Dubois
- Giayetto Francisco
- Morales Franco Dimaria

Deadline:

14 De Noviembre de 2023

Teacher:

- Engineer Frattin John

Revision History

Date	Description	Author	Comments
11/06/23	First renovation and verification	Genari Tomás	Information renewal and correction based on the documenter's indications.
14/11/23	Last revision and finalization	Genari Tomas	Information renewal and correction based on the documenter's indications.

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date

1. Introduction

The purpose of this document is to provide a comprehensive and detailed description of the software requirements for the Electronics Inventory Management System. This document will serve as a guide for the development of the system, ensuring that all necessary functionalities and features are clearly defined. Additionally, the document establishes acceptance criteria to validate that the system meets the standards and expectations of the client. Furthermore, the document provides an overview of the project's objectives, scope, and constraints, enabling efficient and effective project management.

1.1 Purpose

The SRS outlines the system's functional and non-functional requirements, as well as the constraints, assumptions, and dependencies associated with the project. It serves as a reference for all stakeholders involved in the project, providing a clear understanding of what the system is expected to do and how it should behave. Additionally, the SRS helps ensure that the final product meets the expectations of the client and end-users.

1.2 Scope

The system will enable the school to track the inventory of electronic components, as well as provide a user-friendly interface for teachers, students, and staff to manage stock levels, monitor usage, and request restocking when necessary.

The key features of the system include:

Inventory management: The system will enable the school to track the inventory of electronic components, tools, and equipment and monitor stock levels.

Stock request and restocking: The system will provide a platform for teachers to request restocking of items when stock levels are running low.

Budget Calculation: It is a crucial aspect of the project. It involves the determination of a budget based on the stock losses that occurred during the year for various reasons. This budget will be calculated using a specialized function, which will also enable the area managers to estimate the renewal cost of stock. This estimation will be based on the current peso/dollar currency exchange rate. The budget calculation and renewal estimation process are important for ensuring that the project stays within financial constraints and that stock availability is maintained at all times.

Master/student loan system: The system will allow teachers to loan equipment and components to students for use in classroom projects or at home. It will also track the loan history of items borrowed by students from teachers and provide alerts for overdue or lost items.

1.3 References

IceScrum Project.

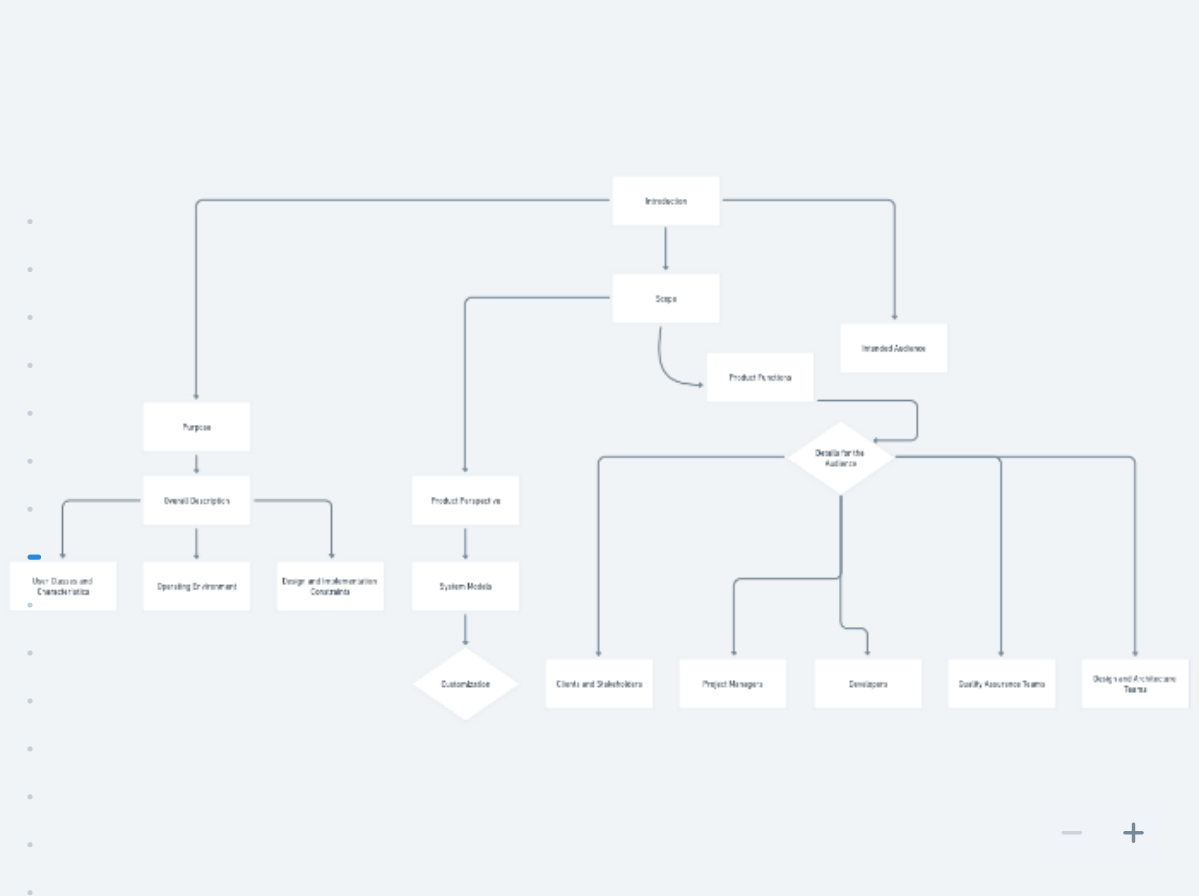
GitHub repo with project progress and documentation.

Postman group project.

1.4 Overview

The Software Requirements Specification (SRS) document serves as a comprehensive blueprint for the development of a software system. It outlines the functional and non-functional requirements, design constraints, and system behavior. The SRS is a critical document in the software development life cycle, acting as a reference for stakeholders, developers, and quality assurance teams.

Document Organization: The SRS document is organized into several key sections, each serving a specific purpose.



2. General Description

This project aims to improve the efficiency with which students and teachers manage both the stock and its tracking. Therefore, as a team, we have projected a series of functionalities based on the preparation of multiple user stories and tasks that later served as a guide to estimate the complexity that the team would encounter when starting.

2.1 General Constraints

Integration with existing systems: The developers need to ensure that the new system integrates seamlessly with the school's existing systems to prevent any disruptions or data inconsistencies.

Security and privacy: The system must adhere to the school's data security and privacy policies, which may include access control, data encryption, and user authentication protocols.

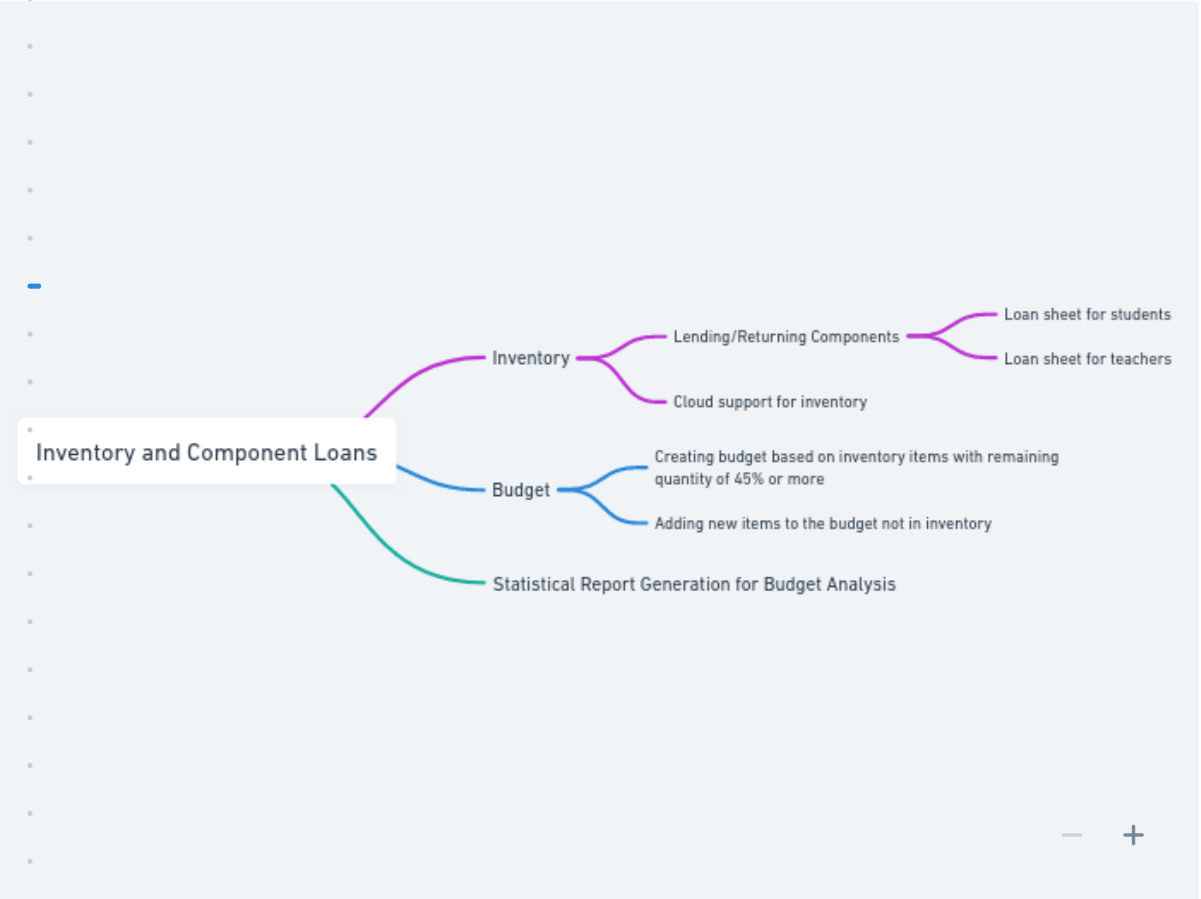
Time constraints: We must deliver the system within a specified timeline, which was already delivered by the school academics.

User interface design: The system must be designed with a user-friendly interface that is easy

to navigate, intuitive, and efficient for all users, including teachers, students, and staff.

---

3. Specific Requirements



Inventory and Component Loans

3.1 External Interface Requirements

1. *Security: The system should have security features such as authentication and access control to protect against unauthorized access to data. The system should also have backup and recovery mechanisms in place to prevent data loss in case of system failure.*
2. *Reporting: The system should allow for the generation of reports and data analysis to provide insights into inventory levels, loan history, and other relevant metrics. Reports should be customizable and exportable to standard formats such as PDF or Excel.*

3.1.1 User Interfaces

1. *The system requires user accounts to be set up for each teacher, student, and staff member who will use the system. User account information, including login credentials and personal information, must be stored securely and accessible only to authorized personnel. The students' accounts will be automatically created adding their data in the*

models section of the project, using for this task an already prepared excel provided by the school directives.

3.1.2 Hardware Interfaces

1. *The system requires a computer or mobile device with internet connectivity to access the software interface. The hardware should be capable of running modern web browsers and have adequate storage to save data.*
2. **Import Formats** To facilitate data uploading, the system will allow users to import data in various formats. The following details the supported import formats:

    a. **JSON**: Users can import data using JSON files. The JSON format will be compatible with the system's data structure and will be validated to ensure data integrity.

```
[    {
        "nombre": "Francisco",
        "apellido": "Giayetto",
        "dni": "12345678",
        "email": "ejemploi@gmail.com",
        "especialidades": "programacion",
        "grupos": "Alumno"
    }
]
```

    b. **XLS (Microsoft Excel)**: The system will accept XLS files for data import. Data in XLS format will be processed and its consistency will be verified before import.

nombre	apellido	dni	email	especialidades	grupos
Francisco	Giayetto	12345678	ejemplo.gmail.com	programacion	Alumno

    c. **CSV (Comma-Separated Values)**: Users can upload data through CSV files. The system will parse CSV files and import them after validating the information.

```
nombre,apellido,dni,email,especialidades,grupos
Francisco,Giayetto,12345678,ejemploi@gmail.com,programacion,Al
umno
```

    d. **XLSX (Microsoft Excel, Modern Format)**: In addition to XLS files, XLSX files will be supported for data import. The system will perform checks to ensure that data conforms to the intended structure.

nombre	apellido	dni	email	especialidades	grupos
Francisco	Giayetto	12345678	ejemplo.gmail.com	programacion	Alumno

e. **TSV (Tab-Separated Values)**: The system will accept TSV files for data uploading. TSV files will be analyzed, and data accuracy will be verified before import.

```
nombre    apellido dni       email           especialidades gr
upos
Francisco  Giayetto  12345678 ejemploi@gmail.com programacion
Alumno
```

f. **YAML**: Users will also have the option to upload data through YAML files. YAML format should adhere to the system's data structure and undergo validation before import.

```yaml
- nombre: Francisco
  apellido: Giayetto
  dni: "12345678"
  email: ejemploi@gmail.com
  especialidades: programacion
  grupos: Alumno
```

3.1.3 Software Interfaces

1. *The system requires a modern web browser such as Google Chrome, Mozilla Firefox, or Microsoft Edge to access the software interface. The software must be compatible with the latest versions of these web browsers.*

3.1.4 Communications Interfaces

1. *The system should allow for the import and export of data in standard formats such as CSV, Excel, or JSON. This will enable data to be easily transferred between the system and other software applications.*

3.2 Functional Requirements

***Inventory management**: The system allows the user with a loan to add, edit, and delete products from the inventory. Each product should have a unique identifier, name, description, stock level, location, and other relevant information.*

***Stock level monitoring**: The system monitors stock levels for each product and notify users when stock levels fall below a specified threshold. Teachers should be able to view the current*

stock level of a product at any time. Also the teachers should be able to calculate the yearly renewal of the stock based on the dollar prices in that actual moment.

Product categorization: *The system allows the user to categorize products by type, brand, or other relevant categories to make it easier to find products and generate reports.*

Loan management*: The system allows the user to loan products to students and staff and keep track of loan history. The user should be able to see which products have been loaned, to whom, and when they are due to be returned.*

Loan request: *The system allows the student to request a loan of a product, and the teacher should be able to approve or reject the request.*

Checkout*: The system allows the user to add products to a checkout chart and process the checkout.*

Reporting*: The system allows the user to generate reports on inventory levels, loan history, checkout history, and other relevant metrics. Reports should be customizable and exportable to standard formats such as PDF or Excel.*

Budget graphic*: The system provides the functionality to generate a comprehensive graphic report of the budget. This report includes visual representations such as charts, graphs, and tables to present the budget data in a clear and intuitive manner. The report displays the allocation of funds across different categories, highlighting the expenditure and income sources.*

Search*: The system allows the user to search for products by name, description, or other relevant criteria to find specific products quickly.*

User roles*: The system has different user roles such as administrator, teacher, and student, with different levels of access to different features.*

Notifications*: The system sends notifications to users when a loan request is approved, when a product is due for return, or when stock levels fall below a specified threshold.*

3.5 Non-Functional Requirements

The non-functional requirements include usability, performance, reliability, security, scalability, accessibility, interoperability, maintainability, availability, and compliance. These requirements ensure that the system provides a high-quality user experience, meets industry standards, and is secure, reliable, and scalable to meet the needs of a growing user base over time.

3.5.1 Performance

1. *The system responds quickly to user requests, provide real-time information, and handle a*

considerable volume of transactions without delays or errors.

3.5.2 Reliability

1. *The system is reliable, available 24/7(available at least in school time).*

3.5.3 Availability

1. *The system has a high level of availability, with a minimum uptime of 99%, and a backup and disaster recovery plan in place.*

3.5.4 Security

1. *The system should be secure, with appropriate measures in place to protect sensitive data and prevent unauthorized access, data loss, or data corruption.*

3.5.5 Maintainability

1. *The system should be maintainable, with well-documented code, easy-to-maintain software architecture, and an efficient update process.*

3.5.6 Portability

1. *The system should be interoperable with other systems, applications, and devices, using open standards and APIs to facilitate integration.*

3.7 Logical Database Requirements

***Inventory Management**: The system must have a database that can store information about the products available in the inventory, such as product name, description, quantity, price, and availability.*

***Loan Management**: The system must have a database that can store information about the loan transactions, such as loan date, return date, borrower name, and borrower ID.*

***User Management**: The system must have a database that can store information about users, such as their name, contact information, and user ID.*

***Authorization Management**: The system must have a database that can store information about user roles, access rights, and privileges.*

***Transaction Management**: The system must have a database that can store information about transactions, such as sales, loans, and returns.*

***Reporting**: The system must have a database that can store information required to generate reports, such as transaction history, inventory levels, and loan status.*

***Backup and Recovery**: The system must have a database backup and recovery mechanism to ensure that data is safe and available in case of a system failure or disaster.*

***Performance**: The database must be designed for optimal performance, including query optimization, indexing, and caching, to ensure that the system operates efficiently.*

3.8 Other Requirements

1. ***Mobile Compatibility**: The system may need to be compatible with mobile devices, to allow users to access the system on the go.*
2. ***Multi-language Support**: The system may need to support multiple languages, to accommodate users who speak different languages.Also we are tracing a plan to include disabled people so we can hoard all the user experience possibilities.*
3. ***Customization**: The system may need to provide customization options, such as customizable fields or reports, to meet the specific needs of different users or organizations.In this case we might be able to let other academics modify the system, in case something goes wrong in the time we can't do software maintenance.*