Software Requirements Specification

for

NSTU\_QuickBill

Prepared by

**Dipraj Dhar MUH2125004M**

**Rehnuma Tahiyan BBH2125011F**

**Md. Sajedul Karim Munna ASH2125024M**

**Md. Eshtiak Jamil ASH2125025M**

**Institute of Information Technology**

**Noakhali Science and Technology University**

**05.12.2024**

Contents

[1. Introduction 5](#_Toc168776247)

[1.1 Problem Statement 5](#_Toc168776248)

[1.2 Purpose 6](#_Toc168776249)

[1.3 Project Scope 6](#_Toc168776250)

[1.4 Features 7](#_Toc168776251)

[1.5 Glossary 8](#_Toc168776252)

[1.7 Overview 9](#_Toc168776253)

[2. Stakeholders and Characteristics 9](#_Toc168776254)

[3. Design and Implementation Constrains 12](#_Toc168776255)

[3.1 JavaScript and PhP 13](#_Toc168776256)

[3.1.1 Programming Language 13](#_Toc168776257)

[3.1.2 CSS Framework 13](#_Toc168776258)

[3.2 Server-Side Technology 13](#_Toc168776259)

[3.2.1 Laravel Rest API 14](#_Toc168776260)

[3.2.2 Database Server 14](#_Toc168776261)

[3.2.3 Cloud Storage 14](#_Toc168776262)

[4. Requirement Specification 14](#_Toc168776263)

[4.1 Functional Requirement 15](#_Toc168776264)

[4.1.1 User login and register 15](#_Toc168776265)

[4.1.2 Bill Creation and Submission 15](#_Toc168776266)

[4.1.3 Bill Review and Approval 15](#_Toc168776267)

[4.1.4 Validation by Registrar's Office 16](#_Toc168776268)

[4.1.5 Validation by Director of Academic Affairs (DAA) Office 16](#_Toc168776269)

[4.1.6 Verification by Exam Control Office 16](#_Toc168776270)

[4.1.7 Financial Verification by Treasurer's Office 17](#_Toc168776271)

[4.1.8 Final Approval by Vice-Chancellor (VC) 17](#_Toc168776272)

[4.1.9 Notification System 17](#_Toc168776273)

[4.1.10 Security 17](#_Toc168776274)

[4.1.11 Performance 18](#_Toc168776275)

[4.1.12 Usability 18](#_Toc168776276)

[4.1.13 User Logout 18](#_Toc168776277)

[4.2 Data Requirement 19](#_Toc168776278)

[4.2.1 User Information 19](#_Toc168776279)

[4.2.2 Exam Bill Data 19](#_Toc168776280)

[4.2.3 Department Data 19](#_Toc168776281)

[4.2.4 Audit Logs 19](#_Toc168776282)

[4.2.5 Notification Data 19](#_Toc168776283)

[4.2.6 Presence Verification Data 19](#_Toc168776284)

[4.2.7 Financial Data 19](#_Toc168776285)

[4.3 Performance Requirement 20](#_Toc168776286)

[4.3.1 Response Time 20](#_Toc168776287)

[4.3.2 Concurrent Users 20](#_Toc168776288)

[4.3.3 Transaction 20](#_Toc168776289)

[4.3.4 Availability 20](#_Toc168776290)

[4.3.5 Security 20](#_Toc168776291)

[4.3.6 Data Backup 20](#_Toc168776292)

[4.3.7 Capacity Requirement 20](#_Toc168776293)

[Here are some capacity requirements for our project : 20](#_Toc168776294)

[4.3.2.1 Bandwidth 20](#_Toc168776295)

[4.3.2.2 Storage Capacity 20](#_Toc168776296)

[4.3.2.3 Processing Power 21](#_Toc168776297)

[4.3.2.4 Database Capacity 21](#_Toc168776298)

[4.3.2.5 Server Capacity 21](#_Toc168776299)

[4.3.2.6 Network Capacity 21](#_Toc168776300)

[4.3.2.7 Memory Capacity 21](#_Toc168776301)

[4.3.3 Safety Critical Requirement 21](#_Toc168776302)

[4.3.4 Robustness or Fault-Tolerance Requirements 21](#_Toc168776303)

[4.4 Maintainability and Supportability 22](#_Toc168776304)

[4.4.1 Maintenance Requirements 22](#_Toc168776305)

[4.4.2 Supportability Requirements 23](#_Toc168776306)

[4.5 Security Requirements 23](#_Toc168776307)

[4.5.1 Authentication and Authorization 23](#_Toc168776308)

[4.5.2 Encryption 23](#_Toc168776309)

[4.5.3 Access Control 23](#_Toc168776310)

[4.5.4 Network Security 23](#_Toc168776311)

[4.5.5 Audit Trails 23](#_Toc168776312)

[4.5.6 Data Backup and Recovery 23](#_Toc168776313)

[4.5.7 System Monitoring 24](#_Toc168776314)

[4.5.8 Incident Response 24](#_Toc168776315)

[4.6 Usability and Human Integrity Requirements 24](#_Toc168776316)

[4.6.1 Ease of Use Requirements 24](#_Toc168776317)

[4.6.2 Accessibility Requirements 24](#_Toc168776318)

[4.7 Look and Feel Requirements 25](#_Toc168776319)

[4.7.1 Appearance Requirements 25](#_Toc168776320)

[4.8 Style Requirements 25](#_Toc168776321)

[5. Requirement Engineering Process 26](#_Toc168776322)

[5.1 Requirement Elicitation Techniques 26](#_Toc168776323)

[5.1.1 Hold Interviews 26](#_Toc168776324)

[5.1.2 Perform Document Analysis 26](#_Toc168776325)

[5.1.3 System Interface Analysis 26](#_Toc168776326)

[5.1.4 Distribute Questionnaires 27](#_Toc168776327)

[5.2 Sample of requirement collection 27](#_Toc168776328)

[5.2.1 Requirement collection -1 27](#_Toc168776329)

[5.2.2 Requirement collection -2 28](#_Toc168776330)

[5.3 Requirement Validation 29](#_Toc168776331)

[5.3.1 Review the Requirements 29](#_Toc168776332)

[5.3.2 Test the Requirements 29](#_Toc168776333)

[5.3.3 Simulate the requirements 29](#_Toc168776334)

[6. Use Case Diagram 30](#_Toc168776335)

[7. Use Case Description 31](#_Toc168776336)

[8. Sequence Diagram 55](#_Toc168776337)

[9. Activity Diagram 65](#_Toc168776338)

[10. State Diagram 66](#_Toc168776339)

[11. Swimlane Diagram 67](#_Toc168776340)

[12. Requirement Traceability Matrix 68](#_Toc168776341)

# Introduction

In the current system at Noakhali Science and Technology University (NSTU), the process of creating and approving exam bills for teachers is entirely manual. This method is not only time-consuming but also prone to errors and delays. Each teacher must manually prepare their bill, which then undergoes a multi-step verification process involving several administrative offices, including the Director of their department, the Registrar’s office, the Director of Academic Affairs (DAA) office, Exam control office, the Treasurer’s office, and finally, the Vice-Chancellor (VC) for approval.

To streamline this cumbersome process, we propose the development of an online exam bill management system. This system aims to digitize and automate the entire workflow, making it more efficient, transparent, and less error-prone. With this new system, teachers will be able to create and submit their exam bills online, which will then be electronically routed through each necessary administrative step for validation and approval.

## Problem Statement

The current system for creating and approving exam bills at Noakhali Science and Technology University (NSTU) is entirely manual, leading to significant inefficiencies. Teachers must manually prepare their bills, a process that is not only time-consuming but also highly susceptible to errors and delays. This manual approach involves a multi-step verification process that requires physical submission and handling of documents across several administrative offices, including the Director of the department, the Registrar’s office, the Director of

Academic Affairs (DAA) office, the Accounts section, the Treasurer’s office, and finally, the Vice-Chancellor (VC) for approval. Each of these steps introduces potential bottlenecks and opportunities for errors, resulting in a cumbersome and protracted workflow.

The manual nature of the current system creates several problems:

1. **Time-Consuming:** The preparation and processing of exam bills take an excessive amount of time, delaying the final approval and payment.
2. **Error-Prone:** Manual data entry and document handling increase the likelihood of mistakes, leading to inaccuracies that must be corrected, further delaying the process.
3. **Lack of Transparency:** Tracking the status of bills is difficult, causing uncertainty and confusion among teachers and administrative staff.
4. **Inefficiency:** Multiple physical submissions and verifications create redundant work, reducing overall productivity.

To address these issues, we proposes the development of an online exam bill management system. This new system aims to digitize and automate the entire workflow, from bill creation to final approval, thereby enhancing efficiency, accuracy, and transparency.This transformation is expected to significantly reduce processing time, minimize errors, and provide clear tracking and accountability throughout the entire exam bill management process.

## Purpose

The purpose of developing the NSTU Quick\_Bill System is to modernize and streamline the process of creating, verifying, and approving exam bills at Noakhali Science and Technology University (NSTU). By digitizing and automating the workflow, the system aims to enhance efficiency, reduce errors, and increase transparency, ultimately providing a more effective and user-friendly experience for teachers and administrative staff.

## Project Scope

The scope of the NSTU Quick\_Bill System project includes the following key components and functionalities:

1. **User Interface Development:**
   * Design and implementation of an intuitive, user-friendly web interface for teachers to create and submit exam bills.
   * Development of administrative dashboards for department Directors, Registrar's office staff, DAA office staff, Accounts section, Treasurer’s office, and the Vice-Chancellor (VC) to review, validate, and approve bills.
2. **Bill Creation and Submission:**
   * Digital forms for teachers to enter exam bill details.
   * Automated submission process that routes bills to the appropriate administrative offices.
3. **Verification and Validation Workflow:**
   * Automated routing of submitted bills to department Directors for initial verification.
   * Integration with the Registrar's office system to verify teacher presence in exam halls.
   * Validation checks by the DAA office to ensure accuracy and completeness of bill details.
   * Financial verification by the Accounts section.
4. **Approval Process:**
   * Final fund availability check by the Treasurer’s office.
   * Approval workflow for the VC to grant final permission for bill processing and payment.
5. **Notifications and Alerts:**
   * Automated email and system notifications to inform teachers and administrative staff of bill status changes and required actions.
6. **Tracking and Reporting:**
   * Real-time tracking of bill status through the system interface.
   * Generation of detailed reports for audit and review purposes.
7. **Security and Access Control:**
   * Implementation of secure login and role-based access control to ensure data privacy and integrity.
   * Data encryption and secure data storage to protect sensitive information.
8. **System Integration:**
   * Integration with existing NSTU administrative systems for seamless data exchange and verification.
9. **Testing and Quality Assurance:**
   * Comprehensive testing to ensure functionality, performance, and security.
   * User acceptance testing (UAT) to validate the system meets user requirements.
10. **Training and Support:**
    * Development of user manuals and training sessions for teachers and administrative staff.
    * Provision of ongoing technical support and system maintenance.

By covering these aspects, the NSTU QuickBill System project aims to create a comprehensive, efficient, and reliable online platform for managing exam bills at NSTU.

## Features

**1.4.1 User Authentication and Authorization:**

Secure login functionally for users with different roles (teachers, officials, directors of departments, director of Accounts and audit, bank officials). Role-based access control to restrict access to specific functionalities based on user roles and permissions.

**1.4.2 Bill submission and Management:**

Ability for teachers to submit bills electronically through the website. Form fields for entering bill details such as amount, purpose, and supporting documentation upload. option to save bill drafts and submit them later.

**1.4.3 Approval Workflow:**

Multi-stage approval process involving officials, directors of departments, directors of accounts and audits, and bank officials. Workflow automation to route bills to the appropriate approves based on predefined criteria. Notifications to inform users of pending approvals and deadlines.

**1.4.4 Document Management and Storage:**

Centralized repository for storing bill documents securely. Version control to track revisions and updates of bills. Search and retrieval functionalities for easy access to achieved bills.

**1.4.5 Billing History and Tracking:**

Log of all bill submissions, approvals, rejections, and modifications. Audit trail to track the history of each bill and its status changes over time. Reporting tools to generate summaries and detailed reports on billing activities.

**1.4.6 Financial Transactions:**

Integration with the university’s financial systems for fund transfer and accounting. Secure payment gateway for bank officials to transfer funds to teachers’ accounts. Tracking of payment statuses and reconciliation then with bill approvals.

**1.4.7 Notifications and Alerts:**

Automated notifications to alert users of pending bills, approvals, and payment status. Customizable notification preference for users to choose their preferred communication channels (e-mail, SMS, etc.).

**1.4.8 User interface and Usability:**

Intuitive and user-friendly interface accessible from desktop and mobile devices. Responsive design to ensure optimal usability across various screen sizes and resolutions. Accessibility features to accommodate users with disabilities.

**1.4.9 Security and Compliance:**

Data encryption to protect sensitive information during transmission and storage. Regular security audits to vulnerability assessments to identify and mitigate risks.

**1.4.10 Electronic Signatures:**

Electronic signature functionality to authenticate approvals and acknowledgements securely. Integration with digital signature platforms for legally binding signature on bill documents.

## Glossary

This section provides definitions for all document names, acronyms, and abbreviations. The application domain's terms and concepts are defined.

GUI - Graphical User Interface

API – Application Programming Interface

SRS – Software Requirement Specification

UI – User Interface

SDLC – Software Development Life Cycle

MB – Megabytes

XML – Extensible Markup Language

RESTful – Representational State Transfer

HTML – Hyper Text Markup Language

JS – Java Script

CSS – Cascade Style Sheet

**1.6 Reference**

* IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.
* Elsievier-*A systematic literature review on agile requirements engineering practices and challenges .*
* Bjarnason, E., Wnuk, K., & Regnell, B. (2011b*). Requirements are slipping through the gaps—A case study on causes & effects of communication gaps in large-scale software development*. In 2011 IEEE 19th international requirements engineering conference (pp. 37–46)
* Wolfgang, E. (2011). *Working with user stories. In Agile requirements engineering workshop*, July 2011.
* Bjarnason, E., Wnuk, K., & Regnell, B. (2011b). *Requirements are slipping through the gaps—A case study on causes & effects of communication gaps in large-scale software development*. In 2011 IEEE 19th international requirements engineering conference (pp. 37–46).

## 1.7 Overview

The NSTU QuickBill System is an innovative online platform designed to revolutionize the manual process of creating and approving exam bills at Noakhali Science and Technology University (NSTU). Currently, the manual system is labor-intensive, error-prone, and time-consuming, involving multiple administrative steps and physical document handling. This project aims to digitize and automate the entire workflow, making it more efficient, transparent, and less susceptible to errors. Teachers will use an intuitive online interface to create and submit their exam bills, eliminating the need for handwritten documents. Once submitted, bills are automatically routed to the relevant department Director for initial verification, followed by the Registrar's office to confirm teacher presence in exam halls. The Director of Academic Affairs (DAA) office will then review and validate the bill details, ensuring accuracy and completeness, while the Accounts section performs a final financial review. The Treasurer's office will verify the availability of funds before the bill is forwarded to the Vice-Chancellor (VC) for final approval and processing.

The NSTU QuickBill System will significantly enhance efficiency by reducing the time required for bill processing through automated routing and validation steps. Digital data entry and automated checks will minimize the risk of errors, while real-time tracking and notifications will keep all stakeholders informed about the status of bills throughout the workflow. The user-friendly interface is designed to be accessible and easy to use for both teachers and administrative staff. The project scope includes developing user interfaces for bill creation and administrative processing, integrating with existing NSTU systems, implementing security measures, and providing training and support for users along with ongoing system maintenance. By streamlining the exam bill process, the NSTU QuickBill System aims to save time and resources, enhance accuracy and reliability, improve transparency and accountability, and provide a modern, efficient solution that meets the needs of NSTU’s faculty and staff.

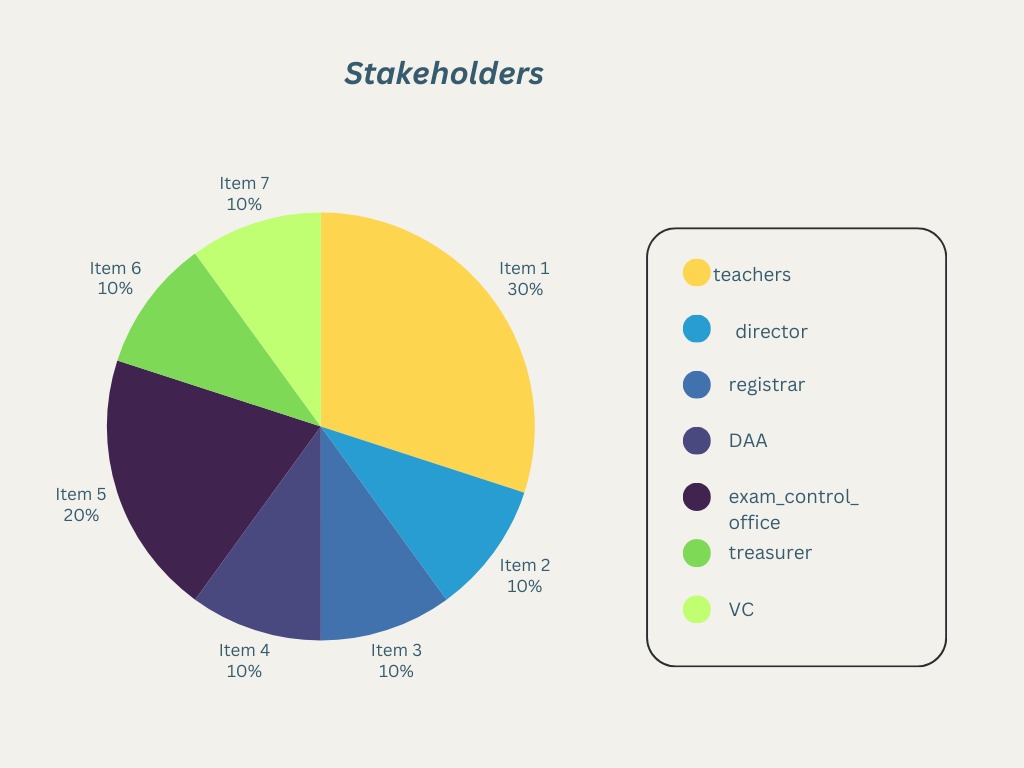
# Stakeholders and Characteristics

**Primary Stakeholders:**

1. **Teachers:**
   * **Role:** Teachers are the primary users of the NSTU QuickBill System, responsible for creating and submitting exam bills that detail their involvement in exam invigilation and related duties.
   * **Impact:** The system will simplify their billing process, reduce the time and effort required for manual bill preparation, and ensure more timely and accurate compensation.
2. **Director:**
   * **Role:** Directors head the academic departments and are responsible for the initial verification of the exam bills submitted by teachers in their department.
   * **Impact:** They ensure that the submitted bills are accurate and valid before forwarding them for further processing, reducing errors and ensuring compliance with departmental policies.

**Secondary Stakeholder:**

1. **Administrative Staff:**
   * **Role:** Support various administrative functions related to exam bill processing, such as data entry, document management, and communication between departments.
   * **Impact:** Facilitate smooth workflow and communication, ensuring that bills move efficiently through the verification and approval stages.
2. **IT Support Team:**
   * **Role:** Responsible for maintaining the technical infrastructure of the NSTU QuickBill System, including troubleshooting issues, performing updates, and ensuring data security.
   * **Impact:** Ensures the system is reliable, secure, and user-friendly, providing necessary technical support to teachers and administrative staff.
3. **Finance Department:**
   * **Role:** Works closely with the Accounts section and Treasurer’s office to manage financial transactions related to exam bill payments, ensuring compliance with financial policies.
   * **Impact:** Ensures accurate financial management and reporting, supporting the financial integrity of the exam bill process.
4. **University Management:**
   * **Role:** Involves senior administrative and management personnel who oversee the implementation and operational efficiency of the NSTU QuickBill System.
   * **Impact:** Ensures the system aligns with the university’s strategic objectives, fostering a culture of efficiency, transparency, and accountability.
5. **Students:**
   * **Role:** Indirect stakeholders who benefit from the streamlined administrative processes, as efficient exam bill processing contributes to a well-organized academic environment.
   * **Impact:** Although not directly involved in the bill process, students benefit from improved administrative efficiency, which can lead to more effective exam management and better overall academic experiences.



# Design and Implementation Constrains

We have employed design and implementation constraints to ensure the success of this project. It also refers to a tool that allows developers and testers to inspect and interact with the application's user interface (UI) elements.

The design and implementation constraints for the NSTU\_QuickBill platform:

**Scalability:** The platform needs to be designed to handle a large number of users, transactions, and data. It should be able to scale easily and efficiently as the user base and demand grow.

**Security:** The platform needs to be secure to protect sensitive information such as user data, financial information, and transaction details. This includes measures such as data encryption, secure payment gateways, and secure user authentication.

**Availability:** The platform needs to be highly available and reliable to ensure users can access the platform and its services at all times. This includes measures such as redundancy, load balancing, and disaster recovery.

**User Experience:** The platform needs to provide a seamless and intuitive user experience for all stakeholders, including teachers, directors, registrar, Daa\_office, exam control office, and terasurer. This includes features such as easy navigation, responsive design, and personalized recommendations.

**Data Management:** The platform needs to be designed to manage and store large amounts of data related to users, products, transactions, and other relevant information. This includes measures such as database design, data security, and data backup.

**Integration:** The platform needs to integrate with various third-party services such as payment gateways, logistics providers, and other relevant services. This requires an API-based architecture that can easily integrate with other systems.

**Technology Stack:** The platform needs to be built using appropriate technology stacks that can support its features and requirements. This may include programming languages such as Php, JavaScript, frameworks such as Laravel and Node.js, and databases such as MySQL.

**Testing:** The platform needs to be thoroughly tested to ensure its functionality, security, and performance. This includes unit testing, integration testing, and acceptance testing.

**Compliance:** The platform needs to comply with relevant laws and regulations related to data protection, financial transactions, and other relevant areas. This includes measures such as GDPR compliance, PCI DSS compliance, and other relevant regulations.

These are the key design and implementation constraints for the NSTU\_QuickBill platform.

## JavaScript and PhP

The visual layout of the components that a user could interact with in a website or technical product is referred to as user interface design, or UI design. In other terms, it is a website's visual design.

### Programming Language

**JavaScript:**

Javascript is an ECMAScript-compliant high-level, frequently just-in-time compiled language. It has first-class functions, dynamic typing, and prototype-based object orientation. It's multi-paradigm, allowing you to program in event-driven, functional, or imperative styles.

React is a front-end JavaScript toolkit for creating user interfaces using UI components that is free and open-source. Meta and a community of individual developers and businesses support it.

JavaScript XML is abbreviated as JSX. It's just a JavaScript syntactic extension. It allows us to create HTML directly in React (within JavaScript code). It is straightforward to generate a template in React using JSX, but it is not a simple template language; instead, it has all of JavaScript's capability.

**PHP:** PHP is ideal for developing NSTU\_quickbill due to its ease of use, wide adoption, and extensive community support. It offers robust frameworks, seamless database integration, and strong security features. PHP ensures efficient performance, compatibility with various web servers, and scalability, making it a reliable choice for creating dynamic, user-friendly web applications.

### CSS Framework

Cascading Style Sheets (CSS) is a language for specifying the appearance of a document written in a markup language like HTML. Along with HTML and JavaScript, CSS is a key component of the World Wide Web. Semantic UI is a website using UI component framework. Developers may use Semantic UI to create websites with quick and clear HTML, as well as a fully mobile responsive experience. Semantic UI offers a React-integrated version called Semantic UI React, which includes the following functionalities:

* jQuery Free.
* Declarative API.
* Augmentation.
* Shorthand Props.
* Sub Components.
* Auto Controlled State

## Server-Side Technology

Server-side development refers to the actions that take place behind the scenes when an application is used. It primarily focuses on databases, scripting, website architecture, backend logic, APIs, and Servers.

### Database Server

MySQL is a renowned open-source relational database management system prized for its reliability, performance, and user-friendly interface. Widely employed in web development, MySQL excels in efficiently storing and managing structured data, boasting features such as scalability, robust security, and seamless integration with various applications. Backed by a vibrant community, MySQL remains a preferred choice for a spectrum of projects, from small-scale websites to enterprise-level applications, where its stability, speed, and ease of use ensure optimal data management and performance. It is a dependable, powerful, and stable solution with sophisticated features such as the following:

* User-defined types.
* Table inheritance.
* Sophisticated locking mechanism.
* Foreign key referential integrity.
* Views, rules, subquery.
* Nested transactions (save points)
* Multi-version concurrency control (MVCC)
* Asynchronous replication.

### Cloud Storage

Amazon S3 is a type of object storage that allows to store and retrieve any quantity of data from any location. It's a straightforward storage solution with industry-leading durability, availability, performance, security, and scalability.

# Requirement Specification

All the requirements based on the elicitation process are described in this section.

## Functional Requirement

Functional requirements are those requirements that are used to illustrate the internal working nature of the system, the description of the system, and explanation of each subsystem. It consists of what task the system should perform, the processes involved, which data the system should hold and the interfaces with the user.

### User login and register

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-01** | User Registration and Login to registered account. | | |
| **Description** | User should register his/her account for the first time and be able to login to the account which was registered once. Already registered users will not face this stage. | | |
| **Stakeholders** | All the users | **Priority** | High |

### Bill Creation and Submission

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-02** | Bill Creation and Submission | | |
| **Description** | Teachers create exam bills online, providing necessary details such as exam date, course code, and amount. | | |
| **Stakeholders** | Teachers | **Priority** | Medium |

### Bill Review and Approval

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-03** | Bill Review and Approval | | |
| **Description** | Department directors review and approve submitted bills before forwarding them to the Registrar's office for further validation. | | |
| **Stakeholders** | Department Directors | **Priority** | Medium |

### Exam Panel

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-04** | Exam panel | | |
| **Description** | This is designed to streamline the creation, organization, and management of exam committees. This functionality ensures proper coordination and efficiency in handling academic evaluations and examination. | | |
| **Stakeholders** | Teachers (Examiners) | **Priority** | High |

### Exam Committee Management

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-05** | Exam Committee Management | | |
| **Description** | Exam Committee Management enables the Director or relevant authority to create and manage exam committees by assigning roles such as Chairman and Members for specific sessions, terms, and batches. | | |
| **Stakeholders** | Committee Chairman, Committee Members | **Priority** | High |

### The Verification System

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-06** | The Verification System | | |
| **Description** | The Verification System ensures accurate validation of key processes, such as verifying teacher presence, bill details, fund availability, and committee assignments. It incorporates multiple levels of checks by authorized stakeholders, ensuring compliance and reliability at every stage. | | |
| **Stakeholders** | Teachers,members,director | **Priority** | Medium |

### Security

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-10** | Security | | |
| **Description** | The system must implement strong encryption mechanisms to protect user data and ensure secure transmission of sensitive information. | | |
| **Stakeholders** | All Users | **Priority** | Medium |

### Performance

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-11** | Performance | | |
| **Description** | The system should handle concurrent user requests efficiently, ensuring minimal downtime and fast response times even during peak usage periods. | | |
| **Stakeholders** | All Users | **Priority** | High |

### Usability

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-12** | Usability | | |
| **Description** | The user interface should be intuitive and user-friendly, with clear navigation and well-designed forms for submitting bills and managing user profiles. | | |
| **Stakeholders** | All users | **Priority** | Medium |

### 4.1.13 User Logout

|  |  |  |  |
| --- | --- | --- | --- |
| **FR-13** | User logout from their account | | |
| **Description** | The system will provide a logout button, manage user sessions, provide a confirmation message, clear the user's session. | | |
| **Stakeholders** | All users | **Priority** | Low |

## Data Requirement

Based on the description of our project, it seems that our system would require several types of data to function effectively. Here are some potential data requirements :

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement Name** | **Requirement Description** | **Stakeholders** | **Priority** |
| User Information | The system must store information about all users, including teachers, administrative staff, and directors. | Teachers, Administrative Staff, Directors | High |
| Exam Bill Data | The system must store details about each exam bill submitted by teachers. | Teachers, Department Directors, Registrar's Office, DAA Office, Exam Control Office, Treasurer's Office, Vice-Chancellor. | High |
| Department Data | Information about various departments within the university. | Teachers, Department Directors, Registrar's Office | High |
| Audit Logs | Logs of all significant actions performed in the system to ensure traceability and accountability. | All Users | High |
| Notification Data | Details of notifications sent to users. | Teachers, Department Directors, Registrar's Office, DAA Office, Exam Control Office, Treasurer's Office, Vice-Chancellor. | High |
| Presence Verification Data | Data to verify teacher presence in exam halls. | Teachers, Registrar's Office | High |
| Financial Data | Financial information related to bill payments and fund availability. | Treasurer's Office, Teachers, Directors | High |

In addition to the above, our system would also require data on the usage of the platform, such as the number of users, active sessions, and engagement metrics, to help you understand how the system is being used and to make improvements over time. It's important to ensure that all the data is collected ethically, stored securely, and used in compliance with privacy regulations.

## Performance Requirement

It is important to maintain the performance of the software system. To ensure performance we maintain these steps:

| **Requirement Name** | | **Requirement Description** | **Stakeholders** | | **Priority** |
| --- | --- | --- | --- | --- | --- |
| Response Time | | The system should respond to user actions (e.g., bill submission, profile updates) within a specified time frame to ensure a smooth user experience. | All users | | High |
| Concurrent Users | | The system must support a high number of concurrent users without performance degradation. | All users | | High |
| Transaction | | The system should be able to process a large number of transactions per minute without any issues. | All users | | High |
| Availability | The system must have an uptime of 99.99% to ensure users can access it at any time. | | | IT, Management | High |
| Security | The system must ensure that customer data is secure and protected from unauthorized access or data breaches. | | | IT Team | High |
| Data Backup | The system must regularly back up data to prevent data loss in case of hardware or software failures. | | | IT team | Medium |

### Capacity Requirement

### Here are some capacity requirements for our project :

### 4.3.2.1 Bandwidth

The system must have sufficient bandwidth to support high traffic volume during peak periods, such as holiday seasons or promotional events.

### 4.3.2.2 Storage Capacity

The system must have sufficient storage capacity to store product information, customer data, and order history.

### 4.3.2.3 Processing Power

The system must have sufficient processing power to handle large volumes of data, including transactions, searches, and analytics.

### 4.3.2.4 Database Capacity

The system must have sufficient database capacity to store customer information, transaction data, and inventory information.

### 4.3.2.5 Server Capacity

The system must have sufficient server capacity to handle user requests and ensure that the system runs smoothly.

### 4.3.2.6 Network Capacity

The system must have sufficient network capacity to support multiple user connections and ensure that the system is accessible from different locations.

### 4.3.2.7 Memory Capacity

The system must have sufficient memory capacity to handle multiple transactions and user requests without any delay or interruption.

### 4.3.3 Safety Critical Requirement

There are no safety critical requirements for our project.

### Robustness or Fault-Tolerance Requirements

#### Error Handling

The system must have proper error handling mechanisms in place to handle unexpected errors or exceptions and prevent the system from crashing or becoming unresponsive.

#### Redundancy

The system must have redundant components, such as servers, databases, or network connections, to ensure that critical functions are available even in case of failures.

#### Failover

The system must have a failover mechanism that automatically switches to a backup system or component in case of a failure to ensure that the system remains operational.

#### Load Balancing

The system must have load balancing mechanisms that distribute the load across multiple servers to ensure that no single server is overwhelmed with requests and that the system remains responsive.

#### Performance Monitoring

The system must have performance monitoring mechanisms that track system performance and alert administrators if the system falls below defined thresholds.

#### Recovery Time Objectives (RTO)

The system must have defined RTOs for each critical component or system function, specifying the maximum acceptable downtime and the required recovery time in case of a failure.

#### Backups

The system must have a backup mechanism that regularly backs up critical data to ensure that data can be recovered in case of a catastrophic failure

## Maintainability and Supportability

### Maintenance Requirements

#### Regular Software Updates

The system must have a mechanism to update the software to the latest version, which includes bug fixes, performance improvements, and security enhancements.

#### Regular Hardware Maintenance

The system must have regular hardware maintenance, which includes cleaning, inspection, and repair or replacement of faulty components.

#### Data Backup and Recovery

The system must have a regular data backup mechanism to ensure that critical data is not lost in case of hardware or software failures.

#### System Monitoring

The system must have a system monitoring mechanism that tracks system performance, usage patterns, and error logs to identify potential issues and optimize system performance.

#### User Support

The system must have a user support mechanism to help users troubleshoot issues, provide guidance, and resolve user complaints or issues.

#### Training

The system must provide training to administrators and users to ensure that they are familiar with the system's features and capabilities, and can use the system effectively.

#### Documentation

The system must have documentation that outlines the system architecture, design, and usage, which is helpful for troubleshooting, maintenance, and training purposes.

### Supportability Requirements

This system meets Testability, Maintainability, Compatibility, Configurability, Serviceability, and install ability which are related to supportability requirements.

## Security Requirements

Securing information is much more important for our system to get users dependability. Here are some of them:

### Authentication and Authorization

The system must have a mechanism to authenticate users and authorize access to system resources based on the user's role and level of access.

### Encryption

The system must use encryption mechanisms to secure sensitive data, such as login credentials, financial data, and personal information, during transmission and storage.

### Access Control

The system must have access control mechanisms to restrict access to sensitive data and system resources to authorized personnel only.

### Network Security

The system must have network security mechanisms, such as firewalls and intrusion detection systems, to protect against network attacks and unauthorized access to the system.

### Audit Trails

The system must have audit trail mechanisms that log user activity and system events, to enable the identification and investigation of security breaches and unauthorized access attempts.

### Data Backup and Recovery

The system must have a data backup and recovery mechanism to ensure that critical data can be restored in case of data loss or corruption.

### System Monitoring

The system must have system monitoring mechanisms that detect and alert administrators of any suspicious or malicious activity on the system.

### Incident Response

The system must have an incident response plan in place to handle security breaches or attacks and minimize the impact of such incidents.

## Usability and Human Integrity Requirements

This system will provide more user-friendly environment

### Ease of Use Requirements

Our system will be easier to use by any type of people and they don’t need any training to use the system.

### Accessibility Requirements

To get access to the application, the application provides authorization/authentication. This application will use various modules.

#### Navigation

The system must have a clear and consistent navigation mechanism, which includes keyboard shortcuts, to help users navigate the system effectively.

#### Text Size and Font

The system must have options to increase or decrease the text size and change the font style to help users with visual impairments read the content easily.

#### Color Contrast

The system must have sufficient color contrast between the background and foreground elements to help users with visual impairments distinguish between different elements on the screen.

#### Alternative Text

The system must provide alternative text for images, videos, and other multimedia elements to help users with visual impairments understand the content.

#### Audio and Video Transcripts

The system must provide transcripts for audio and video content to help users with hearing impairments understand the content.

#### Forms and Input Fields

The system must have accessible forms and input fields, which include labels, hints, and error messages, to help users with disabilities fill out the forms accurately.

#### Assistive Technology Compatibility

The system must be compatible with assistive technologies, such as screen readers, text-to-speech software, and voice recognition tools, to help users with disabilities access the system effectively.

**4.6.2.8** Standards Compliance

The system must comply with accessibility standards, such as Web Content Accessibility Guidelines (WCAG) 2.1, to ensure that the system is accessible to users with disabilities.

## Look and Feel Requirements

Look and feel requirements mainly refer to how the system will look.

### Appearance Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **AR-1** | Text color and font | | |
| **Description** | Our system has to be different and attractive from other existing system using a better look and feel. | | |
| **Stakeholders** | Developer, Customer, Farmer | **Priority** | High |

## Style Requirements

There are no style requirements in our system.

**4.8** Legal Requirements

Legal requirements normally refer to the terms and conditions or privacy policy of any organization. The terms and condition of our application is that, no third-party software or person is allowed to use our data for their business purpose.

# Requirement Engineering Process

Requirements Engineering (RE) determines software requirements according to customer requirements or needs. Requirements engineering process includes requirements elicitation, needs modeling, requirements analysis, requirements assurance & validation, and requirements management.

## Requirement Elicitation Techniques

Requirements elicitation is the practice of researching and finding system requirements for users, customers, and other stakeholders, also referred to as "requirement gathering". Requirement elicitation can be done by contacting participants directly or by doing some research, analysis and testing.

### H**old Interviews**

We hold discussions that can be held individually or with a small group of participants. They are an effective way to access services without spending a lot of time with participants because we meet with people to discuss only certain important requirements of this program. Negotiations are useful for obtaining individual requirements for members in organizing workshops where those members of the program come together to resolve any issues or conflicts. We mainly perform our interview based on some specific criteria.

* Short description about goals and objectives
* Registration process
* Searching Audio Files
* Storage system of each account
* Compression size of audio files

### Perform Document Analysis

Existing documentation can help to show how systems are currently operating or what they are what I should do. Documents include written information about current programs, business processes, needs specifications, and competitor research. Review once textual analysis can help determine which performance should remain and functionality that isn’t in use. After existing document

In analysis, we found several problems with the existing system.

* Existing systems cannot perform file compression.
* A user cannot share a file with others.
* No cloud storage system is provided by the existing systems.

### System Interface Analysis

The first thing to do is to identify which systems the system-to-be shall communicate with. It could be a server on the Internet, a piece of software on the same host as the system-to-be, some hardware or something completely different.

### Distribute Questionnaires

The questionnaire is a useful way to investigate styles, changes in attitudes and users' ideas, and user satisfaction with priorities and preferences. Our lists of questions were as short as possible. The respondent may be tired or frustrated. Had a basic reason for all the questions as well as group the topic areas together for the respondent to focus on. The main advantage of this survey responses was that they were collected in the usual way. Information was summarized by a large number of people.

## Sample of requirement collection

### Requirement collection -1

This report summarizes the results of the questionnaires distribution conducted to gather requirements for our system. The objective of the surveys was to identify the key needs and expectations of the stakeholders and to use this information to develop a comprehensive set of requirements for the system.

**Interview**

**Methodology :**

The interviews were conducted with stakeholders from different department of NSTU.The interviews were conducted in a one-to-one format, lasting approximately 7-10 minutes each.

**Participants :**

Teachers, Department Directors, Registrar’s Office Staff, DAA Office Staff, Exam Control Office Staff, Treasurer’s Office Staff, Vice-Chancellor, System Administrators.

**Findings:**

The following are the key findings from the interviews:

* Web Application is more comfortable.
* Bills will be displayed and can be viewed with description with logging in the system.
* User can search in the search Box.
* User can filter based on category.
* User can submit feedback.
* User can examine their previous bills.
* User can Track Order.

**Key Requirements:**

Based on the findings from the interviews, the following are the key requirements for NSTU\_Quick\_Bill System :

* User-friendly interface with easy navigation.
* Ability to search for bills.
* Filter bills based on category.
* Track bill.
* Submit Feedback

**Assumptions:**

It was assumed during the survey process that the NSTU\_QuickBill system will be accessible via the web and we have to develop an android app for the system.

**Limitations:**

Responses were collected less than the expectation.

**Conclusion:**

The stakeholders’ questionnaires surveys are provided valuable insights into the requirements for the NSTU\_QuickBill system. The key findings and requirements will be used to develop a comprehensive set of requirements for the system.

### Requirement collection -2

This report summarizes the results of the stakeholder interviews conducted to gather requirements for our system. The objective of the interviews was to identify the key needs and expectations of the stakeholders and to use this information to develop a comprehensive set of requirements for the system.

**Distribute Questionnaires**

**Methodology :**

The Questionnaires were distributed to stakeholders from maximum department. It takes 3-4 minutes to complete the questionnaire.

**Participants :**

Teachers, Department Directors, Registrar’s Office Staff, DAA Office Staff, Exam Control Office Staff, Treasurer’s Office Staff, Vice-Chancellor, System Administrators.

**Findings:**

The following are the key findings from the interviews:

* Web Application is more comfortable.
* Bills will be displayed and can be viewed with description with logging in the system.
* User can search in the search Box.
* User can filter based on category.
* User can submit feedback.
* User can examine their previous bills.
* User can Track Order.

**Key Requirements:**

Based on the findings from the interviews, the following are the key requirements for NSTU\_QuickBill System:

* User-friendly interface with easy navigation.
* Ability to search for bills.
* Filter bills based on category.
* Track bill.
* Submit Feedback

**Assumptions:**

It was assumed during the interview process that the NSTU\_QuickBill will be accessible via the web and we have to develop an web app for the system.

**Conclusion:**

The stakeholders’ interviews are provided valuable insights into the requirements for the NSTU\_QuickBill system. The key findings and requirements will be used to develop a comprehensive set of requirements for the system.

## Requirement Validation

Requirement validation ensures that the requirements are correct and reflect the quality you want from this program. In the beginning, our requirements looked good but when we read them and tried to work with them they came out having ambiguities and gaps.

### Review the Requirements

Negative peer review, especially the type of rigorous review called evaluation, is unique among the highest quality software processes available. We had a team of reviewers representing different perspectives and carefully examined written needs, analysis models, and related information on disability.

### Test the Requirements

The test creates another view of the requirements. We also performed writing tests regarding assurance of whether the expected performance was found or not. Getting tested by the user needs to document the expected product behavior under specified conditions.

### Simulate the requirements

To stimulate requirements, trading tools are available that we have used to simulate a proposed system in place or to add details of written requirements. The simulation takes prototyping to the next level.

# Use Case Diagram

A diagram of a diagram

Description automatically generated

*Figure 1: Usecase Diagram*

# Use Case Description

**Table-01:**

**UC-01: Create Exam Bill**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Create Exam Bill** | |
| **Goal** | To create and submit an exam bill for further review and approval. | |
| **Precondition** | 1.Teacher is logged into the system.  2.Teacher has the necessary permissions to create an exam bill. | |
| **Post Conditions** | 1.The exam bill is created and forwarded to the Director for review.  2.The bill status is updated to "Pending Review". | |
| **Trigger** | The teacher initiates the action to create a new exam bill. | |
| **Primary Actor** | Teacher | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | Teacher logs into the system. |
| 2. | Teacher selects the option to create a new exam bill. |
| 3. | Teacher inputs necessary details such as exam name, date, number of students, and amount. |
| 4. | Teacher submits the bill for validation. |
| 5. | The system checks the provided details for completeness and correctness. |
| 6. | The system displays a confirmation message to the teacher. |
| **Alternative Flow** | Step | Action |
| 1.A | A1. Incorrect Credentials During Login.  A2: Missing or Invalid Bill Details.  A3: System Failure During Submission.  A4: Session Timeout.  A5: Unauthorized Access |
| **Quality Requirements** | Step | Action |
| 1. | The system must validate all inputs to ensure data integrity. |
| 2. | The system must provide clear and actionable error messages. |
| 3. | Only authorized teachers should be able to create exam bills. |
| 4. | The interface for creating the bill should be user-friendly and  Intitube. |
| 5. | The bill creation and submission process should be completed within a few seconds. |

**Table-02**

**UC-02: Register**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Register** | |
| **Goal** | Allow new users to create an account in the NSTU\_QuickBill system. | |
| **Precondition** | 1.Teacher is logged into the system.  2.Teacher has the necessary permissions to create an exam bill. | |
| **Post Conditions** | 1.The exam bill is created and forwarded to the Director for review.  2.The bill status is updated to "Pending Review". | |
| **Trigger** | User accesses the registration page and chooses to create a new account. | |
| **Primary Actor** | Any user who wants to access the system functionalities.(like teacher,director etc) | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The user accesses the registration page. |
| 2. | The user enters their personal details (e.g., name, email, password, user type). |
| 3. | The user submits the registration form.. |
| 4. | The system validates the entered details.. |
| 5. | The system creates a new user account and stores the details in the database. |
| 6. | The system sends a registration confirmation to the user. |
| **Alternative Flow** | Step | Action |
| 2.A | A1: Missing or Invalid Details:  a. System displays an error message highlighting the missing or invalid details.  b. User corrects the details and resubmits the form. |
| A2: Email Already Registered:  a. System displays an error message indicating the email is already in use.  b. User provides a different email address and resubmits the form. |
| A3: System Failure:  a. System displays an error message indicating a failure in processing the registration. |
| A4: Session Timeout. |
| A5: Unauthorized Access |
| **Quality Requirements** | Step | Action |
| 1. | The system must validate all inputs to ensure data integrity. |
| 2.    3. | The system must provide clear and actionable error messages.  Only authorized teachers should be able to create exam bills. |
| 4. | The interface for creating the bill should be user-friendly and intuitive. |
| 5. | The bill creation and submission process should be completed within a few seconds. |

**Table-03:**

**UC-03: Log in**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Log in** | |
| **Goal** | Authenticate users and grant access to the NSTU\_QuickBill system based on their credentials. | |
| **Precondition** | 1. The user is already registered in the system.  2. The user is already registered in the system. | |
| **Post Conditions** | 1. The user is successfully authenticated and granted access to the system.  2. The user's session is initiated, and appropriate access rights are assigned based on their user type. | |
| **Trigger** | User accesses the login page and chooses to log into the system. | |
| **Primary Actor** | Any registered user (Teacher, Director, Registrar, DAA\_Office, Exam\_Control\_Office, Treasurer, Vice\_Chancellor). | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The user accesses the login page. |
| 2. | The user enters their personal details (e.g., name, email, password, user type). |
| 3. | The user submits the login form.. |
| 4. | The system validates the entered details.. |
| 5. | If the credentials are valid, the system initiates a session for the user. |
| 6. | The system redirects the user to the appropriate dashboard or homepage based on their user type. |
| **Alternative Flow** | Step | Action |
| 3.A | A1: Missing or Invalid Credentials:  a. System displays an error message indicating missing or incorrect credentials.  b. User corrects the details and resubmits the form. |
| A2: Account Locked:  a. System displays an error message indicating the account is locked due to multiple failed login attempts.  b. User follows the process to unlock their account or contacts support. |
| A3: System Failure:  a. System displays an error message indicating a failure in processing the registration. |
| **Quality Requirements** | Step | Action |
| 1. | The login form should be user-friendly and secure. |
| 2. | Error messages should be clear and help the user correct any issues. |
| 3. | The system should protect against common security threats such as brute force attacks. |
| 4. | The login process should be quick and responsive to provide a good user experience. |

**Table-04**

**UC-04: Edit Profile**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Edit Profile** | |
| **Goal** | Allow users to update their personal information and preferences in the NSTU\_QuickBill system. | |
| **Precondition** | 1. The user must be logged into the system.  2. The user has access to the "Edit Profile" functionality. | |
| **Post Conditions** | 1. The user's profile information is updated in the system.  2. A confirmation message is displayed to the user.  3. The updated information is stored securely in the database. | |
| **Trigger** | The user decides to update their profile information and accesses the "Edit Profile" page. | |
| **Primary Actor** | Any registered user (Teacher, Director, Registrar, DAA\_Office, Exam\_Control\_Office, Treasurer, Vice\_Chancellor). | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The user navigates to the "Edit Profile" page. |
| 2. | The user enters their personal details (e.g., name, email, password, user type). |
| 3. | The user modifies the necessary profile details (e.g., name, email, password, contact information). |
| 4. | The user submits the changes. |
| 5. | The system validates the input data. |
| 6. | If the data is valid, the system updates the user's profile information in the database. |
| **Alternative Flow** | Step | Action |
| 4.A | A1: Invalid Input Data:  a. The system detects invalid or incomplete data  b. The system displays an error message specifying the issues.  c. The user corrects the data and resubmits the form. |
| A2: System Failure:  a. The system encounters an error while updating the profile.  b. The system displays an error message.  c. The user retries the operation or contacts support. |
| **Success End Condition** | 1. | The user's profile information is successfully updated in the system. |
| **Failed End Condition** | 1. | The user's profile information is not updated due to invalid input or system failure. |
| **Quality Requirements** | Step | Action |
| 1. | The profile edit form should be user-friendly and intuitive. |
| 2. | Error messages should be clear and help the user correct any issues. |
| 3. | The system should ensure data integrity and security during the profile update process. |
| 4. | The profile update process should be quick and responsive to provide a good user experience. |

**Table-05:**

**UC-05: Submit Bill**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Submit Bill** | |
| **Goal** | Allow teachers to submit a new exam bill to the NSTU\_QuickBill system for review and approval. | |
| **Precondition** | 1 The teacher must be logged into the system.  2. The teacher must have completed an exam for which they are submitting the bill.. | |
| **Post Conditions** | 1. The exam bill is submitted and stored in the system.  2. The bill is forwarded to the Director for review.  3. A confirmation message is displayed to the teacher. | |
| **Trigger** | The teacher completes an exam and decides to submit a bill for the work done. | |
| **Primary Actor** | Teacher | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The teacher logs into the system. |
| 2. | The teacher navigates to the "Create Exam Bill" page. |
| 3. | The system displays a form for entering bill details. |
| 4. | The teacher fills in the required details (e.g., exam name, date, bill amount).. |
| 5. | The teacher submits the bill. |
| 6. | The system validates the entered data. |
| 7.  8. | If the data is valid, the system stores the bill in the database.  The system updates the bill status to "Pending Review". |
| 9. | The system forwards the bill to the Director for review. |
| 10. | The system displays a confirmation message to the teacher. |
| **Alternative Flow** | Step | Action |
| 5.A | A1: Invalid Input Data:  a. The system detects invalid or incomplete data  b. The system displays an error message specifying the issues.  c. The user corrects the data and resubmits the form. |
| A2: System Failure:  a. The system encounters an error while updating the profile.  b. The system displays an error message.  c. The user retries the operation or contacts support. |
| **Success End Condition** | 1. | The exam bill is successfully submitted and forwarded to the Director for review. |
| **Failed End Condition** | 1. | The exam bill is not submitted due to invalid input or system failure. |
| **Quality Requirements** | Step | Action |
| 1. | The bill submission form should be user-friendly and intuitive.. |
| 2. | Error messages should be clear and help the teacher correct any issues. |
| 3. | The system should ensure data integrity and security during the bill submission process. |
| 4. | The bill submission process should be quick and responsive to provide a good user experience. |

**Table-06**

**UC-06: Enter Bill Details**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Enter Bill Deatils** | |
| **Goal** | Allow teachers to input the necessary details for creating an exam bill. | |
| **Precondition** | 1. The teacher must be logged into the system.  2. The teacher must have completed an exam for which they are submitting the bill.. | |
| **Post Condition** | 1. The bill details are entered into the system.  2. The system temporarily stores the entered details for review and submission. | |
| **Trigger** | The teacher decides to create a new exam bill after completing an exam. | |
| **Primary Actor** | Teacher | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The teacher navigates to the "Create Exam Bill" page. |
| 2. | The system displays a form for entering bill details. |
| 3. | The teacher enters the following details:  - Exam name  - Exam date  - Bill amount  - Any additional comments or notes |
| 4. | The teacher reviews the entered details for accuracy. |
| 5. | The teacher confirms the entered details by clicking on a "Submit" button. |
| 6. | The system validates the entered data. |
| **Alternative Flow** | Step | Action |
| 6.A | A1: Invalid Exam Date:  a.The system detects that the entered exam date is invalid (e.g., in the future or in an incorrect format).  b. The system displays an error message indicating the issue.  c. The teacher corrects the exam date and continues.. |
| A2: Missing Required Fields:  a. The system detects that some required fields are not filled in.  b. The system highlights the missing fields and displays an error message.  c. The teacher fills in the missing information and continues. |
| A3: System Timeout:  a. The system times out due to inactivity.  b. The teacher is prompted to log in again.  c. The previously entered details are either saved as a draft or lost, depending on system design. |
| **Success End Condition** | 1. | The bill details are successfully entered and ready for submission. |
| **Failed End Conditon** | 1. | The bill details are not entered due to validation errors or system issues. |
| **Quality Requirements** | Step | Action |
| 1. | The form should be intuitive and easy to use, with clear labels for each field. |
| 2. | Error messages should be specific and guide the teacher to correct the input. |
| 3. | The system should validate the data in real-time to provide immediate feedback. |

**Table-07:**

**UC-07: Update Bill Status**Top of Form

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Update Bill Status** | |
| **Goal** | To update the status of an exam bill at various stages of the review and approval process. | |
| **Precondition** | 1. The teacher must be logged into the system.  2. The teacher must have completed an exam for which they are submitting the bill.. | |
| **Post Condition** | 1. The bill status is updated to reflect the current stage in the review and approval process.  2. Notifications may be sent to relevant stakeholders. | |
| **Trigger** | The user decides to update the status of an exam bill as part of their role in the review and approval process. | |
| **Primary Actor** | System (interacting with various users: Director, Registrar, DAA\_Office, Exam\_Control\_Office, Treasurer, Vice\_Chancellor) | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The authorized user (Director, Registrar, DAA\_Office, Exam\_Control\_Office, Treasurer, or Vice\_Chancellor) logs into the system. |
| 2. | The user navigates to the "Update Bill Status" section. |
| 3. | The system displays a list of bills pending status updates. |
| 4. | The teacher reviews the entered details for accuracy. |
| 5. | The system displays the current details and status of the selected bill. |
| 6. | The user updates the bill status to one of the following, as appropriate for their role:  a. "Reviewed" (Director)  b. "Teacher Presence Verified" (Registrar)  c. "Finalized" (DAA\_Office)  d. "Amount Checked" (Exam\_Control\_Office)  f. "Fund Verified" (Treasurer)  g. "Approved" (Vice\_Chancellor) |
| 7. | The user may add comments or notes if necessary. |
| 8. | The user submits the updated status. |
| 9. | The system validates the update and saves the new status. |
| **Alternative Flow** | Step | Action |
| 7.A | A1: Unauthorized Access:  a. The system detects that the user does not have the necessary permissions to update the bill status.  b. The system displays an error message and denies access to the update function. |
| A2: Invalid Bill Selection:  a. The user selects a bill that does not exist or is not in a state that can be updated by their role.  b. The system displays an error message and prompts the user to select a valid bill. |
| A3: System Error:  a. The system encounters an error while trying to update the bill status.  b. he system displays an error message and logs the error for further investigation. |
| **Success End Condition** | 1. | The bill status is successfully updated, and the change is reflected in the system. |
| **Failed End Condition** | 1. | The bill status is not updated due to validation errors, unauthorized access, or system issues. |
| **Quality Requirements** | Step | Action |
| 1. | The system should provide real-time validation to ensure that only valid status updates are made. |
| 2. | The user interface should be intuitive, guiding the user through the process with clear instructions and feedback. |
| 3. | The system should log all status updates for audit and tracking purposes. |

**Table-08**

**UC-08: Review Bill**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Review Bill** | |
| **Goal** | To review the submitted exam bill for validity and correctness before forwarding it to the next stage.. | |
| **Precondition** | 1. The bill must be created and submitted by a teacher.  2. The Director must be logged in and authorized to review bills. | |
| **Post Condition** | 1. The bill status is updated to reflect the review outcome..  2The bill is either forwarded to the Registrar for further verification or rejected if found invalid. | |
| **Trigger** | The Director decides to review a submitted exam bill as part of their role in the bill approval process. | |
| **Primary Actor** | Director | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The Director logs into the system. |
| 2. | The Director navigates to the "Review Bill" section. |
| 3. | The system displays a list of submitted bills pending review. |
| 4. | The Director selects a bill from the list. |
| 5. | The system displays the details of the selected bill, including teacher information, bill amount, and submission date. |
| 6. | The Director reviews the bill details for accuracy and completeness. |
| 7. | The user may add comments or notes if necessary. |
| 8. | The user submits the updated status. |
| 9. | The system validates the update and saves the new status. |
| **Alternative Flow** | Step | Action |
| 8.A | A1: Unauthorized Access:  a.The system detects that the user does not have the necessary permissions to update the bill status.  b. The system displays an error message and denies access to the update function. |
| A2: Invalid Bill Selection:  a. The user selects a bill that does not exist or is not in a state that can be updated by their role.  b. The system displays an error message and prompts the user to select a valid bill. |
| A3: System Error:  a. The system encounters an error while trying to update the bill status.  b. The system displays an error message and logs the error for further investigation. |
| **Success End Condition** | 1. | The bill is successfully reviewed, and the status is updated to reflect the review outcome. |
| **Failed End Condition** | 1. | The bill review is not completed due to validation errors, unauthorized access, or system issues. |
| **Quality Requirements** | Step | Action |
| 1. | The system should provide real-time validation to ensure that only valid status updates are made. |
| 2. | The user interface should be intuitive, guiding the user through the process with clear instructions and feedback. |
| 3. | The system should log all status updates for audit and tracking purposes. |

**Table-09**

**UC-09: Verify Teacher Presence**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Verify Teacher Presence** | |
| **Goal** | To verify the presence of the teacher in the exam hall for the corresponding exam bill. | |
| **Precondition** | 1. The bill must be reviewed and forwarded by the Director.  2. The Registrar must be logged in and authorized to verify teacher presence. | |
| **Post Condition** | 1. The bill status is updated to reflect the verification outcome.  2. The bill is either forwarded to the DAA\_Office for finalization or rejected if the teacher's presence is not verified | |
| **Trigger** | The Registrar decides to verify the teacher's presence for a reviewed exam bill as part of their role in the bill approval process.. | |
| **Primary Actor** | Registrar | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The Registrar logs into the system. |
| 2. | The Registrar navigates to the "Verify Teacher Presence" section.. |
| 3. | The system displays a list of reviewed bills pending verification.. |
| 4. | The Registrar selects a bill from the list. |
| 5. | The system displays the details of the selected bill, including teacher information, exam date, and location. |
| 6. | The Registrar checks the attendance records to confirm the teacher's presence in the exam hall. |
| 7. | The Registrar updates the bill status to one of the following:   1. "Presence Verified" if the teacher's presence is confirmed. 2. "Presence Not Verified" if the teacher's presence cannot be confirmed. |
| 8. | The Registrar may add comments or notes to justify the verification decision. |
| 9. | The Registrar submits the verification outcome. |
| 10. | The system validates the verification and updates the bill status accordingly. |
| 11. | If the bill is marked as "Presence Verified," it is forwarded to the DAA\_Office for finalization. |
| **Alternative Flow** | Step | Action |
| 9.A | A1: Unauthorized Access:  a. The system detects that the user does not have the necessary permissions to verify teacher presence.  b. The system displays an error message and denies access to the verification function.. |
| A2: Invalid Bill Selection:  a. The Registrar selects a bill that does not exist or has already been verified.  b. The system displays an error message and prompts the Registrar to select a valid bill. |
| A3: Attendance Record Not Found:  a. The system cannot find the attendance record for the selected bill.  b. The system displays an error message and prompts the Registrar to check the attendance records manually.. |
| A4: System Error:  a. The system encounters an error while processing the verification.  b. The system displays an error message and logs the error for further investigation. |
| **Success End Condition** | 1. | The teacher's presence is successfully verified, and the bill status is updated to reflect the verification outcome. |
| **Failed End Condition** | 1. | The verification is not completed due to validation errors, unauthorized access, or system issues. |
| **Quality Requirements** | Step | Action |
| 1. | The system should provide real-time validation to ensure accurate and complete verification of teacher presence. |
| 2. | The user interface should be intuitive and provide clear instructions and feedback for the Registrar. |
| 3. | The system should log all status updates for audit and tracking purposes. |

**Table-10:**

**UC-10: Finalize Bill**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Finalize Bill** | |
| **Goal** | To finalize the exam bill after verifying all necessary checks and validations. | |
| **Precondition** | 1.The bill must have the "Presence Verified" status.  2. The DAA\_Office must be logged in and authorized to finalize bills. | |
| **Post Condition** | 1. The bill is finalized and marked as ready for the next steps in the billing process..  2. The finalized bill status is updated in the system. | |
| **Trigger** | The DAA\_Office decides to finalize a bill that has passed all previous checks and verifications.. | |
| **Primary Actor** | DAA\_Office (Dean of Academic Affairs Office) | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The DAA\_Office logs into the system. |
| 2. | The DAA\_Office navigates to the "Finalize Bill" section. |
| 3. | The system displays a list of bills pending finalization. |
| 4. | The DAA\_Office selects a bill from the list. |
| 5. | The system displays the details of the selected bill, including the verification status and any associated notes.. |
| 6. | The DAA\_Office reviews the bill details to ensure all necessary checks are complete. |
| 7. | The DAA\_Office updates the bill status to "Finalized." |
| 8. | The DAA\_Office may add comments or notes regarding the finalization decision. |
| 9. | The DAA\_Office submits the finalization outcome. |
| 10. | The system validates the finalization and updates the bill status accordingly. |
| 11. | The bill is now marked as "Finalized" and ready for further processing, such as fund verification or approval. |
| **Alternative Flow** | Step | Action |
| 10.A | A1: Unauthorized Access:  a. The system detects that the user does not have the necessary permissions to verify teacher presence.  b. The system displays an error message and denies access to the finalization function. |
| A2: Invalid Bill Selection:  a. The DAA\_Office selects a bill that does not exist or has already been finalized.  b. The system displays an error message and prompts the DAA\_Office to select a valid bill. |
| A3: Missing Verification:  a. The system detects that the bill has not been verified properly.  b. The system displays an error message and prompts the DAA\_Office to complete all necessary verifications before finalization. |
| A4: System Error:  a. The system encounters an error while processing the finalization.  b. The system displays an error message and logs the error for further investigation. |
| **Success End Condition** | 1. | The bill is successfully finalized, and the bill status is updated to reflect the finalization outcome. |
| **Failed End Condition** | 1. | The finalization is not completed due to validation errors, unauthorized access, or system issues |
| **Quality Requirements** | Step | Action |
| 1. | The system should provide real-time validation to ensure accurate and complete finalization of the bill. |
| 2. | The user interface should be intuitive and provide clear instructions and feedback for the DAA\_Office. |
| 3. | The system should log all finalization actions for audit and tracking purposes. |

**Table-11:**

**UC-11: Check Bill Amount**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Check Bill Amount** | |
| **Goal** | To verify the correctness and accuracy of the bill amount before proceeding to fund verification. | |
| **Precondition** | 1. The bill must be finalized by the DAA\_Office.  2. The Exam\_Control\_Office must be logged into the system and authorized to check bill amounts. | |
| **Post Condition** | 1. The bill amount is confirmed as correct and marked as checked.  2. If any discrepancies are found, the bill is flagged for review and correction.. | |
| **Trigger** | The Exam\_Control\_Office decides to verify the amount of a finalized bill before proceeding to fund verification. | |
| **Primary Actor** | Exam\_Control\_Office | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The Exam\_Control\_Office logs into the system. |
| 2. | The Exam\_Control\_Office navigates to the "Check Bill Amount" section. |
| 3. | The system displays a list of finalized bills pending amount verification. |
| 4. | The Exam\_Control\_Office selects a bill from the list. |
| 5. | The system displays the details of the selected bill, including the total amount and itemized charges. |
| 6. | The Exam\_Control\_Office reviews the bill details to ensure all charges are accurate and justified. |
| 7. | If the bill amount is correct, the Exam\_Control\_Office updates the bill status to "Amount Checked." |
| 8. | The Exam\_Control\_Office may add comments or notes regarding the amount verification. |
| 9. | The Exam\_Control\_Office submits the verification outcome. |
| 10. | The system validates the amount check and updates the bill status accordingly. |
| 11. | The bill is now marked as "Amount Checked" and ready for fund verification.. |
| **Alternative Flow** | Step | Action |
| 11.A | A1: Unauthorized Access:  a. The system detects that the user does not have the necessary permissions to verify teacher presence.  b. The system displays an error message and denies access to the amount checking function. |
| A2: Invalid Bill Selection:  a. The Exam\_Control\_Office selects a bill that does not exist or has already been checked.  b. The system displays an error message and prompts the Exam\_Control\_Office to select a valid bill. |
| A3: Discrepancy Found:  a. The Exam\_Control\_Office identifies a discrepancy in the bill amount  b. The Exam\_Control\_Office flags the bill for review and correction.  c. The system updates the bill status to "Discrepancy Found" and notifies the relevant parties for further action.. |
| A4: System Error:  a. The system encounters an error while processing the amount check.  b. The system displays an error message and logs the error for further investigation. |
| **Success End Condition** | 1. | The bill amount is successfully verified, and the bill status is updated to reflect the amount check outcome. |
| **Failed End Condition** | 1. | The amount check is not completed due to validation errors, unauthorized access, discrepancies found, or system issues. |
| **Quality Requirements** | Step | Action |
| 1. | The system should provide real-time validation to ensure accurate amount checking. |
| 2. | The user interface should be intuitive and provide clear instructions and feedback for the Exam\_Control\_Office. |
| 3. | The system should log all amount checking actions for audit and tracking purposes. |

**Table-12:**

**UC-12: Verify Fund Availability**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Verify Fund Availability** | |
| **Goal** | To ensure that sufficient funds are available to cover the exam bill before proceeding to final approval. | |
| **Precondition** | 1. The bill must have its amount checked and confirmed as correct.    2. The Treasurer must be logged into the system and authorized to verify fund availability. | |
| **Post Condition** | 1. The fund availability is confirmed, and the bill status is updated to reflect the verification.  2. If insufficient funds are found, the bill is flagged, and relevant parties are notified. | |
| **Trigger** | The Treasurer decides to verify the fund availability for a bill that has been amount-checked. | |
| **Primary Actor** | Treasurer | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The Treasurer logs into the system. |
| 2. | The Treasurer navigates to the "Verify Fund Availability" section. |
| 3. | The Treasurer navigates to the "Verify Fund Availability" section. |
| 4. | The Treasurer selects a bill from the list. |
| 5. | The system displays the details of the selected bill, including the total amount and fund status. |
| 6. | The Treasurer reviews the bill details and verifies the available funds. |
| 7. | If sufficient funds are available, the Treasurer updates the bill status to "Funds Verified." |
| 8. | The Treasurer may add comments or notes regarding the fund verification.. |
| 9. | The Treasurer submits the verification outcome.. |
| 10. | The system validates the fund verification and updates the bill status accordingly. |
| 11. | The bill is now marked as "Funds Verified" and ready for final approval. |
| **Alternative Flow** | Step | Action |
| 12.A | A1: Unauthorized Access:  a. The system detects that the user does not have the necessary permissions to verify teacher presence.  b. The system displays an error message and denies access to the amount checking function. |
| A2: Invalid Bill Selection:  a. The Treasurer selects a bill that does not exist or has already been verified for funds.  b. The system displays an error message and prompts the Treasurer to select a valid bill. |
| A3: Insufficient Funds:  a. The Treasurer identifies that there are insufficient funds to cover the bill amount.  b. The Treasurer flags the bill for review and possible action.    c. The system updates the bill status to "Insufficient Funds" and notifies the relevant parties for further action. |
| A4: System Error:  a.The system encounters an error while processing the fund verification.  b. The system displays an error message and logs the error for further investigation. |
| **Success End Condition** | 1. | The fund availability is successfully verified, and the bill status is updated to reflect the verification outcome. |
| **Failed End Condition** | 1. | The fund verification is not completed due to validation errors, unauthorized access, insufficient funds, or system issues. |
| **Quality Requirements** | Step | Action |
| 1. | The system should provide real-time validation to ensure accurate fund verification. |
| 2. | The user interface should be intuitive and provide clear instructions and feedback for the Treasurer. |
| 3. | The user interface should be intuitive and provide clear instructions and feedback for the Treasurer. |

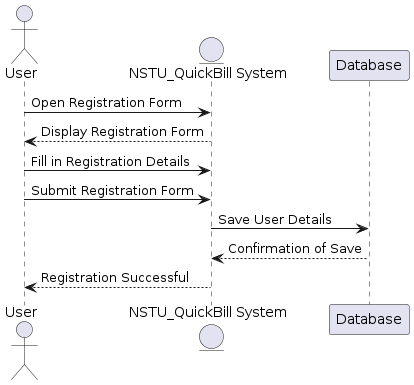
**Table-13:**

**UC-13: Final Approval**

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Final Approval** | |
| **Goal** | To provide the final approval for an exam bill after all previous verifications and checks have been completed. | |
| **Precondition** | 1. The bill must have passed through all prior checks and verifications, including review, teacher presence verification, bill finalization, amount checking, and fund availability verification..  2. The Vice\_Chancellor must be logged into the system and authorized to give final approval. | |
| **Post Condition** | 1. The exam bill is approved and marked as completed.  2. Relevant parties are notified of the final approval. | |
| **Trigger** | The Vice\_Chancellor decides to give the final approval for a bill that has passed all previous verifications. | |
| **Primary Actor** | Vice\_Chancellor | |
| **Secondary Actor** | N/A | |
| **Main Success Flow** | Step | Action |
| 1. | The Vice\_Chancellor logs into the system. |
| 2. | The Vice\_Chancellor navigates to the "Final Approval" section. |
| 3. | The system displays a list of bills pending final approval. |
| 4. | The Vice\_Chancellor selects a bill from the list. |
| 5. | The system displays the details of the selected bill, including all previous verification statuses. |
| 6. | The Vice\_Chancellor reviews the bill details and verification statuses. |
| 7. | The Vice\_Chancellor approves the bill if everything is in order. |
| 8. | The system updates the bill status to "Approved." |
| 9. | The Vice\_Chancellor may add comments or notes regarding the approval. |
| 10. | The Vice\_Chancellor submits the approval outcome. |
| 11. | The system validates the approval and updates the bill status to "Completed." |
| 12. | The system sends notifications to the relevant parties about the final approval. |
| **Alternative Flow** | Step | Action |
| 12.A | A1: Unauthorized Access:  a.The system detects that the user does not have the necessary permissions to grant final approval..  b. The system displays an error message and denies access to the final approval function. |
| A2: Invalid Bill Selection:  a. The Vice\_Chancellor selects a bill that does not exist or has already been approved.  b. The system displays an error message and prompts the Vice\_Chancellor to select a valid bill. |
| A3: Incomplete Verifications:    a. The Vice\_Chancellor identifies that not all required verifications are complete.  b.The Vice\_Chancellor flags the bill for further review and returns it to the relevant department.  c.The system updates the bill status to "Pending Further Verification" and notifies the relevant parties.. |
| A4: System Error:  a.The system encounters an error while processing the final approval.  b.The system displays an error message and logs the error for further investigation. |
| **Success End Condition** | 1. | The exam bill is successfully approved, and the bill status is updated to "Completed." |
| **Failed End Condition** | 1. | The final approval is not completed due to validation errors, unauthorized access, incomplete verifications, or system issues. |
| **Quality Requirements** | Step | Action |
| 1. | The system should provide real-time validation to ensure all prior verifications are completed before allowing final approval. |
| 2. | The user interface should be intuitive and provide clear instructions and feedback for the Vice\_Chancellor.. |
| 3. | The system should log all approval actions for audit and tracking purposes. |
| 4. | The system should ensure the integrity and security of the data throughout the approval process. |

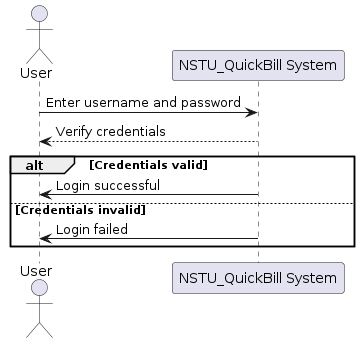
# Sequence Diagram

**8.1 Sequence Diagram** (**Registration)**

****

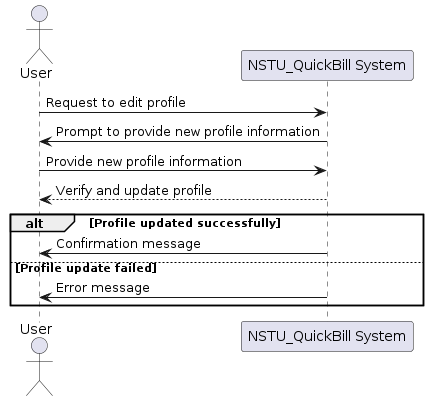
**Figure 2 :** **Registration**

**8.2 Sequence Diagram** (**Login)**

****

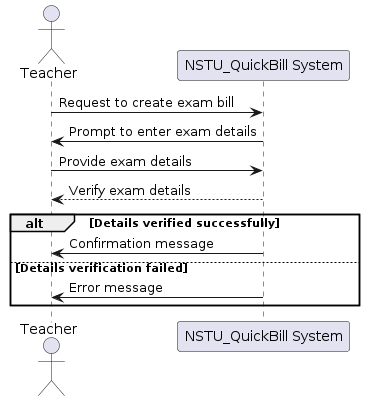
**Figure 3 Login**

**8.3 Sequence Diagram** (**Edit Profile)**

****

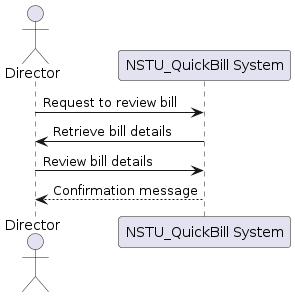
**Figure 4 Edit Profile**

**8.4 Sequence Diagram** (**Create Exam Bill)**

****

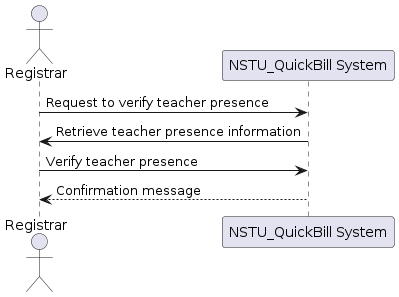
**Figure 5 Create Exam bill**

**8.5 Sequence Diagram** (**Review Bill)**

****

**Figure 6 Review Bill**

**8.6 Sequence Diagram** (**Verify Teacher Presence)**

****

**Figure 7 Verify Teacher Presence**

**8.7 Sequence Diagram** (**Finalize Bill)**

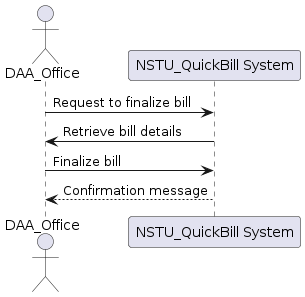
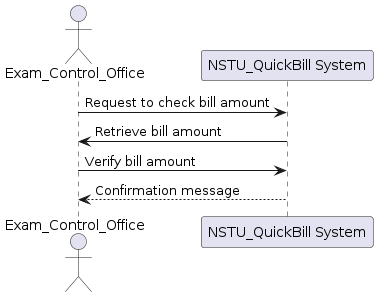
****

Figure 8 Finalize Bill

**8.8 Sequence Diagram** (**Verify Bill Amount)**

****

**Figure 9 Verify Bill Amount**

**8.9 Sequence Diagram** (**Verify Fund Availability)**

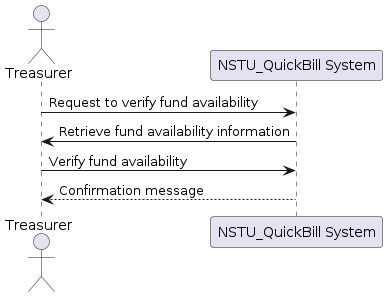
****

Figure 10 Verify Fund Availability

**8.10 Sequence Diagram** (**Final Approval)**

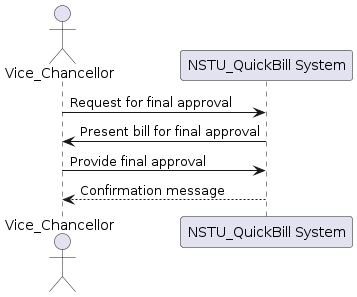
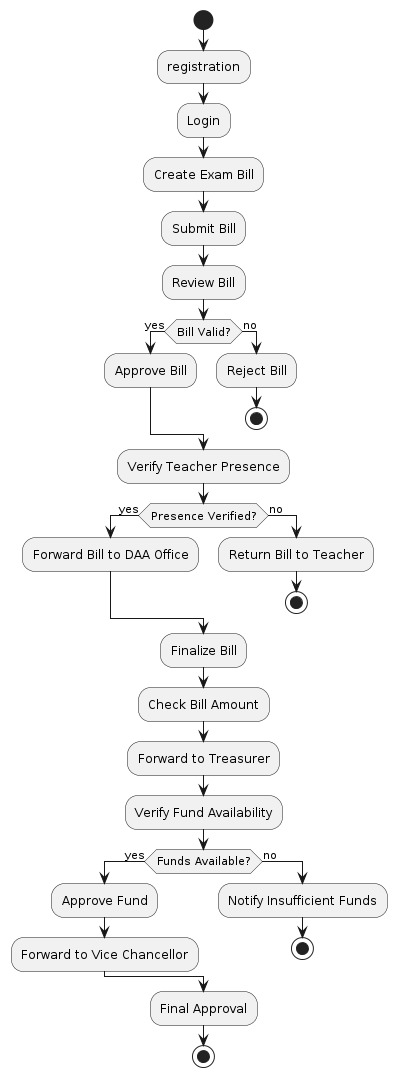
****

Figure 11 Final Approval

# Activity Diagram



**Figure 12 Activity Diagra**

# State Diagram

**Figure 13 State Diagram**

# Swimlane Diagram

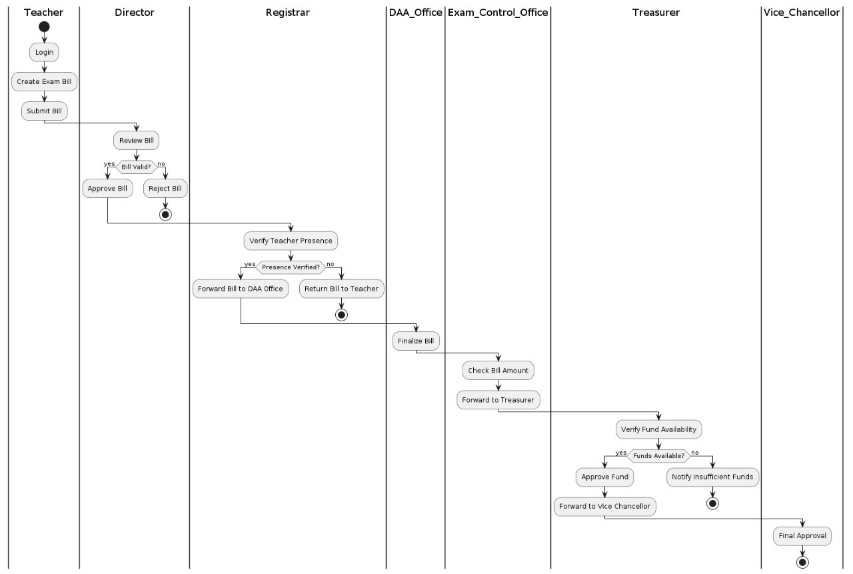


Figure 14 Swimlane Diagram

# Requirement Traceability Matrix

A traceability matrix is a document, usually in the form of a table, used to assist in determining the completeness of a relationship by correlating any two baselined documents using a many-to-many relationship comparison. It is often used with high-level requirements (these often consist of marketing requirements) and detailed requirements of the product to the matching parts of high-level design, detailed design, test plan, and test cases.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Requirements Traceability Matrix** | | | | | |  |
| **Project Name** | **NSTU\_QuickBill** | **Project Area** | | **Noakhali Science**  **and Technology University** | |  |
| **Functional**  **Activity** | **Use Case Reference** | **Design**  **Document**  **Reference** | **Code**  **Module/**  **Reference** | **User**  **Acceptance**  **Validation** | **Comments** | |
| **FR-01** | UC-02,UC-03 |  |  | Verified |  | |
| **FR-02** | UC-01,UC-05 |  |  | Verified |  | |
| **FR-03** | UC-08,UC-12,UC-13 |  |  | Verified |  | |
| **FR-04** | UC-08 |  |  | Verified |  | |
| **FR-05** | UC-10 |  |  | Verified |  | |
| **FR-06** | UC-11 |  |  | Verified |  | |
| **FR-07** | UC-12 |  |  | Pass |  | |
| **FR-08** | UC-13 |  |  | Verified |  | |
| **FR-09** | UC-07 |  |  | Verified |  | |
| **FR-10** | UC-08,UC-09 |  |  | Verified |  | |
| **FR-11** | UC-05,UC-06 |  |  | Pass |  | |
| **FR-12** | UC-07,UC-08,UC-04 |  |  | Verified |  | |
| **FR-13** | UC-03 |  |  | Pass |  | |