Software Requirements Specification

for

E-commerce web application

Version 1.0 approved

Prepared by Jacob Roberts, Ethan Mitchell, Jacob Hollis, Ashanti Cocroft, and Matthew Rester

Intro to SE Group 6

28 August 2024

# Introduction

## Purpose (Jacob R)

Introduction to Software Engineering Group 6 is creating the first revision of a web application for e-commerce. This document is the Software Requirement Specification (SRS) for the e-commerce web application. It describes all of the features of the website.

The purpose of the SRS is to provide detailed requirements for the e-commerce web site. The website will support three user roles: Buyer, Seller, and Admin. Each user role will allow for different actions to be completed, and all users will be able to login.

Buyer actions:

1. Search products
2. Compare products
3. Buy products
4. Return products

Seller actions:

1. Add products
2. Sell products
3. Receive payments

Admin action:

1. Approve/block new user accounts
2. Oversee user actions

<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 (Jacob R)

This SRS uses normal Arial size 11 font. Each requirement has a tag to describe the function it belongs to that is in capital letters followed by a dash and a sequence number.

<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

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>

## Product 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.>

## 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(AC)

The e-commerce platform described in this SRS is a standalone product designed for buyers, sellers, and administrators. It offers essential features like user authentication, product management, and transaction processing. Though it can work on its own, it can also be integrated into larger systems, interacting smoothly with inventory management, CRM tools, and payment gateways through clear interfaces.

<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(AC)

The e-commerce platform needs to handle some essential tasks based on different user roles. Users, whether they’re Buyers, Sellers, or Admins, should be able to log in and out easily. Buyers should be able to search for products, compare different options, buy items, and return them if needed. Sellers should be able to list their products, manage sales, and get paid. Admins will have the ability to approve or block user accounts and product listings, and keep an eye on what everyone’s doing to make sure everything runs smoothly. All these functions are linked, with each role doing their part to keep the platform working as it should.

<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.>

## User Classes and Characteristics (Jacob H)

<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.>

* Admin
  + Approve/block new user accounts and products
  + Oversee different user actions
* Seller (Most important user class)
  + Add new products, sell and receive payments for their available products
* Buyer
  + Able to search, compare, buy and return different types of products from different sellers

## Operating Environment(AC)

The e-commerce will operate and run in a python-based environment. The platform will be developed and running using python with a database Django, or Flask

<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.>

## Design and Implementation Constraints(AC)

Design: Language Python

* Regulatory Policies: The platform must comply with data protection laws like GDPR or CCPA, especially when handling user data and transactions.
* Security Considerations: Developers will need to implement strong security features, such as SSL encryption, secure payment methods, and user authentication to protect user information and prevent security breaches.
* Hardware Limitations: The platform must be optimized to run smoothly on different devices, including those with limited processing power and memory, to ensure a good user experience.
* Integration with Other Systems: The platform may need to connect with existing systems like inventory management, and payment gateways, which might require sticking to specific APIs and protocols, limiting some of the flexibility in how things are built.
* Technology Stack: The choice of programming languages, frameworks, and databases might be limited by what the development team knows best or what the customer's current setup supports, which could influence the overall approach.
* Design Conventions and Coding Standards: Developers will need to follow certain design and coding standards, especially if the customer’s team will be maintaining the software afterward. This includes the following best practices for coding, documentation, and version control.
* **Parallel Operations:** The platform needs to handle multiple users doing things at the same time, so developers will have to make sure it can manage parallel processes efficiently to avoid slowdowns and ensure it can scale as needed.

<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).>

# 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.>

## Login (Jacob R)

### Description and Priority

The login feature allows people to use the website as a registered user. This is a high priority feature.

<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).>

### Stimulus/Response Sequences

1. User selects login
2. System shows fields for email and password
3. User fills in fields
4. System validates user as registered user

<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.>

### 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-Login-1: The website shall have a login page.

REQ-Login-2: The login page shall have an email field.

REQ-Login-3: The login page shall have a password field.

REQ-Login-4: The login page shall have a submit button.

REQ-Login-5: if the email and password combination does not exist as a user account, inform the user.

## Registration (Jacob R)

### Description and Priority

The registration feature allows users without an account to create an account. This is a high priority feature.

### Stimulus/Response Sequences

1. User selects register
2. System shows fields for email and password
3. User fills in fields
4. System creates account with given email and password combination

### Functional Requirements

REQ-Register-1: The website shall have a register page.

REQ-Register-2: The register page shall have an email field.

REQ-Register-3: The register page shall have a password field.

REQ-Register-4: The register page shall have a submit button.

REQ-Register-5: if the email address is already associated with a user, inform the user that an account with that email already exists.

## Search products (Jacob R)

### Description and Priority

The search products feature allows buyers to explore available products by using keywords. This is a medium priority feature.

### Stimulus/Response Sequences

|  |  |
| --- | --- |
| Requirement ID | Desc |

### Functional Requirements

## Compare products

## Buy products

## Return products

## Add products

## Sell products

## Receive payments

## Approve/block new user accounts

## Oversee user actions

# Other Nonfunctional Requirements (Matthew)

## Performance Requirements

Given that many users may access this site simultaneously, the platform must have the ability to accommodate many concurrent users. The performance of the website should not substantially reduce in response to concurrent users. Also, the timing of user interaction needs to be accurately tracked so that features work as intended. For example, if two users place the same item in their cart while there is only one left in inventory, then whichever buyer completes the purchase first should be able to proceed, while the other is accurately denied access due to empty inventory.

The Django framework includes built in tools to evaluate the performance of code. For instance, one tool displays the number of SQL queries the page is generating, alongside the amount of time each one takes. It also displays the number of queries that are similar or identical.

## Safety Requirements

[None as far as I can tell, might just remove this subsection and renumber the ones below]

<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

The website must be made in a secure way that adequately protects the private information of all users. To create an account, users have to provide certain pieces of information such as an email address. This information should not readily be available to other user accounts. Also, all buyers and sellers have to provide information to participate in financial transactions.

Therefore, access to read/write any information from/to a user’s account is restricted to only that user. Admin accounts are about to view and oversee certain actions committed by other user accounts. The ability to read product information is in no way restricted, but the ability to write to product information is exclusive to the seller account that posted the product.

The Django framework is used, which includes many safety features, including protection against cross site scripting, cross site request forgery, SQL injection, and clickjacking.

<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

The website should be available to users at any time of day and should be automated to work without the presence of workers.

<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.>

# 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.>