**CEBU INSTITUTE OF TECHNOLOGY**

**UNIVERSITY**

COLLEGE OF COMPUTER STUDIES

Software Requirements Specifications

for

Task Bounty

Cabiling Patrick  
Vijar Justin

Stephen Aizun D. Rosal

Change History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date Changed** | **Member** | **Modified** |
| 1 | 1/31/2025 | Cabiling Patrick, Vijar Justin, Rosal Stephen | Introduction, Overall Description, Specific Requirements |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

Change History 3

Table of Contents 4

1. Introduction 5

1.1. Purpose 5

1.2. Scope 5

1.3. Definitions, Acronyms and Abbreviations 6

1.4. References 6

2. Overall Description 7

2.1. Product perspective 7

2.2. User characteristics 7

2.4. Constraints 8

2.5. Assumptions and dependencies 8

3. Specific Requirements 10

3.1. External interface requirements 10

3.1.1. Hardware interfaces 10

3.1.2. Software interfaces 10

3.1.3. Communications interfaces 10

3.2. Functional requirements 10

Module 1 11

Module 2 18

3.4 Non-functional requirements 18

Performance 18

Security 19

Reliability 19

# Introduction

## Purpose

* *Describe the purpose of the SRS;*
* *Specify the intended audience for the SRS.*

The purpose of this document is to define the requirements for the **Task Bounty** system. This system is designed to facilitate a bounty-driven task management platform where users can post tasks with associated bounties. Tasks can either be assigned to specific individuals by the **Bounty Creator** or left unassigned to operate on a first-come, first-serve basis. The system ensures secure task resolution and bounty distribution.

The intended audience for this document includes:

* **Developers**: To understand the system requirements and implement the solution.
* **Stakeholders**: To review the system's functionality and ensure it meets business needs.

## Scope

* *Identify the software product(s) to be produced by name (e.g., Host DBMS, Report Generator, etc.);*
* *Explain what the software product(s) will, and, if necessary, will not do;*
* *Describe the application of the software being specified, including relevant benefits, objectives, and goals;*
* *Be consistent with similar statements in higher-level specifications (e.g., the system requirements specification), if they exist.*

The **Task Bounty** system will include the following software products:

* **Web Application**: A ReactJS-based frontend for task posting, management, and bounty distribution.
* **Mobile Application**: An Android Kotlin-based app for task alerts, notifications, and submissions.
* **Backend Server**: A Java Spring Boot-based server to handle business logic, database interactions, and API integrations.

**The system will:**

* Allow users to post tasks with bounties.
* Enable users to search, filter, and apply for tasks.
* Provide a dashboard for Bounty Creators to manage tasks and bounties.
* Integrate with **Stripe** for payment processing and **Google** for third-party authentication.
* Use **MongoDB Atlas** as the external database for data storage and management.

**The system will not:**

* Support offline functionality for the mobile application.
* Provide advanced analytics or reporting beyond basic task and bounty tracking.

## Definitions, Acronyms and Abbreviations

* *provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS*
* **SRS**: Software Requirements Specification.
* **CRUD**: Create, Read, Update, Delete.
* **API**: Application Programming Interface.
* **Bounty Creator**: A user who posts tasks and offers bounties.
* **Hunter:** A user who completes tasks and claims bounties.

## References

* *Provide a complete list of all documents referenced elsewhere in the SRS;*
* *Identify each document by title, report number (if applicable), date, and publishing organization;*
* *Specify the sources from which the references can be obtained.*

# Overall Description

## Product perspective

* *Put software product into perspective with other related products. If the product is independent and totally self-contained, it should be so stated here. If the SRS defines a product that is a component of a larger system, as frequently occurs, then this subsection should relate the requirements of that larger system to functionality of the software and should identify interfaces between that system and the software.*
* *A block diagram showing the major components of the larger system, interconnections, and external inter- faces can be helpful.*
* *Describe the modular decomposition of the components using the format below:*

*Module 1*

*Transaction 1.1*

*Transaction 1.2*

*Module 2*

*Transaction 2.1*

*Transaction 2.2*

*. . .*

The **Task Bounty** system is a standalone application that integrates with external services such as **Stripe** for payments and **Google** for authentication. The system follows a **client-server architecture** with the following components:

* **Web Application**: Built using ReactJS, it serves as the primary interface for task posting and management.
* **Mobile Application**: Built using Android Kotlin, it provides task alerts, notifications, and submission capabilities.
* **Backend Server**: Built using Java Spring Boot, it handles business logic, database interactions, and API integrations.

The system will interact with **MongoDB Atlas** for data storage and management.

## User characteristics

* *Describe all user types and their roles and privileges in the system*
* **Bounty Creator**:
  1. Can post tasks with bounties.
  2. Can assign tasks to specific users or leave them open for anyone to claim.
  3. Can manage tasks and approve submissions.
* **Hunter**:
  1. Can search and filter tasks.
  2. Can claim and submit tasks for approval.
  3. Can receive notifications about new tasks and updates.
* **Admin**:
  1. Can manage users and tasks.
  2. Can monitor system performance and resolve issues.

## 2.4. Constraints

* *Provide a general description of any other items that will limit the developer’s options.*
* *Regulatory policies;*
* *Hardware limitations (e.g., signal timing requirements);*
* *Interfaces to other applications;*
* *Parallel operation;*
* *Audit functions;*
* *Control functions;*
* *Reliability requirements;*
* *Criticality of the application;*
* *Safety and security considerations.*
* The mobile application must support Android devices running **Android 8.0 (Oreo)** or later.
* The system must integrate with **Stripe** for payment processing and **Google** for third-party authentication.
* The database must be hosted on **MongoDB Atlas**.

## 2.5. Assumptions and dependencies

*This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not avail- able, the SRS would then have to change accordingly.*

* The system assumes that **Stripe** and **Google** APIs will be available and functional.
* The system assumes that users will have access to a stable internet connection.
* The system depends on **MongoDB Atlas** for data storage and management.

# Specific Requirements

## External interface requirements

### 3.1.1. Hardware interfaces

*This should specify the logical characteristics of each interface between the software product and the hard- ware components of the system. This includes configuration characteristics (number of ports, instruction sets, etc.). It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.*

* *The system will support standard web browsers (Chrome, Firefox, Safari) for the web application.*
* *The mobile application will support Android devices with a minimum of* ***2GB RAM*** *and* ***Android 8.0 (Oreo)****.*

### 3.1.2. Software interfaces

*This should specify the use of other required software products (e.g., a data management system, an operating system, or a mathematical package), and interfaces with other application systems (e.g., the linkage between an accounts receivable system and a general ledger system).*

* The backend server will use **Java Spring Boot** for business logic and API integrations.
* The web frontend will use **ReactJS** for the user interface.
* The mobile application will use **Android Kotlin** for development.
* The system will integrate with **Stripe** for payment processing and **Google** for third-party authentication.

### 3.1.3. Communications interfaces

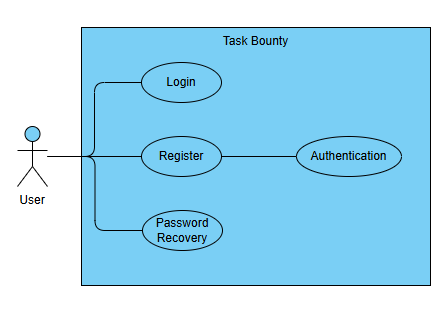
*This should specify the various interfaces to communications such as local network protocols, etc.*

* The system will use **RESTful APIs** for communication between the web application, mobile application, and backend server.
* The system will use **HTTPS** for secure communication.

## Functional requirements

### Module 1

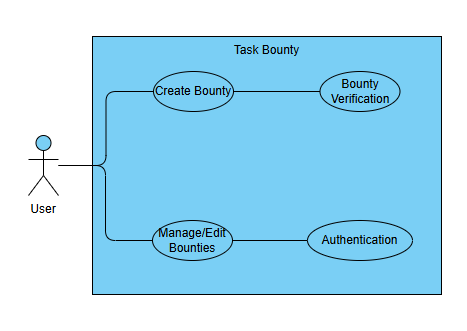
#### 1.1 User Registration & Login



The **User Registration & Authentication** function ensures secure access to the Task Bounty platform. Users can create an account using either **email/password** or **Google Account login** for convenience.

* Email/password or Google Account login.
* Two-factor authentication (2FA) optional for MVP (TOTP).
* Password recovery via email.
* Session management (auto-logout after 30 minutes of inactivity).

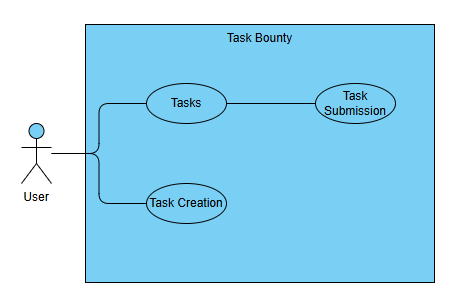
#### 1.2 Project Posting and Management



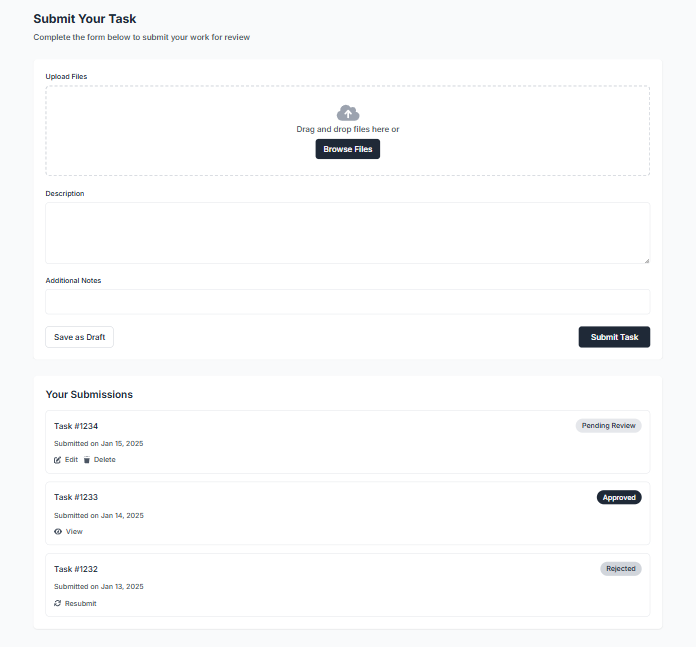
The **Project Posting & Management** function allows Bounty Creators to create, edit, and oversee task bounties while ensuring a secure payment process.

* The Bounty Creator inputs the **title, description, deadline, category, and bounty reward**.
* The system verifies the details and prompts the user for **payment via Stripe**.
* If the payment is successful, the bounty is posted and becomes available for Hunters.
* The Bounty Creator can manage posted bounties through the dashboard (edit or cancel if needed).

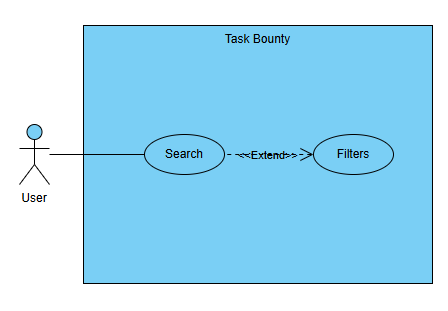
#### 1.3 Task Submissions



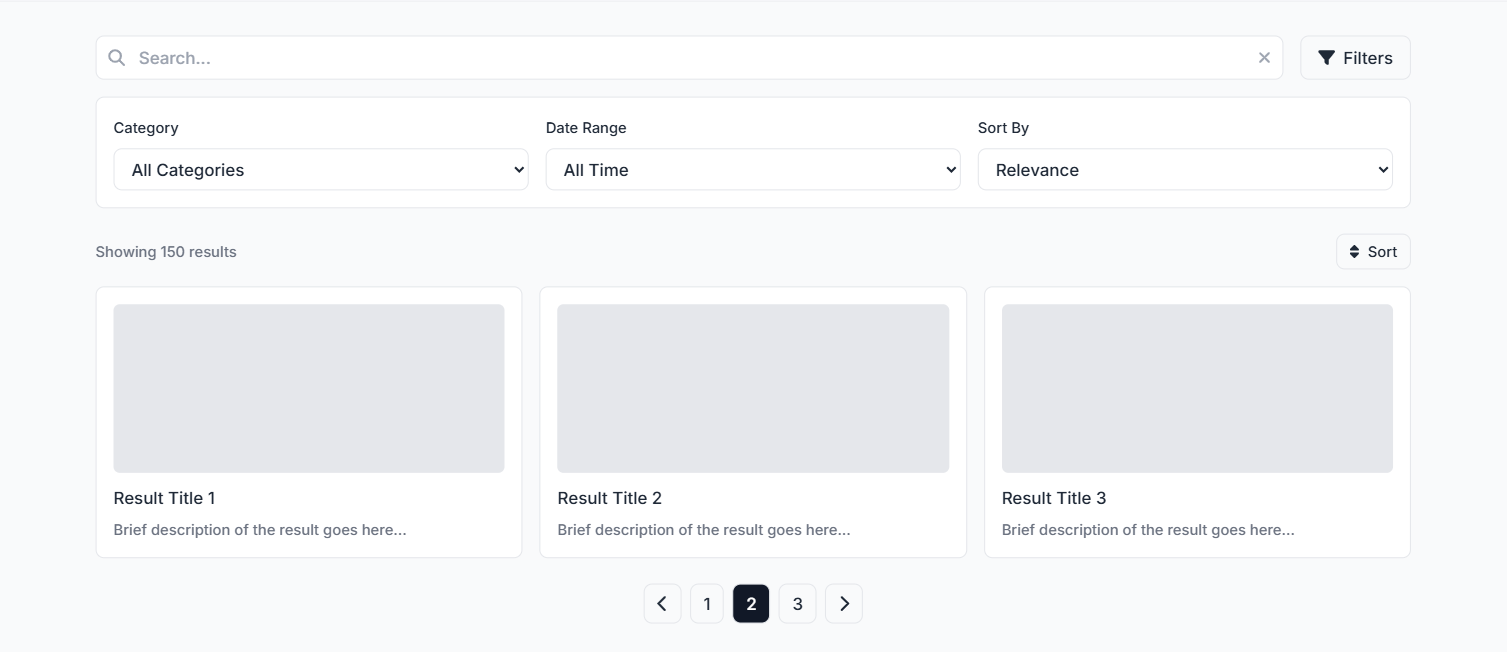
The **Task Submissions** function allows Hunters to submit completed tasks, track their status, and receive rewards upon approval.

* The user navigates to the task submission section.
* The user selects or uploads the required files.
* The user enters any necessary task details (e.g., description, notes).
* The user submits the task.
* The system processes the submission and updates the status.
* The user can view the status of their submitted tasks.
* The user can edit or delete the submission if allowed.
* If the user does not complete the submission, the system may save it as a draft.
* If the Bounty Creator approved the submission, the Hunter will be rewarded with the bounty price
* If the Bounty Creator rejects the submission, the submission will be marked as rejected.

#### 1.4 Search and Filter System

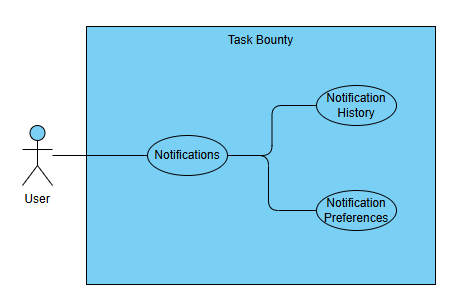
  
Users can discover tasks based on criteria.

* The user accesses the search functionality.
* The user enters a keyword or search query.
* The system processes the search query and retrieves relevant data.
* The user applies filters to refine search results.
* The system updates and displays the filtered results.
* The user can sort the results if needed.
* The user views the final results.
* If no results are found, the system displays a “No results found” message.
* If the user enters an invalid search query, an error message is shown.
* If the system encounters an issue retrieving data, an error message is displayed.



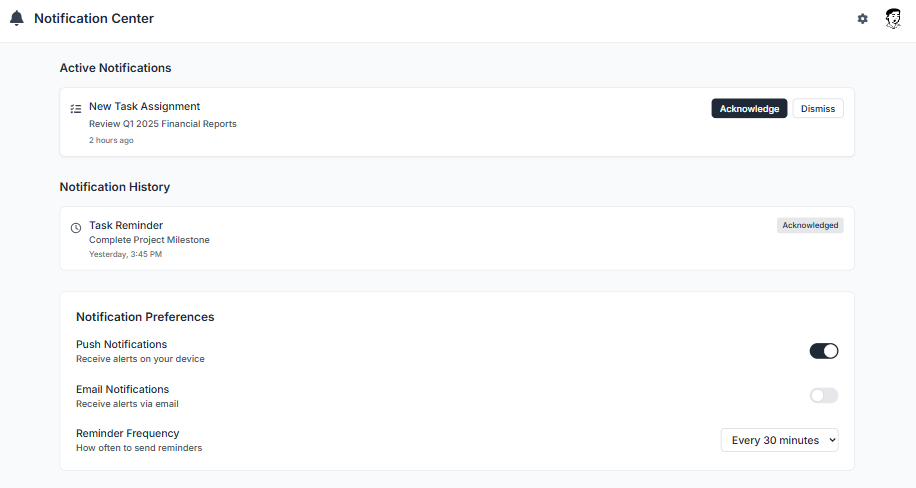
Mobile Application

#### 1.5 Tasks Alerts and Notification

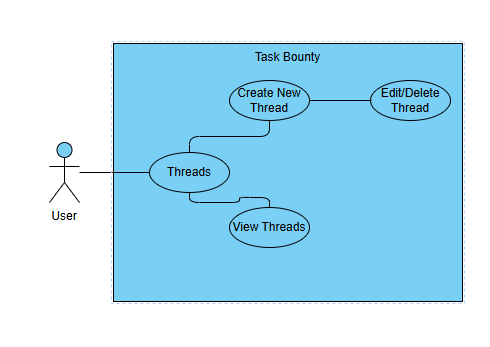


Real-time updates for task-related activities.

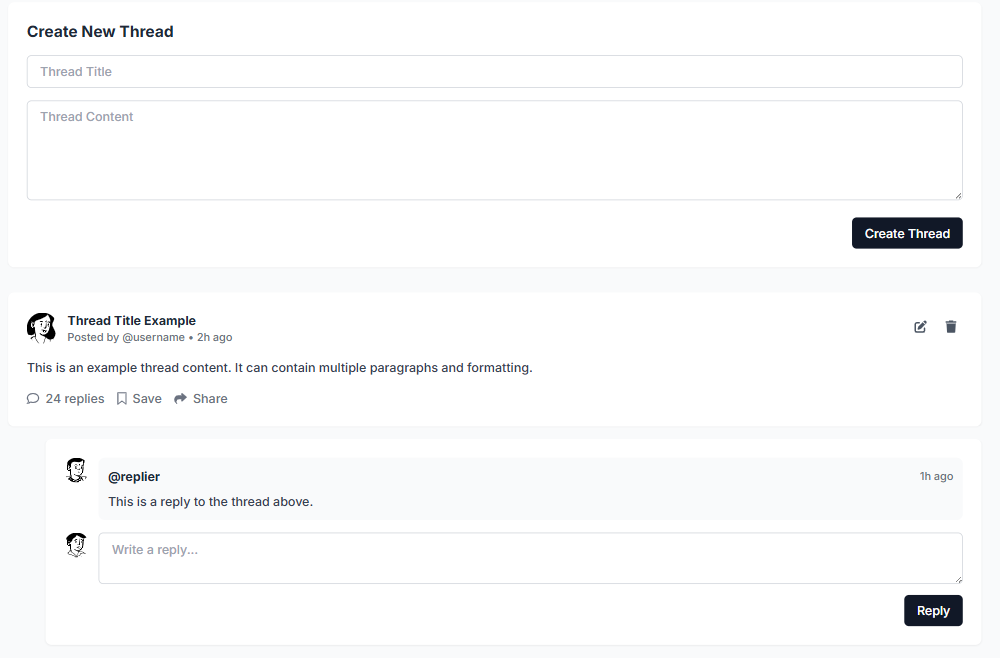
* The system generates a task alert based on predefined triggers.
* The user receives a notification on their mobile device.
* The user views the task details from the notification panel.
* The user acknowledges or dismisses the alert.
* The user can access the notification history for previous alerts.
* The user can modify notification preferences in the settings.
* If notifications are disabled, the system does not send alerts.
* If the user does not acknowledge the alert, a reminder notification is sent.
* If the system fails to send a notification, an error log is recorded.



#### 1.6 Thread System



* The user navigates to the thread system section.
* The user creates a new thread by entering a title and content.
* The system saves the thread and makes it visible to other users.
* The user or other users can view the thread.
* The user or other users can reply to the thread.
* The original poster can edit or delete their thread if necessary.
* If the user attempts to create a thread without content, an error message is displayed.
* If the system fails to save the thread, an error message is logged.
* If a user tries to edit or delete a thread they do not own, an error message is shown.



### Module 2

#### 2.1 Transaction Name

##### Use Case Diagram

##### Use Case Description

##### Activity Diagram

##### Wireframe

#### 2.2 Transaction Name

##### Use Case Diagram

##### Use Case Description

##### Activity Diagram

##### Wireframe

##### . . .

## Non-functional requirements

### Performance

##### Details

### Security

* Details

### Reliability

##### Details