Software Requirements Specification

for

EXAMPILOT

An Automated Examination Management System for NSTU

Prepared by

Mir Mohammad Tahsin MUH2025007M
Imtiaz Chowdhury MUH2025027M
Irfanul Haque ASH1925021M

Institute of Information Technology
Noakhali Science and Technology University

Supervised By

Tasniya Ahmed
Assistant Professor,
Institute of Information technology
Noakhali Science and Technology University

11.06.2023

List of Figure

List of Tables

1. Introduction

The policy, scope, references, and summary of Software Requirements Specification (SRS) are all included in the SRS introduction. By presenting the problem statement in detail, the purpose of this document is to collect, evaluate, and provide a deeper understanding of the whole "EXAMPILOT" system. While it defines high-quality product features, it also focuses on the strengths that participants need and their needs. This page contains detailed "EXAMPILOT" information.

1.1 Problem Statement

In a traditional examination management system, request for setting question papers and the actual question paper from Controller Of Examination (COE) to various teachers and vice-versa , is done physically. Further, the work of sending the now finalized paper from COE to various superintendents of examination is also done physically.
This system generates a lot of security, economic and financial concerns , like -

- After setting the question paper the confidentiality and integrity is totally in the hands of a third person
- The financial cost of sending a physical request to teacher and the teacher sending it back physically is too much
- Moreover the question papers sent to different examination centers to Superintendent requires time .
- The confidentiality of teacher , i.e. the teacher who designed a particular question paper , sometimes is not maintained

There is no way we could solve these concerns without spending more money on this. On the other hand after the examination while processing the result from teacher to students also done physically via relevant department & Exam Controller.Also Students have to take marksheet from department physically. So there is a need of automation in this field also , an automated system which covers all these concerns in a cost effective way.

1.2 Purpose

The purpose of this document is to collect and analyze all the various ideas that have emerged in order to develop the program and its standards for alumni members. Also, to get a better idea of the project, describe the ideas that may be developed later, and write down ideas that are being considered but which can be eliminated as the system progresses, we will predict and plan how we expect this plan to be implemented.

In short, the purpose of this SRS document is to provide a comprehensive overview of our software product, including its specifications and objectives. This document describes the user interface, hardware, and software requirements, as well as its partners. Specifies how the system and its functions are viewed by administrators, alumni students, and general members. However, it assists any designer or developer in software delivery processes (SDLC).

1.3 Project Scope

This report will focus on evaluating the effectiveness of ExamPilot’s services and identifying potential areas for improvement in the platform's business model.

The project aims to automate the entire examination management system, including question paper setting, moderation, distribution, result processing, and publication. It will introduce a web-based platform that will enable administrators, teachers, exam controllers, superintendents, and students to perform their respective tasks efficiently and securely.
The main objectives of the project are:

1. To provide an automated system for setting, moderating, and distributing question papers, ensuring confidentiality and integrity.
2. To reduce the financial cost of physical communication and distribution of question papers to different examination centers.
3. To ensure the security and confidentiality of teacher information.
4. To provide an automated system for result processing and publication, reducing manual labor and errors.
5. To generate an automated mark sheet for students and enable them to view and download their results online.

The project scope includes the following modules:

Transfer of Question Paper: While doing question set or moderation, a teacher will set a question paper of a course and upload it. Another teacher will also upload a question paper (different set)of the same course/exam. Both questions will be received by exam moderation committee. Committee will moderate the question paper. After that they will send it to the exam controller and exam controller will send this question Superintendent(teacher) who will download the question paper on the exam day. One thing to mention Superintendent can't download the question paper before the exam time(specific time) .

Automated Result Generation: After the exam , first examiner will upload the marks of the answer script of a course . Second examiner will also upload the marks of the same answer script of the same course. If the difference of their marks is less than 14 then the average of marks of both examiners will be count. Otherwise it will send to third examiner who will review the answer script and give marks . His marks will be averaged with marks of the examiner which is close enough to third examiner .

Automated Marksheet Generation: After the evaluation an automated marksheet will be generated of each student. Result/marksheet will be mailed to each students mail address . Also students can see their result by logging into student portal .

1.4 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

1.5 References

- 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).

1.6 Overview

This report evaluates the effectiveness of ExamPilot's services and identifies potential areas for improvement in the platform's model. The proposed system aims to automate the traditional examination management process and address the concerns surrounding it, such as security, confidentiality, and financial costs. The system has two main features: question set and moderation using blockchain and IPFS, and result management.

The system involves three types of users: admin, teacher, and student. The admin manages the courses, students, faculties, subjects, and profiles of all users. Teachers are assigned to courses and can upload questions, moderate questions, enter marks, mark attendance, generate mark sheets, and view notifications. Students can view their courses, faculties, mark sheets, attendance reports, update their profile, and receive notifications.

The question set and moderation feature involves a teacher uploading a question paper for a course, and a second teacher uploading another question paper for the same course. Both papers will be received by the exam moderation committee, which will moderate the papers. The committee will then send the question paper to the exam controller, who will send it to the superintendent (teacher) on the exam day. The system prevents the superintendent from downloading the question paper before a specific time to maintain confidentiality and integrity.

The result management feature allows teachers (1st examiner and 2nd examiner) of each course to give results/marks. After that, an automated generated mark sheet will be created with the help of an algorithm based on the pre-defined rules. The student can see their result after logging into their portal, and the result/marksheet will be mailed to their mail address.

In conclusion, the proposed system aims to automate the manual examination management process and address the concerns surrounding it, such as security, confidentiality, and financial costs. By introducing automation and technology, the system can improve efficiency, accuracy, and transparency in the examination management process.

2. Stakeholders and Characteristics

Sure, below are the stakeholders and their characteristics for your project:

2.1. Students:

- **Stakeholders:** students who are enrolled in different courses
- **Characteristics:**
    - Can log in to their account using their Edu mail and password
    - Can view their assigned courses and faculties
    - Can view their attendance report and mark sheet for each course they are enrolled in
    - Can download their mark sheet in PDF format
    - Can update their profile details, including photo upload and password change

2.2. Teachers:

- **Stakeholders**: teachers who are responsible for setting questions, moderating them, evaluating the answer scripts, and marking attendance
- **Characteristics**:
    - Can log in to their account using their Edu mail and password
    - Can view their assigned courses and students' data
    - Can upload questions for moderation and submit the final question paper to the exam controller
    - Can download question papers on the exam day at a specific time
    - Can enter marks of the answer scripts, and if marks difference between 1st and 2nd examiner is more than 14 then requested for third examiner
    - Can mark attendance for each class
    - Can update their profile details
    - Can receive notifications related to their work

2.3. Exam Controller:

- **Stakeholders:** person responsible for managing the entire examination process
- **Characteristics:**
    - Can log in to their account using their Edu mail and password
    - Can send requests to the 1st and 2nd examiner for question set and moderation
    - Can accept questions from the moderation committee with the question set
    - Can send questions to the superintendent (teacher) on the exam day
    - Can approve the publication of results and send them to the department head

2.4. Administrators:

- **Stakeholders:** administrator responsible for managing the system as a whole
- **Characteristics:**
    - Can log in to their account using their Edu mail and password
    - Can manage courses, students, faculties, and subjects
    - Can search for any information related to the system
    - Can update their profile details

3. 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 E-Farm 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 farmers, consumers, agro-solution providers, agricultural equipment providers, logistics providers, and financial institutions. 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 Python, JavaScript, and Java, frameworks such as Django and Node.js, and databases such as PostgreSQL and MongoDB.

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 E-Farm platform.

3.1 JavaScript, JSX and React.js

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.

3.1.1 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.

It is faster than standard JavaScript because it optimizes when converting to standard JavaScript. Rather than dividing the markup and functionality in different files, React makes use of components.

Java: Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need to recompile.Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client–server web applications, with a reported 9 million developers.

3.1.2 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

3.2 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.

3.2.1 Python, Django Rest API

Python is a dynamically semantic, interpreted, object-oriented high-level programming language. Its high-level built-in data structures, together with dynamic typing and dynamic binding, make it ideal for Rapid Application Development and as a scripting or glue language for connecting existing components. Python's concise, easy-to-learn syntax prioritizes readability, which lowers software maintenance costs.

Python is widely used to create an application's back end. It should come as no surprise that there are a variety of Python frameworks available to help with server-side programming. The Django REST Framework (DRF) is a popular, robust, and versatile framework for creating Web APIs.

To get data from our server to our application, we'll use the Django Rest API. Representational State Transfer is abbreviated as REST. Application Programming Interface is what API stands for.

3.2.2 Database Server

PostgreSQL is an advanced, enterprise class open-source relational database that supports both SQL (relational) and JSON (non-relational) querying. PostgreSQL's speed, security and robustness make it suitable for 99% of applications, so it's a great starting place for our application. 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.

3.2.3 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. So we will use S3 to store audio files.

3.2.4 Blockchain Technology

Blockchain is a revolutionary technology that has gained significant attention and popularity in recent years. It is essentially a decentralized digital ledger that records transactions across multiple computers or nodes. Each transaction, or block, is added to a chain of previous transactions, creating a transparent and immutable record of all activities.The concept of blockchain was first introduced in 2008 by an anonymous person or group of individuals known as Satoshi Nakamoto, as the underlying technology behind the cryptocurrency Bitcoin. However, its potential applications extend far beyond digital currencies, with numerous industries and sectors exploring its transformative capabilities.At its core, blockchain technology relies on a distributed network of computers, often referred to as nodes, that work together to validate and verify transactions. This decentralized nature eliminates the need for a central authority or intermediary, such as a bank or government, to facilitate and authenticate transactions. Instead, consensus algorithms, like Proof of Work (PoW) or Proof of Stake (PoS), ensure the integrity and security of the network.One of the key features of blockchain is its immutability. Once a transaction is recorded and added to a block, it becomes virtually impossible to alter or delete. This characteristic makes blockchain highly secure and resistant to fraud or tampering. Each block contains a unique cryptographic hash that is derived from the previous block, ensuring the integrity and continuity of the chain.Blockchain technology also offers transparency and accountability. Since the ledger is distributed among multiple nodes, anyone with access to the network can view and validate transactions. This transparency helps to foster trust and reduce the need for blind reliance on centralized authorities. Moreover, the use of cryptographic techniques ensures the privacy and confidentiality of sensitive information, making it accessible only to authorized participants.The potential applications of blockchain span across various industries. In finance, blockchain can revolutionize traditional banking systems by enabling faster, more secure, and cost-effective cross-border transactions. It can also facilitate the creation of decentralized digital identities, streamlining identity verification processes and reducing the risk of identity theft.Supply chain management is another area where blockchain shows immense promise. By providing a transparent and traceable record of every transaction and movement of goods, blockchain can enhance efficiency, reduce fraud, and promote ethical practices. Consumers can verify the authenticity and origin of products, fostering trust and sustainability.Blockchain technology also holds great potential in healthcare, as it can securely store and share patient records, streamline medical research, and ensure the integrity of pharmaceutical supply chains. Additionally, blockchain has the ability to revolutionize the voting system, intellectual property management, real estate transactions, and much more.However, it's important to acknowledge that blockchain is not a one-size-fits-all solution. While it offers numerous advantages, it also faces challenges such as scalability, energy consumption (in the case of PoW), and regulatory concerns. Overcoming these hurdles requires ongoing research, development, and collaboration among stakeholders.

In conclusion, blockchain technology has the power to transform industries by providing secure, transparent, and decentralized solutions to long-standing problems. Its potential applications are vast, and as the technology continues to evolve and mature, we can expect to see even more innovative use cases emerging in the future.

3.2.5 IPFS Technology

IPFS, which stands for InterPlanetary File System, is a distributed peer-to-peer protocol and network designed to create a more efficient and resilient method of storing and sharing files on the internet. Unlike traditional web protocols that rely on centralized servers, IPFS utilizes a decentralized approach, making it a fundamental building block of the decentralized web.The core concept of IPFS is to replace the traditional location-based addressing of files with content-based addressing. In IPFS, files and other data are identified and retrieved based on their cryptographic hash, which represents the content itself rather than its location. This means that files are uniquely identified by their content, and any change to the content results in a different hash.IPFS organizes files into a global, distributed network of nodes, each hosting and providing access to specific files. When a file is added to IPFS, it is divided into smaller chunks, and each chunk is assigned a unique hash. These chunks are then distributed across the IPFS network and can be retrieved by their hashes, ensuring redundancy and resilience.When a user requests a file from IPFS, their request is routed through the network using a Distributed Hash Table (DHT) to find the nodes hosting the file's chunks. The file is then reconstructed by retrieving the chunks from the respective nodes. Since files are distributed across multiple nodes, IPFS allows for faster and more efficient content delivery, as files can be fetched from nearby nodes or nodes with the requested content.IPFS also incorporates a versioning system, enabling efficient updates and changes to files. Instead of creating new versions of an entire file, only the modified chunks are added to the network, reducing duplication and saving storage space. This versioning system, combined with content addressing, ensures that files on IPFS are immutable, preventing unauthorized modifications and providing a transparent history of changes.One of the key benefits of IPFS is its ability to enable decentralized and censorship-resistant file sharing. Since files on IPFS are distributed across multiple nodes, there is no single point of failure or reliance on a central authority. This makes it resilient to censorship attempts and provides robustness against network failures or attacks. IPFS also encourages data permanence, as long as at least one node in the network hosts a particular file, it remains accessible.IPFS has a wide range of potential applications. It can be used for distributing large files, hosting websites, sharing data between applications, and creating decentralized applications (DApps). Additionally, IPFS integrates well with blockchain technology, as it can be used to store and distribute decentralized storage solutions, such as blockchain-based file storage platforms.

In summary, IPFS is a decentralized peer-to-peer protocol and network that introduces content-based addressing and distributed storage to the internet. By eliminating reliance on centralized servers and enabling efficient file sharing, IPFS offers improved performance, resilience, and censorship resistance. As the decentralized web continues to evolve, IPFS plays a crucial role in enabling a more open, secure, and decentralized approach to data storage and sharing.

3.2.6 Solidity

Solidity is a high-level programming language specifically designed for developing smart contracts on the Ethereum blockchain. It is the most widely used language for writing smart contracts and is integral to the Ethereum ecosystem. Solidity combines elements of C++, Python, and JavaScript, making it accessible to developers with different programming backgrounds.Smart contracts are self-executing contracts with the terms of the agreement directly written into code.

These contracts run on the Ethereum Virtual Machine (EVM), which is a decentralized, Turing-complete virtual machine that executes smart contracts on the Ethereum network. Solidity allows developers to write these smart contracts and deploy them onto the Ethereum blockchain.One of the key features of Solidity is its ability to define the behavior of smart contracts and handle the interaction between different participants. It provides a wide range of data types, including integers, strings, booleans, arrays, and structs, along with advanced features such as events and modifiers. Solidity also supports inheritance, allowing developers to create modular and reusable code structures.Solidity is statically typed, meaning variables must be declared with their specific data type. This enhances security and prevents unexpected behavior during contract execution. It also includes various control structures like loops and conditionals, enabling developers to create complex logic within their contracts.Security is of utmost importance when it comes to smart contracts, as they handle real-world assets and value. Solidity includes several security mechanisms and best practices to help developers write secure code. However, due to its complex nature, smart contracts can still be vulnerable to bugs and exploits. It is crucial for developers to thoroughly test and audit their code to minimize risks.Solidity integrates with the Ethereum development ecosystem, including popular development tools like Truffle, Remix, and Hardhat. These tools provide frameworks, testing environments, and debugging capabilities to streamline the development and deployment process. They also offer features for interacting with smart contracts, such as deploying contracts, invoking functions, and reading data from the blockchain.Since its introduction, Solidity has evolved and undergone several updates to improve its functionality, security, and developer experience. The Solidity community actively contributes to its development and maintains a comprehensive documentation library, making it easier for developers to learn and leverage the language.In addition to Ethereum, Solidity has gained attention in other blockchain platforms and frameworks that support smart contracts. For example, it is used in platforms like Tron, EOS, and VeChain, which have their own virtual machines and blockchain ecosystems.Solidity has played a pivotal role in the growth and adoption of decentralized applications (DApps) and the expansion of the Ethereum ecosystem. It has enabled developers to build a wide range of applications, including decentralized finance (DeFi) protocols, non-fungible token (NFT) marketplaces, decentralized exchanges, and more. As blockchain technology continues to evolve, Solidity remains a critical tool for developers looking to create and deploy smart contracts on various platforms.

3.2.7 Ethereum

Ethereum is a decentralized, open-source blockchain platform that enables the development and execution of smart contracts and decentralized applications (DApps). It was proposed by Vitalik Buterin in late 2013 and launched in 2015. Ethereum introduced a groundbreaking concept by extending the capabilities of blockchain beyond cryptocurrency transactions, paving the way for programmable, self-executing contracts and a new era of decentralized applications.At its core, Ethereum operates as a global, decentralized virtual machine, known as the Ethereum Virtual Machine (EVM). The EVM executes smart contracts, which are pieces of code that define the rules and logic of agreements between parties. These contracts automatically enforce the terms, eliminating the need for intermediaries and enhancing transparency and efficiency.One of the key features of Ethereum is its native cryptocurrency, Ether (ETH). Ether serves multiple purposes within the Ethereum ecosystem. It is used as a digital currency for value transfer, incentivizing miners to validate transactions and secure the network. Additionally, Ether is required to pay for computational resources and gas fees, which are associated with executing smart contracts and interacting with DApps on the Ethereum network.Smart contracts on Ethereum are written in high-level programming languages such as Solidity and Vyper. These contracts are stored on the blockchain and can be accessed, executed, and interacted with by anyone on the network. They enable the creation of various decentralized applications, ranging from decentralized finance (DeFi) platforms, decentralized exchanges, gaming applications, supply chain solutions, and much more. Ethereum has played a crucial role in the development of the DeFi ecosystem. DeFi refers to a set of financial applications and protocols built on top of blockchain networks, primarily Ethereum, aiming to provide open and decentralized alternatives to traditional financial services. DeFi protocols enable activities such as lending and borrowing, yield farming, decentralized exchanges, and asset management, all governed by smart contracts. Ethereum has also popularized the concept of non-fungible tokens (NFTs). NFTs are unique digital assets that can represent ownership or proof of authenticity for various types of digital and physical items, such as artwork, collectibles, music, and real estate. The ERC-721 and ERC-1155 token standards on Ethereum have become the foundation for the creation and trading of NFTs.To maintain the integrity and security of the network, Ethereum currently operates on a Proof of Stake (PoS) consensus mechanism known as Ethereum 2.0. This upgrade aims to address scalability issues and reduce energy consumption by transitioning from the energy-intensive Proof of Work (PoW) consensus used in the earlier version of Ethereum.Ethereum's success has led to a vibrant ecosystem of developers, entrepreneurs, and innovators who continue to build and expand the possibilities of blockchain technology. The Ethereum community actively contributes to the development of the platform, proposing and implementing upgrades through Ethereum Improvement Proposals (EIPs). Ethereum's open-source nature encourages collaboration and fosters the creation of interoperable solutions.

In summary, Ethereum has revolutionized blockchain technology by introducing the concept of smart contracts and enabling the development of decentralized applications. It has opened up new avenues for innovation, transforming industries and creating opportunities for a more decentralized and inclusive future. With ongoing development and scalability improvements, Ethereum remains at the forefront of blockchain innovation and continues to shape the decentralized ecosystem.

3.2.8 Ganache & MetaMask

Ganache and MetaMask are two essential tools in the Ethereum ecosystem that greatly enhance the development and interaction with Ethereum-based applications. While Ganache is primarily used for local blockchain development and testing, MetaMask serves as a web extension wallet to securely manage Ethereum accounts and interact with DApps.Ganache, formerly known as TestRPC, is a local development blockchain that allows developers to create a personal Ethereum network on their machines. It is part of the Truffle suite, a popular development framework for Ethereum. Ganache provides a simulated environment that mimics the behavior of the Ethereum network, allowing developers to deploy and test smart contracts, run transactions, and inspect the blockchain state without interacting with the live Ethereum network. With Ganache, developers can easily create and manage multiple accounts, each with its own private key and associated balance of Ether. This enables them to simulate different user scenarios and test various functionalities of their smart contracts. Ganache also offers additional features like advanced transaction logging, deterministic mining, and configurable gas settings, providing developers with fine-grained control over their local blockchain environment. MetaMask, on the other hand, is a web extension wallet that allows users to manage Ethereum accounts and securely interact with DApps directly from their web browsers. It acts as a bridge between users and the Ethereum blockchain, providing a user-friendly interface for accessing and controlling Ethereum assets. MetaMask serves as a non-custodial wallet, meaning users retain control over their private keys and funds. It provides users with a mnemonic phrase (seed phrase) that can be used to restore their accounts in case of device loss or failure. Users can import existing accounts or create new ones within MetaMask, and it supports multiple Ethereum networks, including the mainnet, testnets, and custom networks. MetaMask integrates seamlessly with web browsers, injecting the necessary Ethereum-related functionalities into web pages. This allows users to interact with DApps and sign transactions securely without exposing their private keys. When a user interacts with a DApp, MetaMask prompts them to review and approve transactions, ensuring transparency and control over their funds. MetaMask also provides additional features like token management, allowing users to view and manage ERC-20 and ERC-721 tokens. It supports decentralized exchanges, enabling users to swap tokens directly within the wallet interface. Additionally, MetaMask supports various Ethereum Improvement Proposals (EIPs) that enhance the user experience and expand functionality. Together, Ganache and MetaMask provide powerful tools for developers and users in the Ethereum ecosystem. Ganache simplifies the process of local development and testing, enabling developers to iterate quickly and ensure the reliability of their smart contracts. MetaMask, on the other hand, offers a secure and convenient way for users to interact with DApps, manage their Ethereum assets, and participate in the decentralized web.Both Ganache and MetaMask contribute significantly to the growth and adoption of Ethereum by empowering developers and users to leverage the capabilities of the blockchain and participate in the decentralized economy.

4. Requirement Specification

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

4.1 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.

4.1.1 User login

FR-1	User accessed account		
Description	Users (Admin, Teacher, Student, Exam Controller) should be able to log in using their educational email and password.		
Stakeholders	Teacher, Student, Exam Controller,Admin	**Priority**	High

4.1.2 Update Profile

FR-2	User updates their profile		
Description	This functional requirement describes the process of allowing users to update their profile information. The feature allows users to edit their profile details such as name, email address, profile picture, password, and any other relevant information. To use the profile update feature, users must be authenticated, and access should be limited to authorized users only. The system should handle errors and display confirmation messages after the update is successful. The user interface should be intuitive and easy to use, and the system should maintain an audit trail of all profile updates. Overall, this feature should be implemented effectively to provide users with a seamless and secure experience while updating their profile.		
Stakeholders	Teacher, Student, Exam Controller,Admin	**Priority**	Medium

4.1.3 Password Recovery

FR-3	Password recovery for forgot password.		
Description	This functional requirement describes the process of password recovery for forgotten passwords in an online product purchasing system. The feature allows users to reset their password by providing a verification mechanism, such as an email or SMS, to ensure the user's identity. The system should enforce password strength requirements, display appropriate error messages, and provide a confirmation message to the user after they have successfully reset their password. The system should maintain an audit trail of all password recovery activities and ensure secure communication between the user's browser and the server. After resetting their password, users should be authenticated to the system and redirected to the appropriate page.		
Stakeholders	Teacher, Student, Exam Controller, Admin	**Priority**	Medium

4.1.4 Courses Management

FR-4	Courses Management		
Description	Admin should be able to manage courses, assign examiners to courses, and view course credits.		
Stakeholders	Admin	**Priority**	High

4.1.5 Department Management

FR-5	Department Management		
Description	Admin should be able to manage departments and their details.		
Stakeholders	Admin	**Priority**	High

4.1.6 Faculties Management

FR-6	Faculties Management.		
Description	Admin should be able to manage faculty members and add new faculty details		
Stakeholders	Admin	**Priority**	High

4.1.7 Student Management

FR-7	Manage students.		
Description	The student management module of the proposed automated examination management system aims to streamline various processes related to student information, course enrollment, marksheet generation, result publication, attendance tracking, and profile management.		
Stakeholders	Admin	**Priority**	High

4.1.8 Search Functionality

FR-8	Search Functionality		
Description	Admin should be able to search for specific information within the system, such as courses, students, faculties, etc.		
Stakeholders	Admin	**Priority**	Medium

4.1.9 Pending Requests

FR-9	Pending Requests (Question Upload & Question Download		
Description	Teachers should receive requests from the exam committee for question upload. They can accept the request, submit the question paper, or reject the request. Also, as an invigilator they can download question paper received from Exam Controller.		
Stakeholders	Teachers, Exam Committee	**Priority**	High

4.1.10 Enter Marks

FR-10	Enter Marks		
Description	Teachers (1st examiner and 2nd examiner) should be able to enter marks for answer scripts of a course. If the difference between their marks is more than 14, a 3rd examiner should be notified to review and provide marks. The final marks should be averaged based on the closest marks from the examiners.		
Stakeholders	Teachers	**Priority**	High

4.1.11 Mark Attendance

FR-11	Mark Attendance		
Description	Teachers should be able to mark attendance for students by selecting the date, course name, and semester. Attendance reports may be generated optionally.		
Stakeholders	Teachers	**Priority**	Medium

4.1.12 Generate Attendance Report

FR-12	Attendance report generation based on students' daily attendance		
Description	Teachers should be able to view & download final attendance report .		
Stakeholders	Teacher	**Priority**	High

4.1.13 Assigned Subjects

FR-13	Teachers should be able to view the subjects assigned to them.

Description	Teachers should be able to view the subjects assigned to them.		
Stakeholders	Teacher	**Priority**	Low

4.1.14 Marksheet Report Generation

FR-14	Marksheet Report		
Description	Teachers should be able to generate automated marksheets for each student. The marksheet should be sent to the exam controller via the department head for approval before it can be downloaded.		
Stakeholders	Teachers, Exam Controller, Department Head (Teacher)	**Priority**	High

4.1.15 System Notifications

FR-15	System Notification		
Description	User should receive notifications related to their assigned tasks and system updates.		
Stakeholders	Teacher, Exam Controller, Student	**Priority**	Medium

4.1.16 View Courses

FR-16	View Courses.		
Description	Students should be able to view their assigned courses.		
Stakeholders	Students	**Priority**	High

4.1.17 View faculties

FR-17	Students should be able to view information about their faculty members.		
Description	Students should be able to view information about their faculty members.		
Stakeholders	Students	**Priority**	Medium

4.1.18 View Marksheet

FR-18	View & Download marksheet.		
Description	Students should be able to view and download their result/marksheet for each semester. They can also download previous year results if available.		
Stakeholders	Students	**Priority**	High

4.1.19 View Attendance Report

FR-19	View Attendance Report		
Description	Students should be able to view and download their attendance report for each course.		
Stakeholders	Students	**Priority**	Medium

4.1.20 Accept Question

FR-20	Accept Question from Moderation Committee

Description	Exam Committee will receive questions from teachers . Also, Exam Controller should be able to accept questions from the moderation committee, along with the question set.		
Stakeholders	Exam Controller, Moderation Committee (Teacher)	**Priority**	High

4.1.21 Send Question

FR-21	Send Questions to invigilators		
Description	Teachers will send questions to exam committee & then exam committee will send it to Moderation Committee .Also Exam Controller should send the final question set to the invigilators (teachers) on the exam day. Invigilators should be able to download the question paper only after a specific time.		
Stakeholders	Exam Controller, Invigilator (Teacher)	**Priority**	High

4.1.22 Approval of Result Publication

FR-22	Approval of Result Publication		
Description	Exam Controller should review and approve the publication of results. The approved results should be sent to the department head.		
Stakeholders	Exam Controller, Department Head (Teacher)	**Priority**	High

4.1.23 User logout

FR-23	User logout from their account .		
Description	The user will be able to log out of his account at the end of his need. Users will need to login again for later use.		
Stakeholders	Admin, Teacher, Exam Controller, Student	**Priority**	Medium

4.2 Data Requirement

Based on the description of your project, it seems that your system would require several types of data to function effectively. Here are some potential data requirements for each of the three main services you are providing:.

Requirement Name	Requirement Description	Stakeholders	Priority
DR-1:User Authentication	Users should be able to log in to the system using their edu mail and password.	Admin, Teacher, Student	**High**
DR-2:Captcha Verification	The login page should include a captcha verification to prevent automated login attempts.	Admin, Teacher, Student	**Medium**
DR-3:Forgot Password	Users should be able to reset their password through OTP verification or confirmation URL.	Admin, Teacher, Student	**High**
DR-4:Dashboard	Each user type (Admin, Teacher, Student) should have a dashboard with relevant information and functionalities.	Admin, Teacher, Student	**High**
DR-5:Course Management	Admin should be able to manage courses, assign examiners, and view course details, including credits and department.	Admin	**High**
DR-6:Student Management	Admin should be able to manage student information, add new students, and view student details.	Admin	**High**
DR-7:Subject Management	Admin should be able to manage subject details such as subject names (e.g., IIT, CSE).	Admin	**Medium**
DR-8:Faculty Management	Admin should be able to manage faculty members, add new faculty, and view faculty details.	Admin	**Medium**

Requirement Name	Requirement Description	Stakeholders	Priority
DR-9:Search Function	The system should include a search functionality for users to search for specific information.	Admin, Teacher, Student	**Medium**
DR-10:Admin Profile	Admin should have a profile page to update their personal information and settings.	Admin	**Medium**
DR-11:Teacher Home	Teachers should have a home page with information about assigned courses and students.	Teacher	**High**
DR-12:Pending Requests	Teachers should be able to view and respond to pending requests from the exam controller, such as question uploads.	Teacher	**High**
DR-13:Question Download	Teachers should receive requests to download questions as superintendents on the exam day at a specific time.	Teacher	**High**
DR-14:Moderator Committee	Teachers should receive questions from the 1st and 2nd examiners for moderation and send the final question to the exam controller.	Teacher	**High**
DR-15:Enter Marks	Teachers should be able to enter marks for each student, including CT1, CT2, CT3, attendance, and final marks.	Teacher	**High**
DR-16:Mark Attendance	Teachers should be able to mark attendance by selecting the date, course, semester, and student data.	Teacher	**Medium**
DR-17:Attendance Report	Teachers should be able to generate attendance reports, optionally, and download them in PDF format.	Teacher	**Medium**
DR-18:Assigned Subjects	Teachers should be able to view the subjects assigned to them.	Teacher	**Medium**
DR-19:Marksheet Report	Teachers should be able to generate automated marksheet reports and optionally send them to the exam controller.	Teacher	**High**
DR-20:Student Marksheet	Students should be able to view their marksheet for previous years and semesters, and download them in PDF format.	Student	**High**
DR-21:Update Profile	Students should be able to update their profile information, including photo upload, password change, and phone number.	Student	**High**

Requirement Name	Requirement Description	Stakeholders	Priority
DR-22:Notifications	Users should receive notifications through email, such as marksheet availability or important updates.	Admin, Teacher, Student	**Medium**
DR-23:Logout	All users should have the ability to log out of the system.	Admin, Teacher, Student, Exam Controller	**High**
DR-24:Request Management	Exam committee should be able to send requests to 1st and 2nd examiners for question set and moderation.	Exam Controller	**High**
DR-25:Result Approval	Exam controllers should be able to approve the publication of results and send them to the department head.	Exam Controller, Department Head	**High**

In addition to the above, your 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.

4.3 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
Login Page Performance	The login page should load quickly and handle concurrent user logins efficiently.	Admin, Teacher, Student, Exam Controller	**High**
Dashboard Performance	The dashboard should load swiftly and display relevant information without significant delays.	Admin, Teacher, Student	**High**
Question Upload Speed	The system should allow teachers to upload question papers quickly and efficiently, ensuring minimal waiting times.	Teacher, Exam Controller	**High**
Question Download Speed	Superintendents should be able to download question papers swiftly on the exam day, preventing delays in distributing papers to examination centers.	Superintendent, Exam Controller	**High**
Mark Entry Efficiency	Teachers should be able to enter marks efficiently, with the system responding promptly and without any significant delays.	Teacher	**High**
Marksheet Generation	The automated marksheet generation process should be fast and handle large volumes of data effectively, ensuring minimal processing time to generate marksheet reports for students.	Teacher, Exam Controller	**High**

Requirement Name	**Requirement Description**	**Stakeholders**	**Priority**
Result Publication Speed	The system should publish results promptly and handle the distribution of result notifications efficiently, ensuring timely access to results for students and stakeholders.	Teacher, Exam Controller, Student	**High**
Search Function Speed	The search functionality within the system should deliver search results quickly, even when dealing with large databases of students, courses, or other relevant information.	Admin, Teacher, Student	**Medium**
Attendance Marking	The process of marking attendance should be fast and responsive, allowing teachers to record attendance efficiently and without significant delays.	Teacher	**Medium**
Performance Reporting	The system should be capable of generating performance reports, such as attendance reports or marksheet reports, in a timely manner, ensuring that stakeholders have access to up-to-date information.	Teacher, Exam Controller	**Medium**
Notifications Delivery	The system should deliver notifications promptly, whether through email or within the user interface, to keep stakeholders informed about important updates, results, or other relevant information.	Admin, Teacher, Student	**Medium**
Database Performance	The underlying database system should be optimized to handle concurrent access and large datasets efficiently, ensuring fast retrieval and manipulation of data.	System Administrators	**High**

4.3.1 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.

4.3.2.1.1 Login Page

The system should be able to handle at least 500 simultaneous logins from teachers, students, and administrators.

4.3.2.1.2 Dashboard

The system should be able to handle a large amount of data for courses, students, subjects, and faculties. It should be scalable to accommodate growth in user base and course offerings.

4.3.2.1.3 Pending Requests

The system should be able to manage and process multiple pending requests from exam controllers to teachers for question set and moderation.

4.3.2.1.4 Enter Marks

The system should be able to handle multiple entries of marks for each student, including CT marks, attendance marks, and final marks.

4.3.2.1.5 Mark Attendance

The system should be able to handle marking attendance for a large number of students.

4.3.2.1.6 Marksheet Report

The system should be able to generate automated mark sheets for a large number of students and store them securely**.**

4.3.2.1.7 Notifications

The system should be able to send notifications to teachers, students, and administrators via email or in-app notifications.

4.3.2.1.8 Database

The database used by the system should be optimized for high performance and low latency to handle a large volume of data.

4.3.2.1.9 Network Bandwidth

The system should require minimal network bandwidth to ensure fast and efficient data transfer.

4.3.2.1.10 Security

The system should ensure the confidentiality, integrity, and availability of data. Access controls should be in place to prevent unauthorized access**.**

4.3.2.1.11 Scalability

The system should be scalable to accommodate a large number of users during peak hours without any significant degradation in performance.

4.3.2.1.12 Availability

The system should be available 24/7 with minimal downtime for maintenance and upgrades.

4.4 Maintainability and Supportability

4.4.1 Maintenance Requirements

4.4.1.1 Code Maintainability: The code should be written in a modular format with well-defined functions and classes to make it easy to maintain and modify.

4.4.1.2 Continuously Updating: The system should be updated regularly to include new features and address any issues that may arise.

4.4.1.3 Documentation: The system should have proper documentation for all its modules, functions, and APIs to ensure that it can be easily understood and maintained by other developers.

4.4.1.4 Version Control: The system should use version control software like Git to keep track of changes made to the codebase and facilitate collaboration between developers.

4.4.1.5 Testing: A comprehensive testing suite should be built into the system to ensure that all features are working as intended and to catch any bugs or errors.

4.4.1.6 Scalability: The system should be designed with scalability in mind, so that it can handle an increasing number of users and data without performance issues.

4.4.1.7 Security: The system should be secure and protected against unauthorized access, data breaches, and other security threats.

4.4.1.8 Error Handling: The system should have robust error handling mechanisms to detect and report any errors that occur during runtime.

4.4.1.9 Monitoring: The system should be monitored regularly to identify and resolve any performance issues, security vulnerabilities, or other problems.

4.4.1.10 Upgrades: The system should be able to support upgrades and updates to its underlying technologies, such as databases or third-party APIs, without breaking existing functionality**.**

4.4.2 Supportability Requirements

4.4.2.1 Easy Maintenance: The system should be designed in a way that makes it easy to maintain by having well-documented code and clear documentation.

4.4.2.2 Technical Support: There should be a dedicated technical support team available to provide assistance to users if they encounter any issues or have questions about the system.

4.4.2.3 User Training: The system should come with user training materials to help users understand how to use the system effectively.

4.4.2.4 Compatibility: The system should be compatible with a wide range of devices and operating systems to ensure that all users can access it.

4.4.2.5 Scalability: The system should be able to handle an increasing number of users and data as the user base grows.

4.4.2.6 Backups: There should be automatic backups of the system's data to prevent loss of important information in case of system failure.

4.4.2.7 Disaster Recovery: The system should have a disaster recovery plan in place, which should include regular backups, data redundancy, and a tested recovery process.

4.4.2.8 Security Updates: The system should receive regular security updates to protect against emerging threats and vulnerabilities.

4.4.2.9 Monitoring: The system should have a monitoring system in place to detect and resolve any performance issues, security breaches, or other problems.

4.4.2.10 Upgrades: The system should be able to support upgrades and updates to its underlying technologies, such as databases or third-party APIs, without breaking existing functionality.

4.5 Security Requirements

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

4.5.1 Authentication: All users should be authenticated using their unique credentials (edu mail and password) to ensure that only authorized users can access sensitive data.

4.5.2 Authorization: The system should implement role-based access control to ensure that users can only access the data that they are authorized to view or modify.

4.5.3 Encryption: All sensitive data, such as question papers, answer scripts, and student information, should be encrypted to prevent unauthorized access or tampering.

4.5.4 Access Control: The system should have strict access control measures in place to prevent unauthorized access to sensitive data, such as firewalls, intrusion detection systems, and antivirus software.

4.5.5 Backup and Recovery: The system should have a robust backup and recovery strategy in place to ensure that all data is backed up regularly and can be recovered quickly in case of system failure or data loss.

4.5.6 Logging and Auditing: The system should have logging and auditing capabilities to track and monitor all activities performed by users and detect any suspicious behavior or security breaches.

4.5.7 Secure Communication: The system should use secure communication protocols, such as SSL/TLS encryption, to protect data transmitted between different components of the system.

4.5.8 Vulnerability Assessment: The system should undergo regular vulnerability assessments and penetration testing to identify any potential security weaknesses and address them promptly.

4.5.9 Data Privacy: The system should comply with relevant data privacy regulations, such as GDPR, and ensure that all user data is kept confidential and protected at all times.

4.5.10 Secure Storage: The system should store all data securely, using appropriate storage technologies and encryption methods, to prevent unauthorized access or data theft.

4.6 Usability and Human Integrity Requirements

This system will provide more user-friendly environment

Usability Requirements:

- ➢ The system should be user-friendly with a simple and intuitive interface that is easy to navigate.
- ➢ The system should be accessible from multiple devices, such as computers, tablets, and smartphones.
- ➢ The system should provide clear instructions and helpful feedback to users, especially in case of errors or unexpected events.
- ➢ The system should be responsive and have a fast loading speed, to minimize user frustration.
- ➢ The system should allow users to customize their settings and preferences, such as language and notification options.
- ➢ The system should have a helpdesk or support team available to assist users with technical issues or questions.

Human Integrity Requirements:

- ➢ The system should maintain the privacy and confidentiality of all exam-related data, including student information, question papers, and answer scripts.
- ➢ The system should ensure that all users are verified and authenticated before accessing sensitive data or performing any actions.
- ➢ The system should prevent any unauthorized modifications, deletions, or tampering of data by implementing strict access control measures and encryption protocols.
- ➢ The system should protect against any potential security breaches, such as hacking, malware, or phishing attacks, by regularly monitoring and updating its security features.
- ➢ The system should guarantee equal and fair treatment of all students, regardless of their background, gender, or ethnicity, by using unbiased grading methods and avoiding any form of discrimination.
- ➢ The system should comply with all relevant laws, regulations, and ethical standards related to the handling of exam-related data, such as GDPR, FERPA, and HIPAA.

4.7 Look and Feel Requirements

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

4.7.1 Appearance Requirements

- The user interface should be clean, modern, and intuitive, with a consistent design across all pages.

- The color scheme should be professional and easy on the eyes, with clear contrasts between foreground and background elements.
- All text should be legible and appropriately sized, with good use of spacing and typography.
- Buttons, links, and other interactive elements should be clearly labeled and easy to click or tap.
- Navigation menus should be well-organized and easily accessible from any page.
- The system should include helpful tooltips, hints, and explanations to guide users through complex processes or unfamiliar tasks.
- Icons and graphics should be used sparingly and only where necessary, to avoid clutter and confusion.
- Forms and input fields should be designed with usability in mind, including validation and error handling.
- The system should be responsive and adapt to different screen sizes and resolutions, including mobile devices.
- The overall look and feel of the system should convey professionalism, reliability, and trustworthiness, to inspire confidence in its users.

4.8 Style Requirements

Here are some style requirements for the proposed examination management system:

- The system should follow a consistent and cohesive design style throughout all its pages and modules.
- All pages should have a clear and identifiable header section, which includes the logo of the institution or organization, as well as navigation menus and search bars.
- The use of color should be consistent with the branding of the institution or organization, and the color palette should be carefully chosen to convey a professional and trustworthy image.
- Typography should be legible and easy to read across different devices and screen sizes.
- Buttons and other interactive elements should have a modern and sleek design, with hover and click effects that provide clear feedback to users.
- Forms and input fields should use a modern and minimalist design, with clear labels and placeholders that guide users through the process.
- Icons and images should be used sparingly, but where necessary, they should be high-quality and consistent with the overall design style.
- Animations and transitions should be used to enhance the user experience, but they should not be excessive or distracting.
- The system should have a clean and uncluttered layout, with appropriate spacing and alignment between different elements.
- The overall style of the system should be visually appealing, professional, and user-friendly, to create a positive impression on its users.

4.9 Legal Requirements

There are several legal requirements that the proposed examination management system must comply with:

4.9.1 Privacy laws: The system must comply with data privacy laws and regulations, such as GDPR, FERPA, and HIPAA, to protect the privacy and confidentiality of student information and exam-related data.

4.9.2 Intellectual property rights: The system must respect intellectual property rights, including copyrights and trademarks, when using third-party materials or intellectual property.

4.9.3 Accessibility requirements: The system must be accessible to users with disabilities, in compliance with accessibility laws and regulations, such as the Americans with Disabilities Act (ADA) and Web Content Accessibility Guidelines (WCAG).

4.9.4 Security laws: The system must comply with security laws and regulations, including data breach notification laws, to prevent unauthorized access, theft, or misuse of sensitive data.

4.9.5 Anti-discrimination laws: The system must comply with anti-discrimination laws and regulations, to ensure equal treatment and opportunities for all students, regardless of their background, gender, or ethnicity.

4.9.6 Ethical standards: The system must adhere to ethical standards and best practices, such as those set forth by professional organizations like IEEE and ACM, to promote fairness, transparency, and integrity in the examination process.

4.9.7 Contractual obligations: The system must comply with any contractual obligations between the institution or organization and its stakeholders, including teachers, students, examiners, and other third parties involved in the examination process.

Overall, the system should prioritize legal compliance and ethical considerations, to ensure a fair, secure, and trustworthy examination management process for all stakeholders.

5. 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.

5.1 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.

5.1.1 Hold 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

5.1.2 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.

5.1.3 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.

5.1.4 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.

5.2 Sample of requirement collection

5.2.1 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.

Distribute Questionnaires

Methodology:

The questionnaires were distributed to stakeholders from many students of Noakhali Science & Technology University, Noakhali. It takes just a few minutes to complete the questionnaires.

Participants:

A total of 152 responses were collected for both project requirements & functional requirements respectively.

Findings:

The following are the key findings from the interviews:

- ➢ Current examination management has flaws in terms of confidentiality & Integrity, cost-time effectiveness, accessibility.
- ➢ There are much difficulties in accessing exam results or mark sheets. So, it will be better if the whole system turned to automation.
- ➢ In the current result publication process, student have to face difficulties as well as have to wait for much long time to get the results.
- ➢ Blockchain & IPFS System should be used to make the transfer of questions highly secured
- ➢ Mobile device would be more convenient.
- ➢ Pre-defined password system would be more convenient.
- ➢ Additionally sign in with reCAPTCHA can be used.
- ➢ For forgetting password OTP verification would be more convenient.
- ➢ In student dashboard courses, faculties, marksheet, attendance report, result, update profile features should be included.
- ➢ After result publication notified through mail would be more convenient for students.
- ➢ Communicating with relevant course teacher using same platform will be much more convenient.
- ➢ For Assignments & Projects, everything in the same platform would be much more
- ➢ convenient
- ➢ Respective course material for each course should be included.
- ➢ General notices of university can also be shown.
- ➢ Notify students about their progress and awarded them

Key Requirements:

Based on the findings from the interviews, the following are the key requirements for **ExamPilot** System:

- User-friendly interface with easy navigation.
- Pre-defined password system.
- OTP verification for password recovery.
- Marksheet download in PDF format.

Assumptions:

It was assumed during the survey process that the **ExamPilot** system will be accessible via the web .

Limitations:

Responses were collected less than the expectation.

Conclusion:

The stakeholders' questionnaires surveys are provided valuable insights into the requirements for the **ExamPilot** system. The key findings and requirements will be used to develop a comprehensive set of requirements for the system.

5.2.2 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.

Interview

Methodology:

The interviews were conducted with stakeholders (teachers & Exam Controller) from many departments of Noakhali Science & Technology University. The interviews were conducted in a one-to-one format, lasting approximately 7-10 minutes each.

Participants:

A total of 14 were interviewed, including:
13 Teachers &
One assistant Exam Controller.

Findings:

The following are the key findings from the interviews:

- Very Unsatisfaction with the current examination management system.
- Current examination management has flaws in terms of confidentiality & Integrity, cost-time effectiveness, accessibility.
- Teachers have to play role as question setter & invigilator frequently.
- An automated examination management system with features for transfer like blockchain & IPFS protocol-based question transfer would improve the security of exam papers.
- Laptop will be much more convenient to use the proposed system.
- Pre-defined password system would be more convenient for sign in & verification.
- For forgetting password, mail verification would be more convenient.
- Moderator will receive question papers from teachers & after modification it will be sent to exam controller through automated system.

- On exam day teacher will get question paper's soft copy from exam controller in a specific time as an invigilator.
- After result publication receiving result through mail & student portal.
- Communicating with course teacher or faculty easily through the portal for doubts or queries features would be convenient for students.

Key Requirements:

Based on the findings from the interviews, the following are the key requirements for **ExamPilot** System:

- User-friendly interface with easy navigation.
- Secure Transfer mandatory
- Auto marksheet generator
- Auto attendance report generator

Assumptions:

It was assumed during the interview process that the ExamPilot system will be accessible via the web.

Limitations & Challenges:

Most of teachers are not familiar with blockchain & IPFS technology.

Conclusion:

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

5.3 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.

5.3.1 Review the Requirements

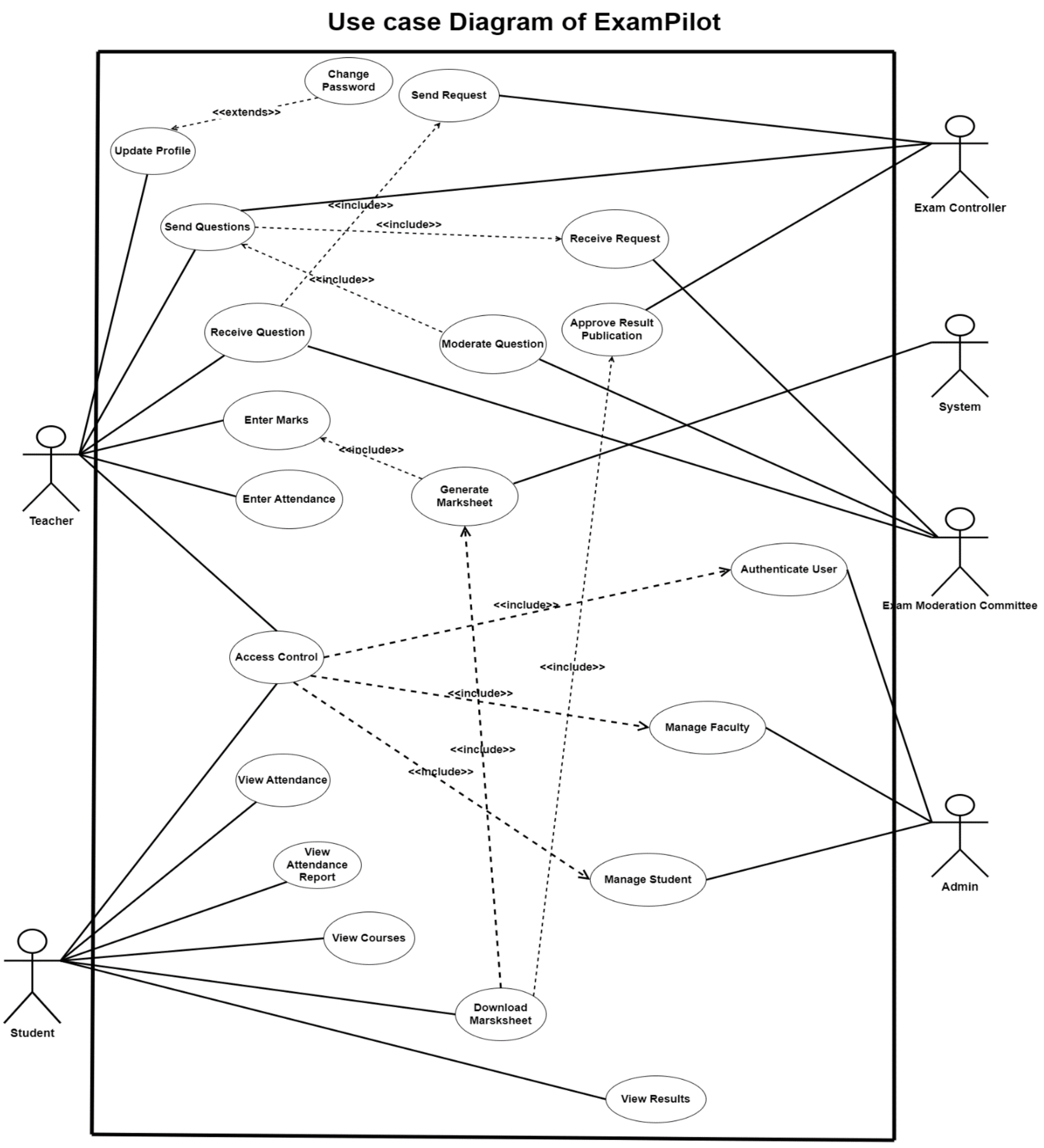
Negative peer review, especially the type of rigorous review called evaluation, is unique among the highest quality software processes available. We had a team of reviewers representing different perspectives and carefully examined written needs, analysis models, and related information on disability.

5.3.2 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.

5.3.3 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.



6. Use Case Diagram

Figure 1: Usecase Diagram

7. Use Case Description

UC-01: Create Account

Use Case	**Create Account**
Goal	Allow customers to create an account in the ExamPilot System
Precondition	N/A
Success End Condition	Customer account is created and the system confirms the creation
Failed End Conditions	1)The username or email is already in use 2)The customer's information is incomplete or invalid
Primary Actor	Customer
Secondary Actor	N/A
Trigger	The customer requests to create an account
Main Success Flow	1)The system presents the customer with a registration form, 2)The customer enters their personal information (e.g. name, address, email, username, password) 3)The customer submits the form, 4)The system validates the customer's information, 5)The system creates a new customer account and assigns a unique ID, 6)The system sends a confirmation email to the customer, 7) The system displays a success message to the customer.
Alternative Flow	2a)If the customer enters invalid or incomplete information, the system displays an error message and asks the customer to correct the information 6a)If the username or email is already in use, the system displays an error message and asks the customer to choose a different username or email
Quality Requirements	1)The system should be able to handle a large number of customer account creations simultaneously without crashing, 2) The customer's information should be securely stored and protected from unauthorized access.

UC-02: Access Control

Use Case	**Access Control**
Goal	Allow customers to log in to their account in the E-Farm System
Precondition	The customer must have an existing account in the system
Success End Condition	Customer is logged in and has access to their account information

Use Case	Access Control
Failed End Conditions	1)The customer enters incorrect login credentials, 2) The customer's account has been deactivated
Primary Actor	Customer
Secondary Actor	N/A
Trigger	The customer requests to log in to their account
Main Success Flow	1)The system presents the customer with a login page 2)The customer enters their username and password, 3)The customer clicks the "Log in" button 4)The system validates the customer's login credentials, 5) The system logs the customer in and grants access to their account information
Alternative Flow	2a)If the customer enters incorrect login credentials, the system displays an error message and asks the customer to re-enter their credentials 5a)If the customer's account has been deactivated, the system displays an error message and asks the customer to contact customer support
Quality Requirements	1)The system should protect the customer's login credentials from unauthorized access or disclosure, 2) The system should be able to handle a large number of simultaneous login requests without crashing or slowing down.

UC-03: View Product

Use Case	View Products
Goal	Allow customers to view the available agricultural products in the system
Precondition	The customer must be logged in to their account
Success End Condition	Customer is presented with a list of available products and can view their details
Failed End Conditions	None
Primary Actor	Customer
Secondary Actor	None
Trigger	The customer requests to view the available products
Main Success Flow	1)The system presents the customer with a list of available products, 2)The customer can filter the list based on criteria such as product type or price range, 3)The customer selects a product from the list to view its details, 4)The system presents the customer with the selected product's details (e.g. product name, description, price, image)
Alternative Flow	1a)If there are no products available, the system displays a message informing the customer that there are no products to display
Quality Requirements	1)The system should be able to handle a large number of products without slowing down or crashing 2)The system should display accurate and up-to-date information about the products.

UC-04: Order Product

Use Case	Order Product
Goal	Allow customers to order agricultural products from the system
Precondition	The customer must be logged in to their account and have selected a product to order
Success End Condition	Customer has successfully placed an order for the selected product
Failed End Conditions	1)The customer has insufficient funds to complete the purchase, 2)The selected product is out of stock
Primary Actor	Customer
Secondary Actor	Admin
Trigger	The customer requests to place an order for a selected product
Main Success Flow	1)The system presents the customer with a form to fill out their shipping and payment information 2)The customer fills out the form and clicks the "Place Order" button 3)The system validates the customer's payment information and confirms the order, 4)The system updates the product inventory to reflect the purchase, 5)The system sends a confirmation email to the customer
Alternative Flow	3a)The customer has insufficient funds to complete the purchase, the system displays an error message and asks the customer to enter a different payment method, 5a)If the selected product is out of stock, the system displays an error message and asks the customer to select a different product
Quality Requirements	1)The system should protect the customer's payment information from unauthorized access or disclosure 2) The system should update the product inventory in real-time to prevent overselling or underselling.

UC-05: Search Product

Use Case	Search Product
Goal	The goal of this use case is to allow the customer to search for and find the desired agri-product on the platform.
Preconditions	The customer must have a valid account on the platform and the platform must have a sufficient number of agri-products available for purchase.
Success end condition	The customer successfully finds the desired agri-product and adds it to their cart and proceeds to checkout.
Failed end condition	The customer is unable to find the desired agri-product or the customer's search criteria are too specific, resulting in no matching results.
Primary Actor	Customer
Secondary Actor	Admin
Trigger	The customer clicks on the search button and enters their search criteria.
Main Success Flows	1. Customer enters their search criteria into the platform's search input field. 2. The platform displays a list of matching products. 3. Customer views detailed information about the desired product. 4. Customer adds the desired product to their cart and proceeds to checkout.
Alternative Flow	2a) Customer navigates to different categories or sections of the platform to find the desired product. 3a) Customer contacts customer support for assistance with finding the desired product.
Quality Requirements	1. The system must display the search results within 3 seconds of the user submitting the search query.

Use Case	Search Product
	2. The platform must provide relevant and up-to-date information about each product. 3. The platform must provide clear and detailed product descriptions, images, and specifications.

UC-06: View Recommend Products

Use Case	View Recommended Products
Goal	The goal of this use case is to allow the customer to view recommended agri-products on the platform.
Preconditions	The customer must have a valid account on the plaedtform and the platform must have a sufficient number of agri-products available for recommendation.
Success end condition	The customer successfully views the recommended agri-products.
Failed end condition	The customer is unable to view the recommended agri-products or the recommended products are not relevant to the customer’s preferences.
Primary Actor	Customer
Secondary Actor	Admin
Trigger	The customer clicks on the “View Recommended Products” button.
Main Success Flows	1. The platform displays a list of recommended agri-products for the customer. 2. Customer views detailed information about the recommended products. 3. Customer adds a recommended product to their cart and proceeds to checkout.
Alternative Flow	3a)The customer can choose to ignore the recommended products and continue searching for other products.
Quality Requirements	1. The platform must provide relevant and up-to-date recommendations based on the customer’s previous purchases and viewing history. 2. The platform must provide clear and detailed product descriptions, images, and specifications for each recommended product. 3. The platform’s recommendation system must be fast and accurate.

UC-07: Add To Cart

Use Case	Add To Cart
Goal	The goal of this use case is to allow the customer to add agri-products to their shopping cart.
Preconditions	The customer must have a valid account on the platform and have selected an agri-product to purchase.
Success end condition	The customer successfully adds the selected agri-product to their shopping cart.
Failed end condition	The customer is unable to add the selected agri-product to their cart or the platform experiences an error during the add to cart process.
Primary Actor	Customer
Secondary Actor	Payment Gateway
Trigger	The customer clicks on the "Add to Cart" button for the selected agri-product.
Main Success Flows	1. The platform updates the customer's shopping cart with the selected agri-product. 2. The customer is able to view the updated shopping cart. 3. The customer can choose to proceed to checkout or continue shopping.

Use Case	**Add To Cart**
Alternative Flow	3a) The customer may cancel the add to cart operation if they change their mind about purchasing the selected product.
Quality Requirements	1. The platform must provide clear and accurate information about the selected agri-product and its availability. 2. The platform must accurately update the customer's shopping cart. 3. The platform must be able to handle multiple add to cart operations simultaneously without errors. 4. The platform must provide real-time updates to the customer's shopping cart.

UC-08: Authenticate Customer

Use Case	Authenticate Customer
Goal	The goal of this use case is to authenticate the customer's identity to ensure secure access to their account and personal information.
Preconditions	The customer must have created an account on the platform.
Success end condition	The customer successfully logs into their account using their login credentials.
Failed end condition	The customer is unable to log in due to incorrect login credentials or the platform experiences an error during the authentication process.
Primary Actor	Customer
Secondary Actor	Admin
Trigger	The customer clicks on the "Login" button and enters their login credentials.
Main Success Flows	1. The platform validates the customer's login credentials. 2. The platform grants access to the customer's account and personal information. 3. The customer is able to view their account information and access the platform's features and services.
Alternative Flow	2a) The customer may choose to reset their password if they have forgotten their login credentials.
Quality Requirements	1. The platform must use secure methods for storing and transmitting login credentials. 2. The platform must provide clear and accurate error messages in case of incorrect login credentials. 3. The platform must provide real-time updates to the customer's account information. 4. The platform must ensure the confidentiality of the customer's personal information.

UC-09: Search Category

Use Case	Search Category
Goal	The goal of this use case is to allow customers to search for products in specific categories in the E-Fram System.
Preconditions	The customer must have access to the platform and have basic knowledge of the available product categories.
Success end condition	The customer is able to view a list of products within the selected category, filtered by their desired criteria (e.g. price, availability, etc.).
Failed end condition	The customer is unable to search for products due to technical issues or a lack of relevant products in the desired category.
Primary Actor	Customer
Secondary Actor	System
Trigger	The customer selects the "Category Search" option and selects the desired category.
Main Success Flows	1. The platform displays a list of subcategories within the selected category. 2. The customer selects a subcategory and the platform displays a list of relevant products. 3. The customer is able to filter the products based on desired criteria.
Alternative Flow	3a) The customer may choose to navigate back to the main product catalog or search for products using a different method (e.g. keyword search).

Use Case	Search Category
Quality Requirements	1. The system must display the targeted search results within 3 seconds of the user submitting the search category. 2. The platform must allow customers to easily navigate between categories and subcategories. 3. The platform must provide accurate and up-to-date information on product availability. 4. The platform must respond quickly to category search requests and display the results in a user-friendly manner. 5. The platform must provide relevant product filters to help customers quickly find what they are looking for.

UC-10: Show Products Availability

Use Case	Show Products Availability
Goal	To show the availability of products in the Agri Product Delivery System
Preconditions	1. The customer has selected a product in the system. 2. The product information and availability data is stored in the system database.
Success end condition	The customer can view the availability of the selected product in the system
Failed end condition	The customer is unable to view the availability of the selected product in the system
Primary Actor	Customer
Secondary Actor	Admin
Trigger	The customer selects a product in the system.
Main Success Flows	1. The customer selects a product in the system 2. The system retrieves the product information and availability data from the database 3. The system displays the availability information of the selected product to the customer
Alternative Flow	3a) If the product availability data is not found in the database, the system displays an error message indicating that the product information is not available
Quality Requirements	1. The availability information should be accurate and up-to-date 2. The availability information should be displayed in a clear and concise manner 3. The system must display the availability of the searched product within 3 seconds of the user selecting the product.

UC-11: Select Delivery Types

Use Case	Select Delivery Types
Goal	To allow the customer to select a delivery type for their agricultural product order.
Preconditions	1. Customer has successfully placed an order for agricultural products. 2. Delivery options are available for the customer's location.
Success end condition	The customer has selected a delivery type and the order is updated with the selected delivery information.
Failed end condition	The customer is unable to select a delivery type due to unavailability or errors in the delivery options.

Use Case	Select Delivery Types
Primary Actor	Customer
Secondary Actor	Delivery System
Trigger	Customer selects the option to choose a delivery type while placing an order.
Main Success Flows	1. Customer selects a delivery type. 2. System confirms the selected delivery type. 3. System updates the order with the selected delivery type.
Alternative Flow	1a) Delivery options are unavailable for the customer's location. 2a) System informs the customer that delivery options are not available.
Quality Requirements	1. Delivery options should be accurate and up-to-date. 2. The system should provide clear information about delivery options and fees. 3. The selection process should be user-friendly and efficient. 4. The products should be delivered with in 6h.

UC-12: Select Payment Types

Use Case	Select Payments Types
Goal	To allow the customer to select a payment type for their agricultural product order.
Preconditions	• Customer has successfully placed an order for agricultural products. • Payment options are available for the customer's location.
Success end condition	The customer has selected a payment type and the order is updated with the selected payment information.
Failed end condition	The customer is unable to select a payment type due to unavailability or errors in the payment options.
Primary Actor	Customer
Secondary Actor	Admin
Trigger	Customer selects the option to choose a payment type while placing an order.
Main Success Flows	1. Customer selects a payment type. 2. System confirms the selected payment type. 3. System updates the order with the selected payment type.
Alternative Flow	1a) Payment options are unavailable for the customer's location. 3a) System informs the customer that payment options are not available.
Quality Requirements	1. Payment options should be secure and reliable. 2. The system should provide clear information about payment options and fees. 3. The selection process should be user-friendly and efficient.

UC-13: Deliver Products

Use Case	Deliver Products
Goal	To deliver the agricultural product to the customer
Preconditions	1. Customer has successfully placed an order for agricultural products. 2. Payment for the order has been processed. 3. Delivery information has been confirmed and updated.
Success End Condition	The agricultural product has been successfully delivered to the customer.

Use Case	Deliver Products
Failed End Condition	The agricultural product delivery fails due to errors or unavailability of resources.
Primary Actor	Admin
Secondary Actor	Customer, delivery personnel
Trigger	The delivery system receives a request to deliver the agricultural product to the customer.
Main Success Flow	1. Delivery system dispatches delivery personnel to the customer's location. 2. Delivery personnel deliver the agricultural product to the customer. 3. Customer confirms receipt of the product. 4. Delivery system updates the delivery status as successful.
Alternative Flow	1a) Delivery personnel is unable to locate the customer's location. 2. Delivery system updates the delivery status as failed and reschedules delivery.
Quality Requirements	1. Delivery personnel should be professional and well-trained. 2. Delivery should be timely and efficient. 3. Delivery personnel should handle the product with care to avoid damage. 4. The system should provide real-time tracking information to the customer.

UC-13: Track Order

Use Case	Track Order
Goal	To allow customers to track the delivery status of their agricultural product orders.
Preconditions	1. Customer has successfully placed an order for agricultural products. 2. Payment for the order has been processed.
Success End Condition	The customer can view the current delivery status of their order.
Failed End Condition	The system fails to provide accurate or timely delivery status information.
Primary Actor	Customer
Secondary Actor	Admin
Trigger	The customer wants to track the delivery status of their order.
Main Success Flow	1. Customer accesses the system and selects the "Track Order" option. 2. System retrieves the current delivery status of the order. 3. System displays the delivery status to the customer.
Alternative Flow	2a) Delivery personnel encounters issues that affect the delivery status update, such as traffic delays or mechanical problems. 3a) System updates the estimated delivery time and informs the customer of the delay.
Quality Requirements	1. Delivery status should be accurate and updated in real-time. 2. System should provide clear and concise delivery status information to the customer. 3. System should provide timely updates if there are any delays or issues with the delivery. 4. The system should be accessible and easy to use for the customer.

UC-13: Submit Feedback

Use Case	Submit Feedback
Goal	To allow customers to submit feedback on the quality of the product, delivery service, or overall experience.
Preconditions	The customer has received their order and wants to provide feedback.
Success End Condition	The system receives the customer's feedback and updates the appropriate records.
Failed End Condition	The system does not receive the customer's feedback or the feedback is incomplete or inaccurate.
Primary Actor	Customer
Secondary Actor	Admin
Trigger	The customer wants to submit feedback on their order or delivery experience.
Main Success Flow	1. Customer accesses the system and selects the "Submit Feedback" option. 2. System prompts the customer to enter feedback on the product, delivery service, or overall experience. 3. Customer enters feedback and submits it. 4. System updates the appropriate records with the customer's feedback.
Alternative Flow	N/A
Quality Requirements	1. System should provide clear and easy-to-use options for submitting feedback. 2. Feedback forms should be concise and easy to understand. 3. The system should store and track customer feedback and use it to improve the product and delivery service. 4. The system should have a way to follow up with customers if necessary.

UC-13: Update Profile

Use Case	Update Profile
Goal	To allow customers to update their profile information, including personal details and delivery preferences.
Preconditions	The customer has an existing profile in the system and wants to update their information.
Success End Condition	The system updates the customer's profile with the new information.
Failed End Condition	The system fails to update the customer's profile, or the updated information is incomplete or inaccurate.
Primary Actor	Customer
Secondary Actor	N/A
Trigger	The customer wants to update their profile information.
Main Success Flow	1. Customer accesses the system and selects the "Update Profile" option. 2. System prompts the customer to enter their updated profile information. 3. Customer enters their updated information and submits it. 4. System updates the customer's profile with the new information.
Alternative Flow	N/A
Quality Requirements	1. System should provide clear and easy-to-use options for updating profile information.

Use Case	Update Profile
	2. Profile forms should be concise and easy to understand. 3. The system should store and track customer profile information accurately. 4. The system should have a way to confirm changes with the customer.

UC-17: Update Address

Use Case	Update Address
Goal	To allow customers to update their delivery address.
Preconditions	The customer has an existing address in the system and wants to update it.
Success End Condition	The system updates the customer's address with the new information.
Failed End Condition	The system fails to update the customer's address, or the updated information is incomplete or inaccurate.
Primary Actor	Customer
Secondary Actor	N/A
Trigger	The customer wants to update their delivery address.
Main Success Flow	1. Customer accesses the system and selects the "Update Address" option. 2. System prompts the customer to enter their updated address information. 3. Customer enters their updated information and submits it. 4. System updates the customer's address with the new information.
Alternative Flow	N/A
Quality Requirements	1. System should provide clear and easy-to-use options for updating address information. 2. Address forms should be concise and easy to understand. 3. The system should store and track customers address information accurately. 4. The system should have a way to confirm changes with the customer.

UC-18: Display Confirmation Report

Use Case	Display Confirmation Report
Goal	To allow the customer to view a confirmation report for their order.
Preconditions	The customer has placed an order and wants to view a confirmation report.
Success End Condition	The system displays the confirmation report for the customer's order.
Failed End Condition	The system is unable to generate a confirmation report or the report is incomplete or inaccurate.
Primary Actor	Customer
Secondary Actor	N/A
Trigger	The customer wants to view a confirmation report for their order.
Main Success Flow	1. Customer accesses the system and selects the "View Confirmation Report" option. 2. System retrieves the customer's order information and generates a confirmation report. 3. System displays the confirmation report for the customer to review.
Alternative Flow	N/A

Use Case	Display Confirmation Report
Quality Requirements	1. The system should generate accurate and complete confirmation reports for all customer orders. 2. The confirmation report should be presented in a clear and easy-to-understand format. 3. The system should be able to retrieve order information with in 5s.

UC-19: Make Payment

Use Case	Make Payment
Goal	To allow customers to make payment for their order.
Preconditions	The customer has selected their products, added them to the cart, and has selected the "Make Payment" option.
Success End Condition	The system accepts payment from the customer and the order is confirmed.
Failed End Condition	The system rejects the payment or the payment is not processed correctly.
Primary Actor	Customer
Secondary Actor	Payment Gateway
Trigger	The customer selects the "Make Payment" option.
Main Success Flow	1. Customer selects "Make Payment" option. 2. System redirects the customer to the payment gateway to process payment. 3. Customer enters payment information (e.g., credit card number) on the payment gateway. 4. Payment gateway processes the payment and sends a response to the system. 5. System confirms the payment and finalizes the order.
Alternative Flow	1a) The payment gateway is not available or offline. 2a) The payment is rejected or not processed correctly. 3a) The customer cancels the payment or closes the payment gateway window.
Quality Requirements	1. The payment gateway should be secure and reliable. After complete payment, dashboard will be redirected within 2s. 2. The payment gateway should be integrated with the system for a seamless payment experience. 3. The system should provide clear and concise payment options for customers. 4. The system should provide an error message for failed payments.

UC-20: View Agri Information

Use Case	Search Agri Information
Goal	To allow farmers to search information related to their agriculture products.
Preconditions	The farmer has logged into the system and searched the "View Agri Information" option.
Success End Condition	The system displays the agriculture information related to the farmer's products.
Failed End Condition	The system is unable to display the agriculture information due to technical issues or the information is not available.

Use Case	Search Agri Information
Primary Actor	Farmer
Secondary Actor	N/A
Trigger	The farmer selects the "Search Agri Information" option.
Main Success Flow	1. Farmer selects "Search Agri Information" option. 2. System retrieves the agriculture information related to the farmer's products. 3. System displays the agriculture information to the farmer.
Alternative Flow	2a) The agriculture information related to the farmer's products is not available. 3a) Technical issues prevent the system from displaying the agriculture information.
Quality Requirements	1. The system should provide accurate and up-to-date agriculture information with 3s after searching. 2. The system should be user-friendly and easy to navigate for farmers. 3. The system should be secure and protect farmers' information.

UC-21: Take Consultancy Service

Use Case	Take Consultancy Service
Goal	To allow farmers to access consultancy services for their agriculture products.
Preconditions	The farmer has logged into the system and has selected the "Consultancy Service" option.
Success End Condition	The system provides the necessary consultancy services related to the farmer's products.
Failed End Condition	The system is unable to provide the consultancy services due to technical issues or the services are not available.
Primary Actor	Farmer
Secondary Actor	Admin
Trigger	The farmer selects the "Consultancy Service" option.
Main Success Flow	1. Farmer selects "Consultancy Service" option. 2. System retrieves the list of available consultancy services. 3. Farmer selects a consultancy service. 4. System connects the farmer to a consultant. 5. Consultant provides necessary consultancy services related to the farmer's products.
Alternative Flow	3a) The consultancy service selected by the farmer is not available. 5a) Technical issues prevent the system from providing the consultancy services.
Quality Requirements	1. The system should provide a range of consultancy services related to agriculture products with 24/7. 2. The system should connect farmers with experienced and knowledgeable consultants. 3. The system should be user-friendly and easy to navigate for farmers. 4. The system should be secure and protect farmers' information.

UC-22: Rent Agri Tools

Use Case	Rent Agri Tools
Goal	To allow farmers to rent agricultural tools for their farming activities.

Use Case	Rent Agri Tools
Preconditions	The farmer has logged into the system and has selected the “Rent Agri Tools” option.
Success End Condition	The system provides the necessary tools to the farmer for a rental period.
Failed End Condition	The system is unable to provide the tools due to technical issues or the tools are not available.
Primary Actor	Farmer
Secondary Actor	Agri Tool Rental Service Provider
Trigger	The farmer selects the “Rent Agri Tools” option.
Main Success Flow	1. Farmer selects “Rent Agri Tools” option. 2. System retrieves the list of available tools. 3. Farmer selects the tool and rental period. 4. System connects the farmer to the rental service provider. 5. Rental service provider provides the tools to the farmer for the rental period.
Alternative Flow	3a) The tool selected by the farmer is not available. 4a) Technical issues prevent the system from providing the tools. 5a) The farmer is unable to return the tools on time.
Quality Requirements	1. The system should provide a range of agricultural tools for rental with in 30minutes after selecting for renting. 2. The system should connect farmers with reliable and quality rental service providers. 3. The system should be user-friendly and easy to navigate for farmers. 4. The system should be secure and protect farmers’ information.

UC-23: Report Issues

Use case	Report Issues
Goal	To allow farmers to report any issues they encounter during their use of the Agri Product Delivery System.
Preconditions	The farmer has logged into the system and has encountered an issue that needs to be reported.
Success End Condition	The system receives the issue report and takes appropriate action to resolve the issue.
Failed End Condition	The system does not receive the issue report, or the issue is not resolved.
Primary Actor	Farmer
Secondary Actor	Admin
Trigger	The farmer encounters an issue that needs to be reported.
Main Success Flow	1. Farmer selects "Report Issue" option. 2. System directs the farmer to a form for issue reporting. 3. Farmer fills out the form with details of the issue. 4. System receives the issue report and sends a confirmation to the farmer. 5. Support team reviews the issue report and takes appropriate action to resolve the issue.
Alternative Flow	2a) The system does not receive the issue report due to technical issues. 4a) The support team is unable to resolve the issue.

Use case	Report Issues
Quality Requirements	1. The system should have a user-friendly and accessible issue reporting process. 2. The system should have a responsive and effective support team to address reported issues. 3. The system should have a mechanism to track reported issues and ensure they are resolved in a timely manner. 4. The system should communicate with farmers about the status of their reported issues.

UC-24: Upload Product Details

Use Case	Upload Product Details
Goal	Allow farmers to upload product details on the Agri Product Delivery System to sell their products directly to customers.
Preconditions	The farmer must be registered and logged into the Agri Product Delivery System. The farmer must have products available for sale.
Success End Condition	The product details are successfully uploaded to the Agri Product Delivery System and are available for purchase by customers.
Failed End Conditions	The product details fail to upload to the Agri Product Delivery System due to technical errors or incorrect input.
Primary Actor	Farmer
Secondary Actor	Admin
Trigger	The farmer decides to upload product details for sale on the Agri Product Delivery System.
Main Success Flows	1. Farmer selects the "Upload Product Details" option 2. System prompts the Farmer to enter product details 3. Farmer enters product details such as product name, category, description, quantity, price, and images 4. Farmer confirms the product details and submits them to the system 5. System verifies the product details and updates the database 6. System displays a confirmation message to the Farmer
Alternatives Flows	3a. Farmer selects an option to upload product images 3b. Farmer uploads product images to the system 5a. System detects invalid or missing data and prompts the Farmer to correct the data
Quality Requirements	The system must ensure the security and privacy of the farmer's information and validate the input to avoid errors and ensure the accuracy of the product details. Uploaded Products details will be uploaded within 3s after uploading.

UC25: Request Membership

Use Case	Request Membership
Goal	Allow farmers to request membership on the Agri Product Delivery System to sell their products directly to customers.
Preconditions	The farmer must have an email address and be a registered farmer.

Use Case	**Request Membership**
Success End Condition	The farmer's membership request is approved and they are granted access to sell products on the Agri Product Delivery System.
Failed End Conditions	The farmer's membership request is denied due to incomplete or incorrect information, or the farmer is not eligible for membership.
Primary Actor	Farmer
Secondary Actor	Admin
Trigger	The farmer decides to request membership on the Agri Product Delivery System.
Main Success Flows	1. The farmer selects "Request Membership" on the Agri Product Delivery System. 2. The system prompts the farmer to input their personal information, including their name, address, contact information, and type of products they sell. 3. The farmer enters their personal information and submits the membership request. 4. The system verifies the eligibility of the farmer and approves the membership request. 5. The farmer is granted access to sell their products on the Agri Product Delivery System.
Alternative Flows	1a) If the farmer's membership request is denied, the system displays a message explaining the reason and prompts the farmer to update their information and try again. 3a) If the farmer's eligibility is unclear, the system may require additional information or documentation to make a decision. 4a) If the farmer has previously requested membership, they can check the status of their request and make updates as necessary.
Quality Requirements	The system must ensure the security and privacy of the farmer's information and verify their eligibility to maintain the quality of products on the Agri Product Delivery System.

UC-26: Select Package

Use Case	**Select Package**
Goal	Allow farmers to select a package on the Agri Product Delivery System to sell their products at a discounted commission rate.
Preconditions	The farmer must be registered and logged into the Agri Product Delivery System.
Success End Condition	The farmer selects a package and the discounted commission rate is applied to their product sales.
Failed End Conditions	The farmer is unable to select a package due to technical errors or lack of available packages.
Primary Actor	Farmer
Secondary Actor	Admin
Trigger	The farmer decides to select a package on the Agri Product Delivery System.
Main Success Flows	1. The farmer selects "Select Package" on the Agri Product Delivery System. 2. The system displays a list of available packages with details on the discounted commission rate, duration, and features. 3. The farmer selects the desired package and confirms the selection.

Use Case	**Select Package**
	4. The system applies the discounted commission rate to the farmer's product sales for the duration of the package
Alternative Flows	1a) If there are no available packages, the system displays a message and prompts the farmer to check back at a later time. 3a) If there are technical issues with the package selection, the system displays an error message and prompts the farmer to try again.
Quality Requirements	The system must ensure the accuracy of the package details and apply the discounted commission rate correctly to maintain the trust and satisfaction of farmers.

UC-27: Generate Recommended Products

Use Case	**Generate Recommended Products**
Goal	Generate a list of recommended products based on customer preferences
Preconditions	The customer has logged into the system and provided their preferences and past purchase history
Success End Condition	The system generates a list of recommended products that match the customer's preferences
Failed End Conditions	The system is unable to generate recommended products or generates a list that does not match the customer's preferences
Primary Actor	Customer
Secondary Actor	System
Trigger	The customer indicates they want to view recommended products
Main Success Flows	1. The customer selects the "View Recommended Products" option. 2. The system retrieves the customer's preferences and past purchase history. 3. The system generates a list of recommended products based on the customer's preferences and purchase history. 4. The system displays the list of recommended products to the customer. 5. The customer can view the details of each recommended product and add them to their cart.
Alternative Flows	N/A
Quality Requirements	The system should generate recommended products accurately within 3s. The system should also protect customer data and ensure the privacy of their preferences and purchase history.

8. Activity Diagram

(Create Account)

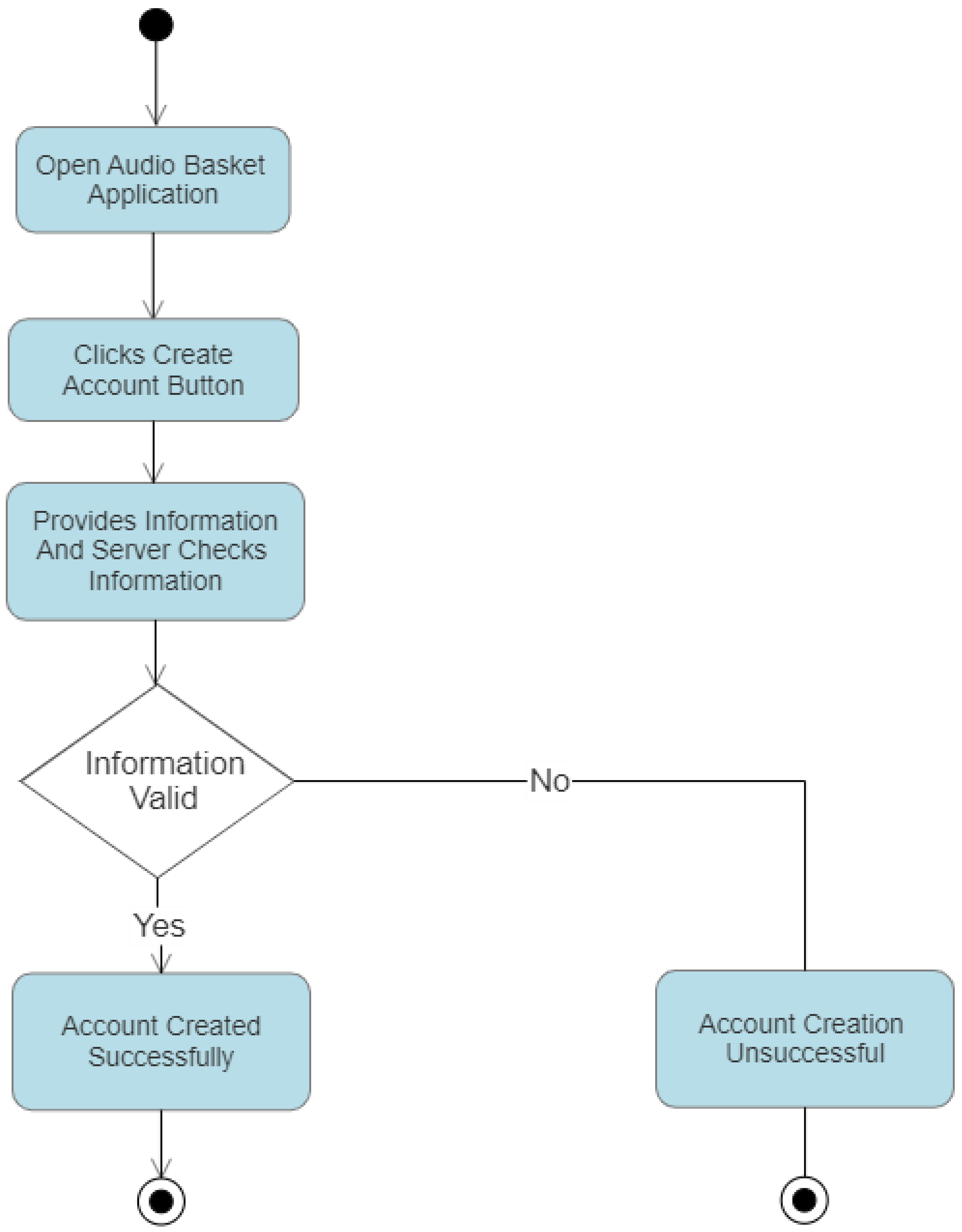

Open Audio Basket Application
Clicks Create Account Button
Provides Information And Server Checks Information
Information Valid
No
Yes
Account Created Successfully
Account Creation Unsuccessful

Figure 2: Create Account

Activity Diagram (Log in)

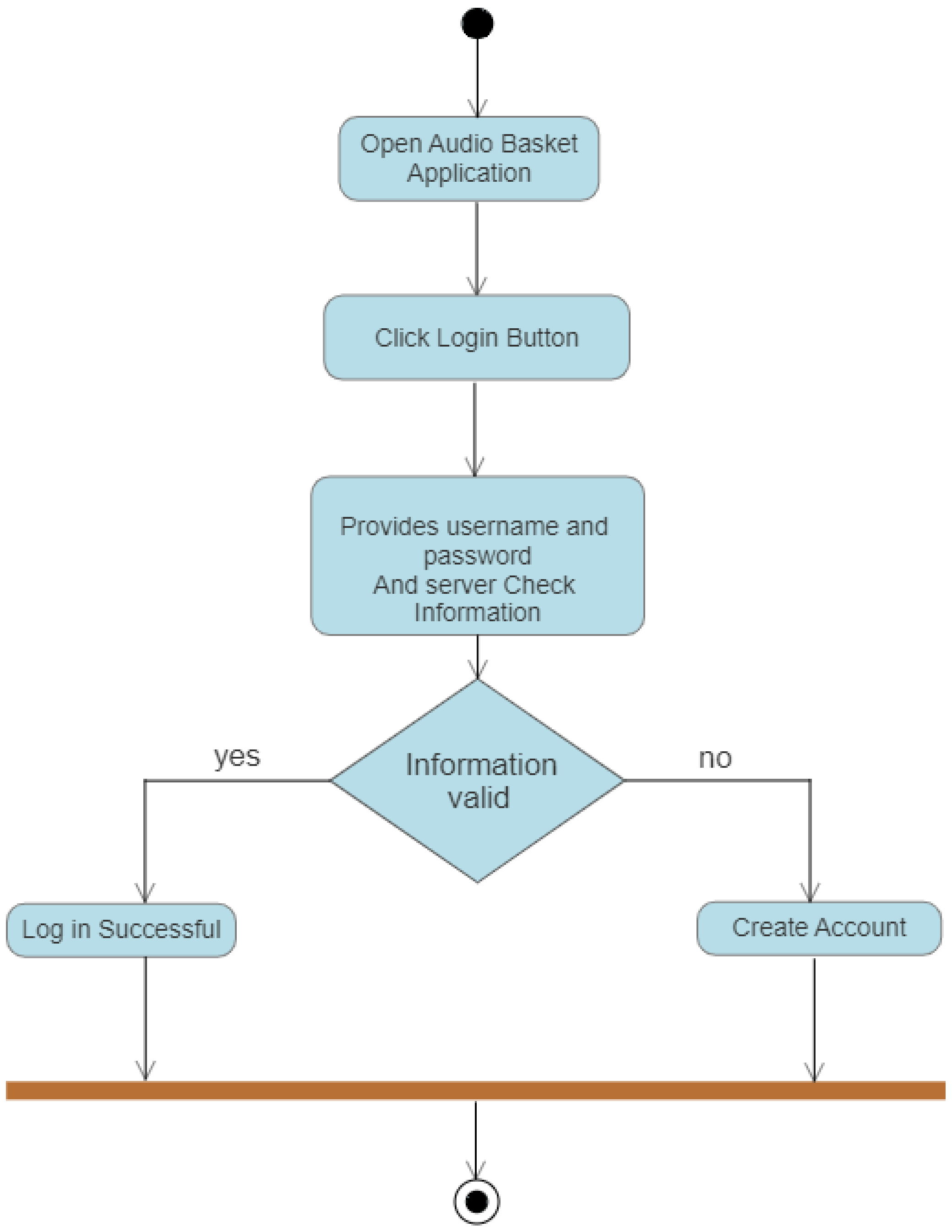

Open Audio Basket Application
Click Login Button
Provides username and password And server Check Information
Information valid
yes
no
Log in Successful
Create Account

Figure 3: Log in

Activity Diagram (Search File)

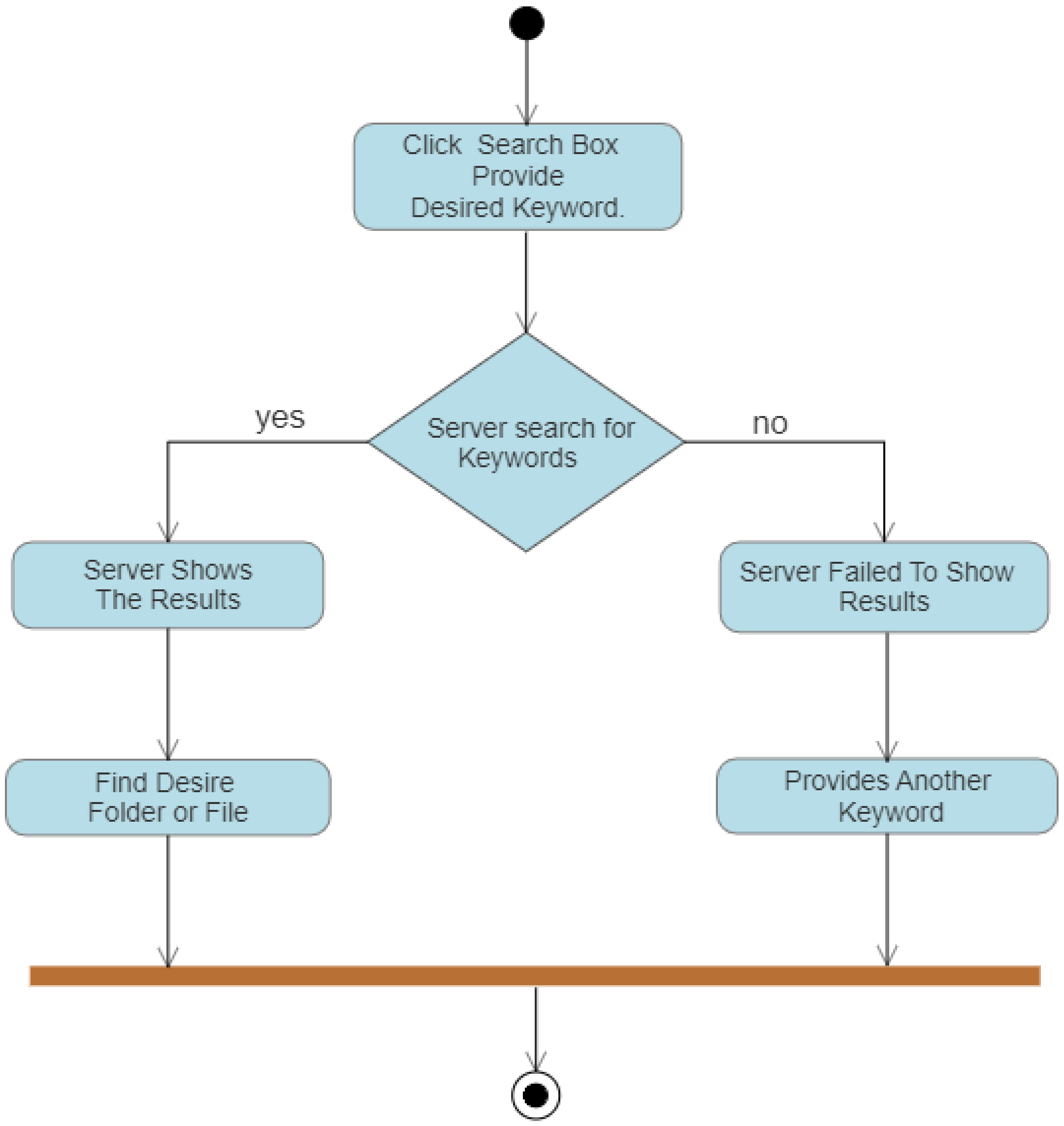
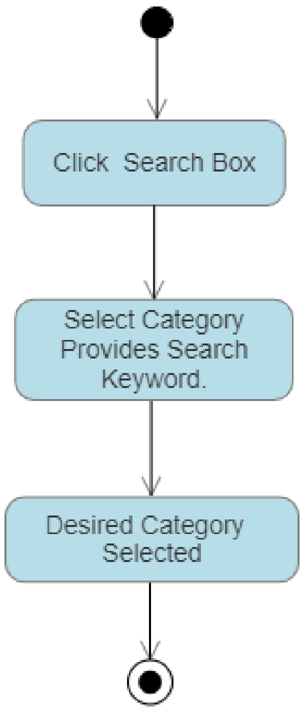


Figure 4: Search File

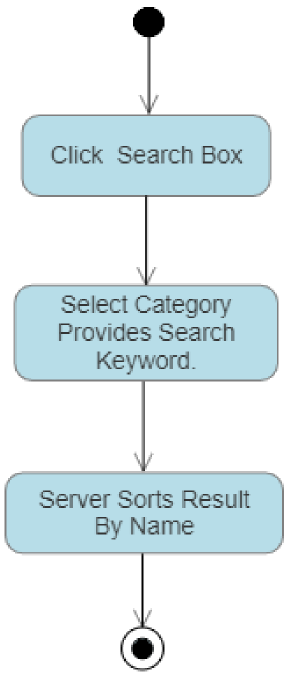
Activity Diagram (Search Category)



Click Search Box
Select Category Provides Search Keyword.
Desired Category Selected

Figure 5:Search Category

Activity Diagram (Sort File)



Click Search Box
Select Category Provides Search Keyword.
Server Sorts Result By Name

Figure 6: Sort File

Activity Diagram (Show Closer Result)

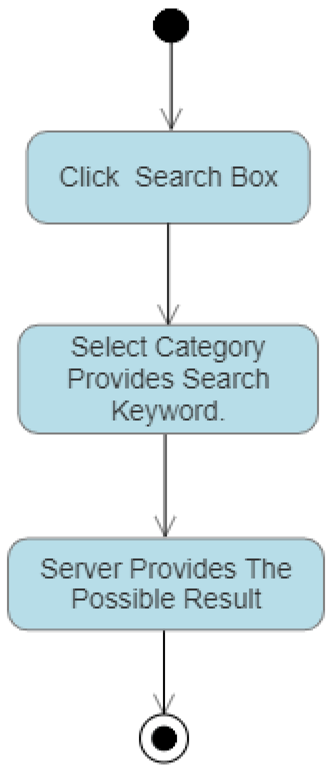

Click Search Box
Select Category Provides Search Keyword.
Server Provides The Possible Result

Figure 7: Show Closer Result

Activity Diagram (Select File)

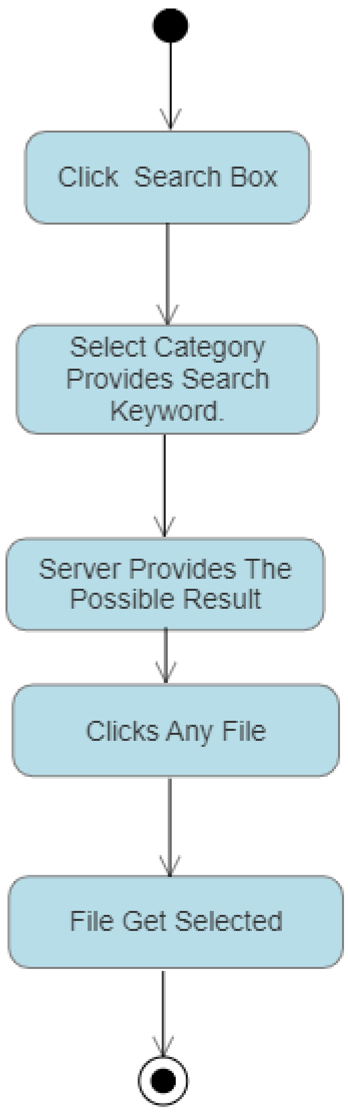

Click Search Box
Select Category Provides Search Keyword.
Server Provides The Possible Result
Clicks Any File
File Get Selected

Figure 8: Select File

Activity Diagram (Share File)

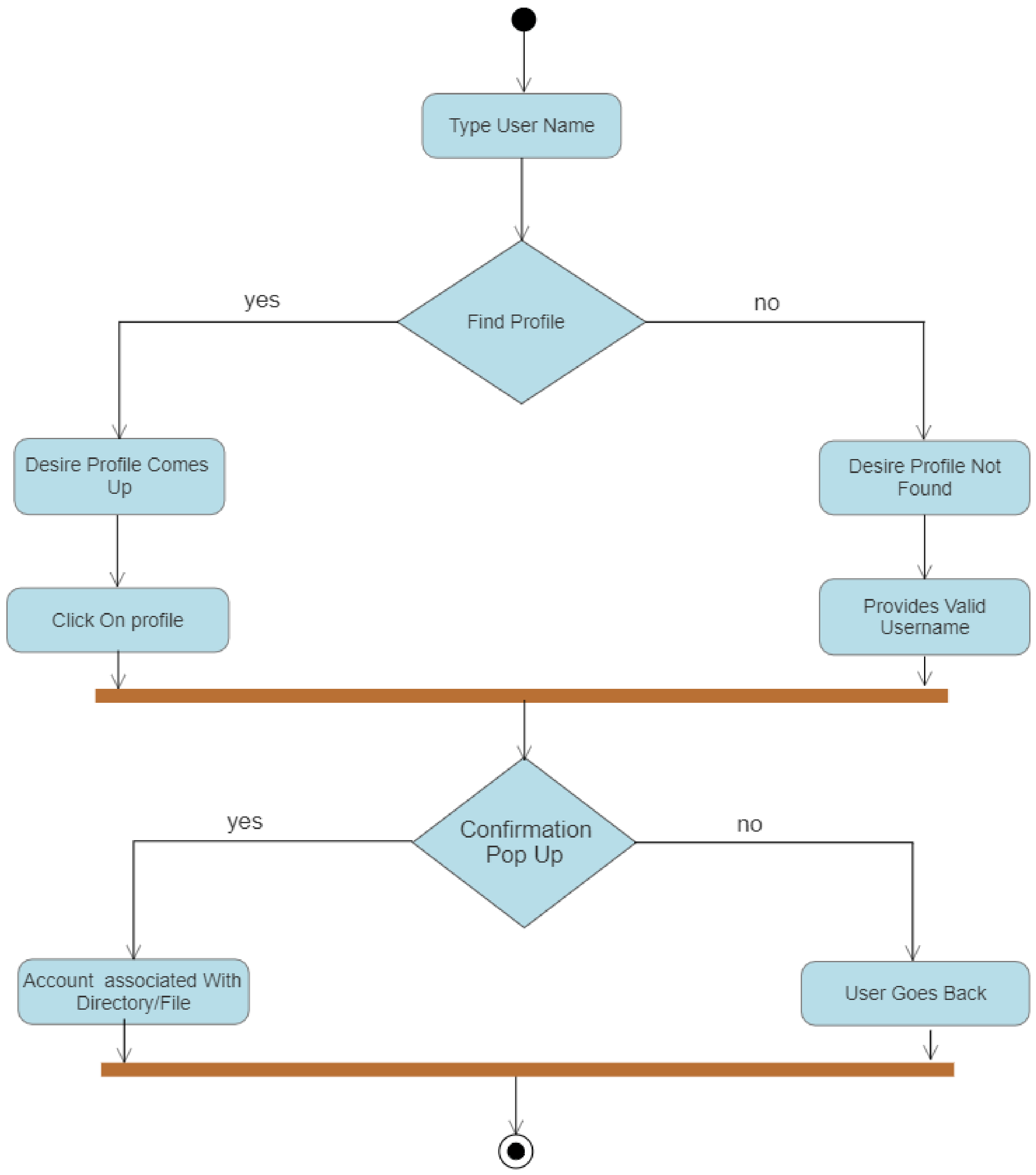


Figure 9: Share File

Activity Diagram (Download File)

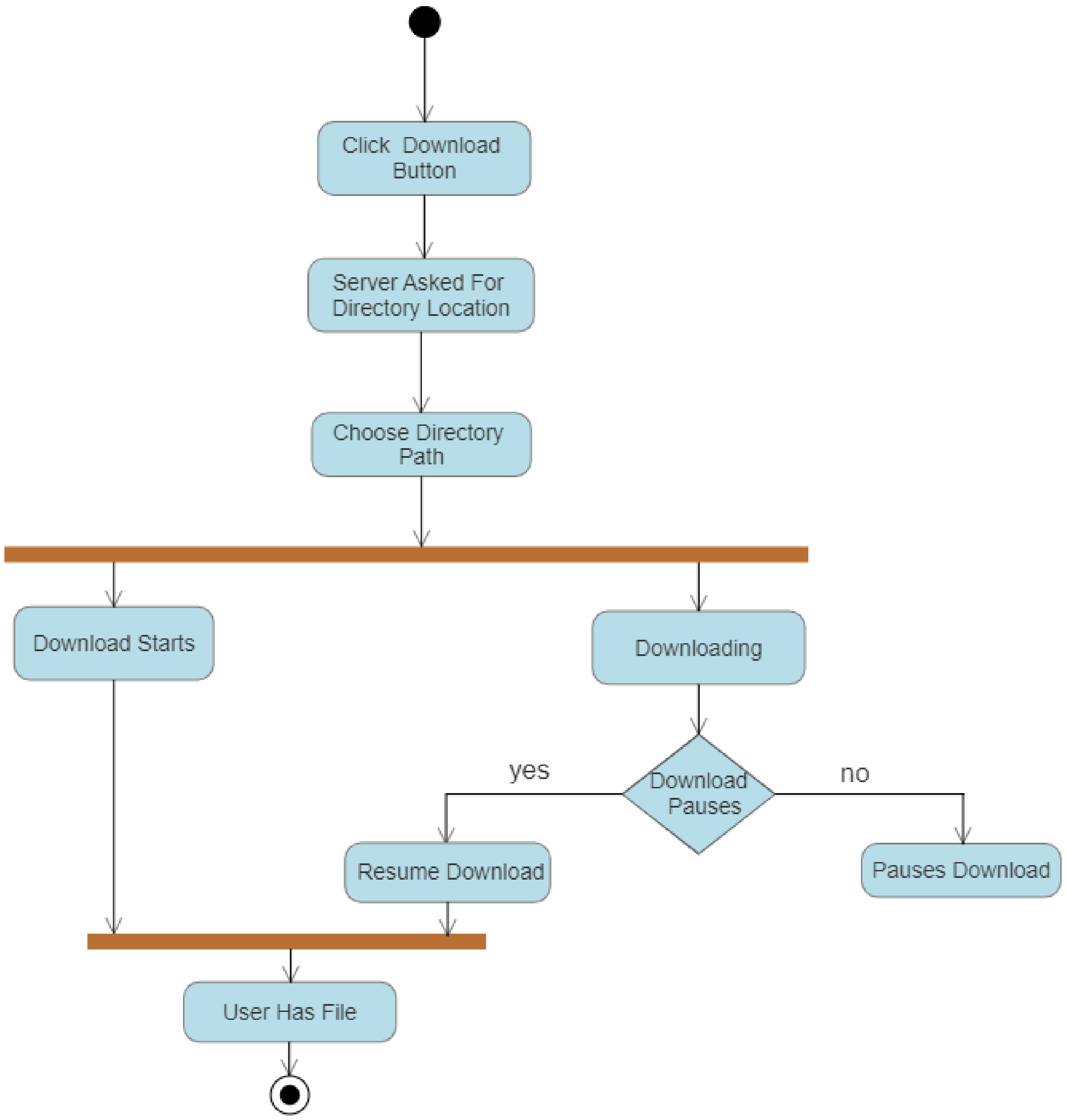


Figure 10: Download File

Activity Diagram (Access Token)

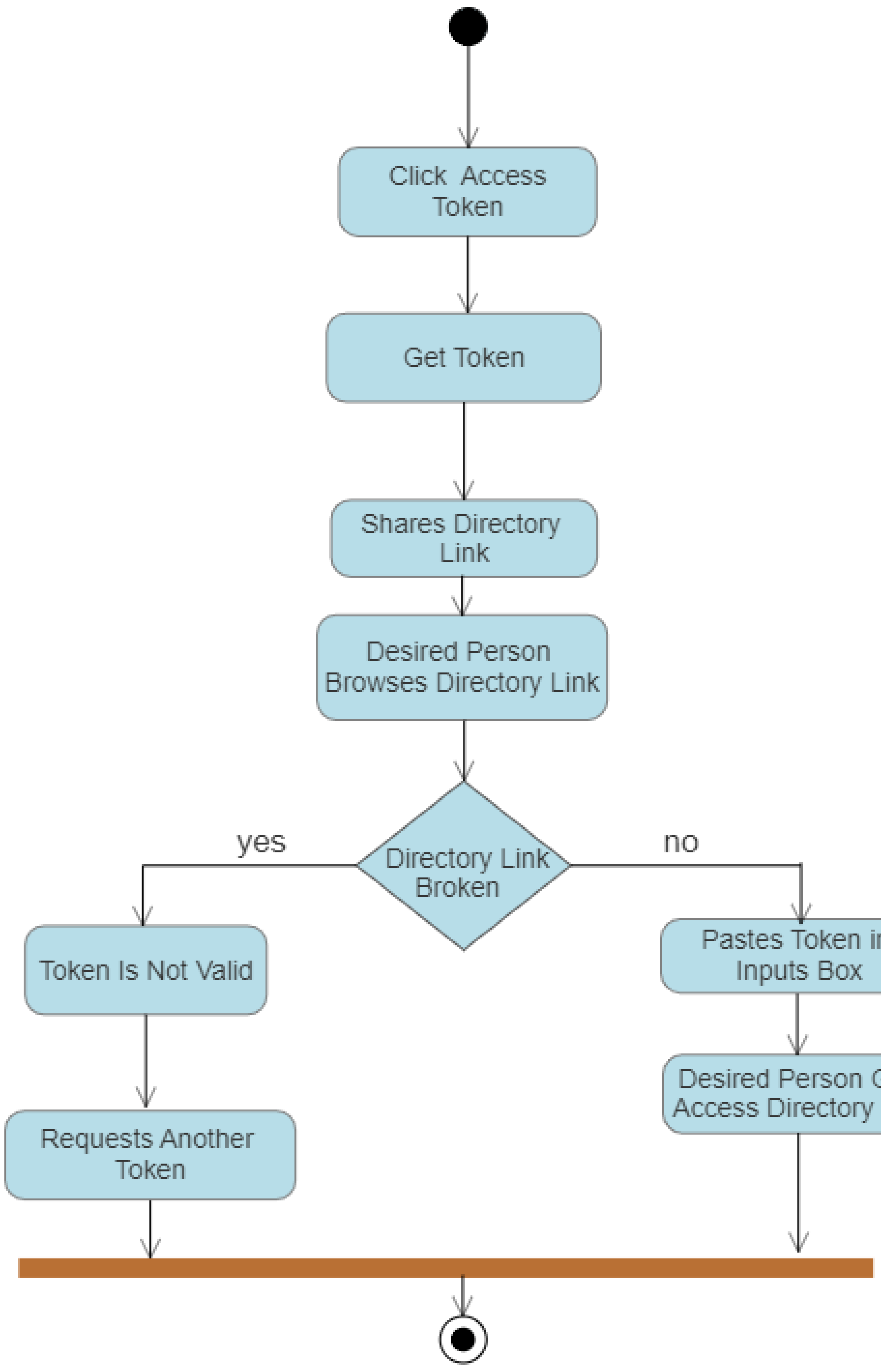


Figure 11: Access Token

Activity Diagram (Add to wishlist)

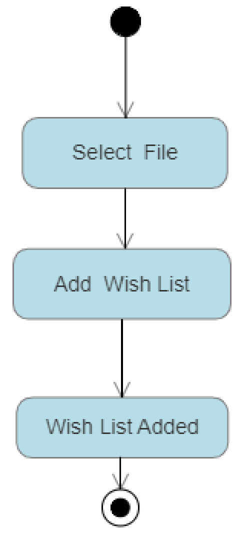


Figure 12: Add to wish list

Activity Diagram (Trash Bin)

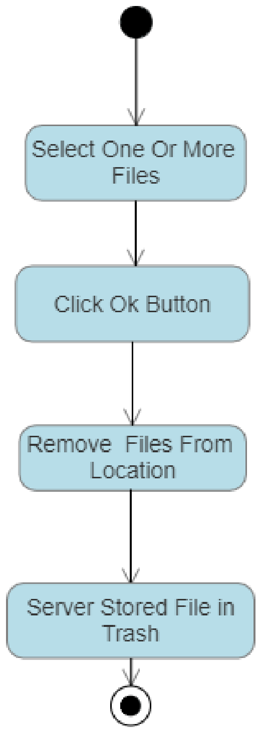


Figure 13: Trash Bin

Activity Diagram (Delete File)

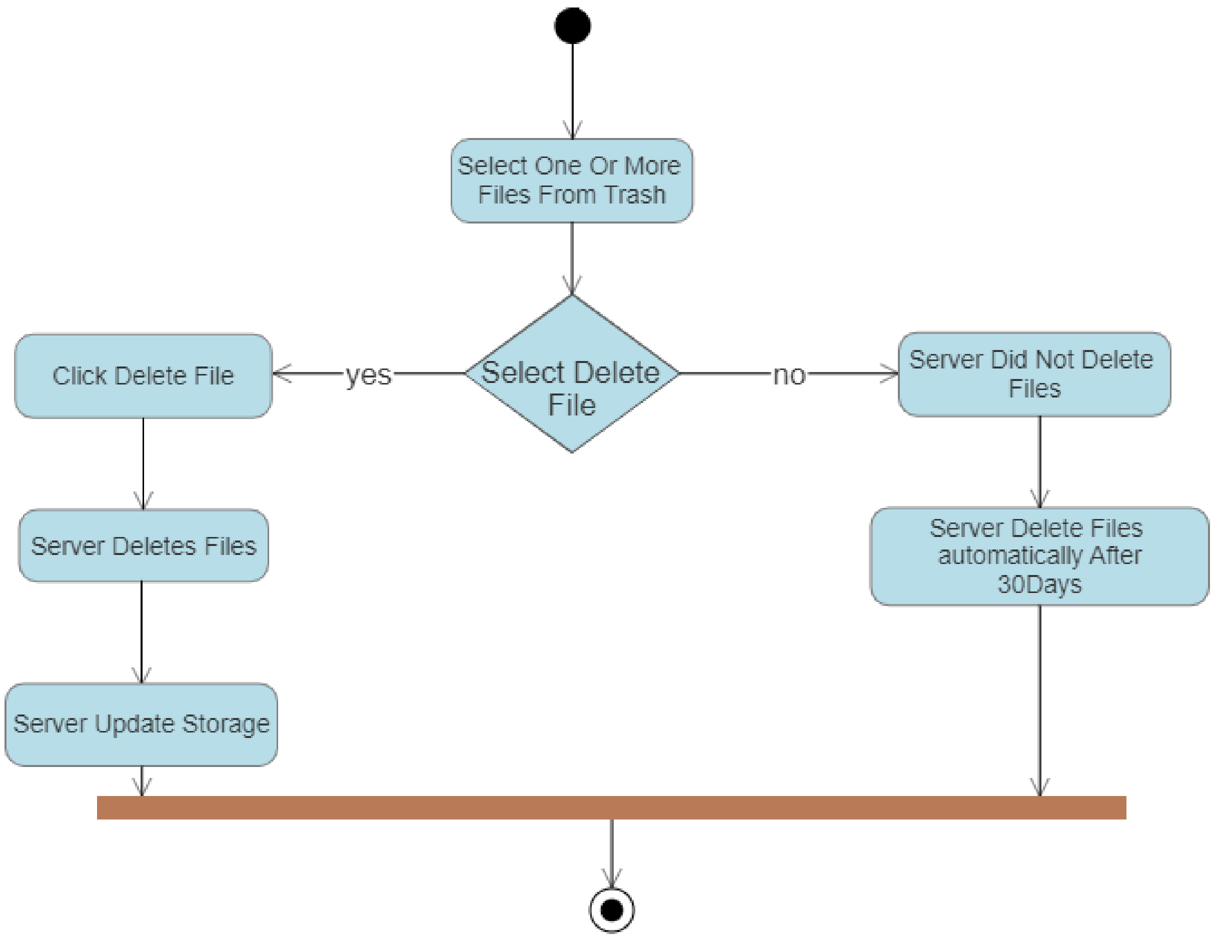

Select One Or More Files From Trash
Select Delete File
yes
no
Click Delete File
Server Did Not Delete Files
Server Deletes Files
Server Delete Files automatically After 30Days
Server Update Storage

Figure 14: Delete File

Activity Diagram (Create Folder)

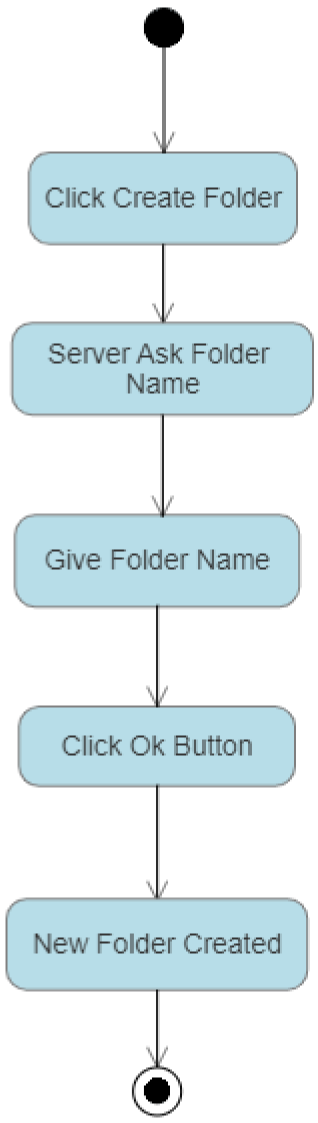


Figure 15: Create Folder

Activity Diagram (Add file)

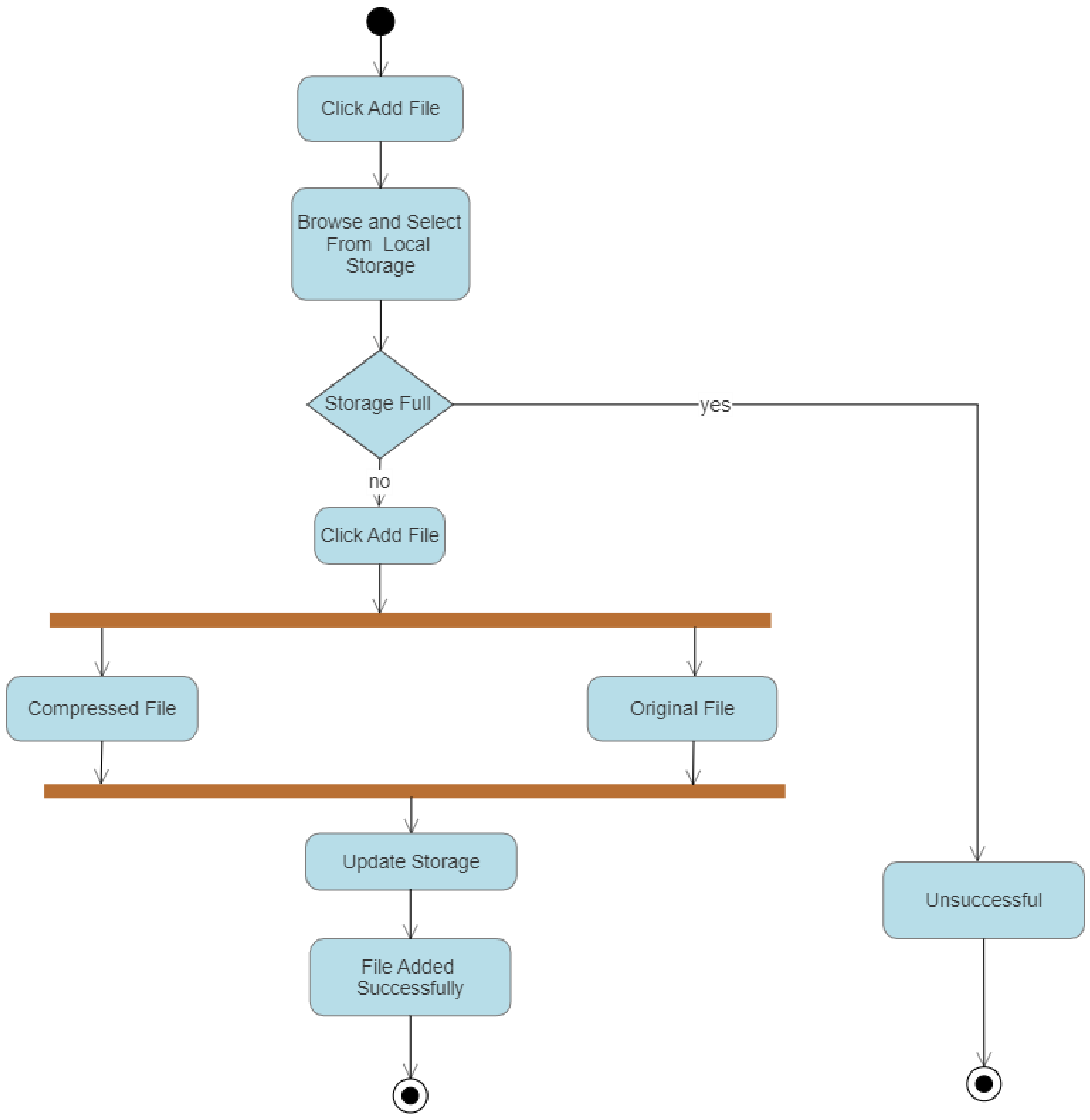


Figure 16: Add File

Activity Diagram (Compress File)

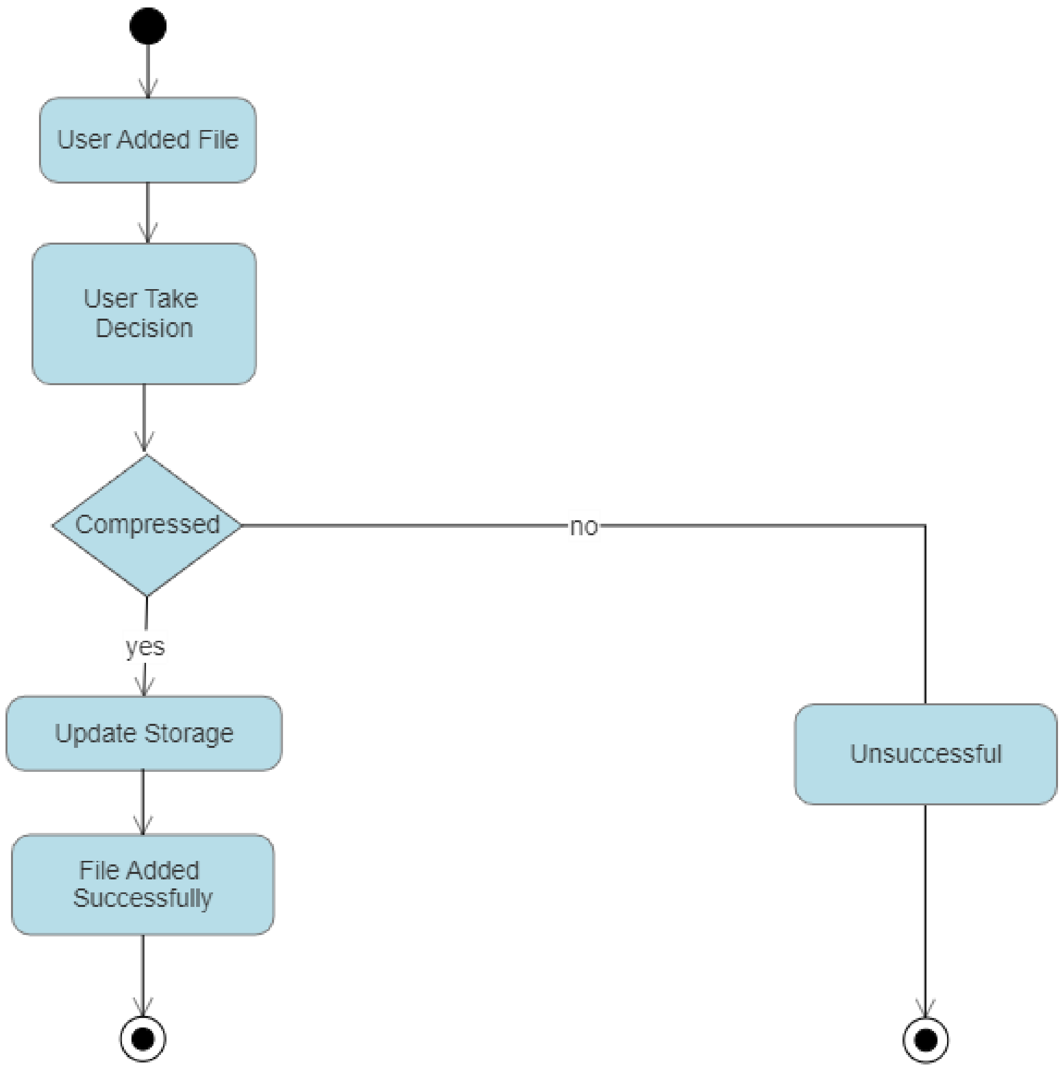


Figure 17: Compress File

Activity Diagram (Check Space)

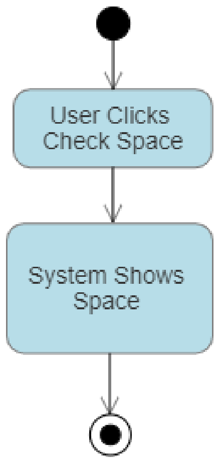


Figure 18: Check Space

Activity Diagram (Payment Information)

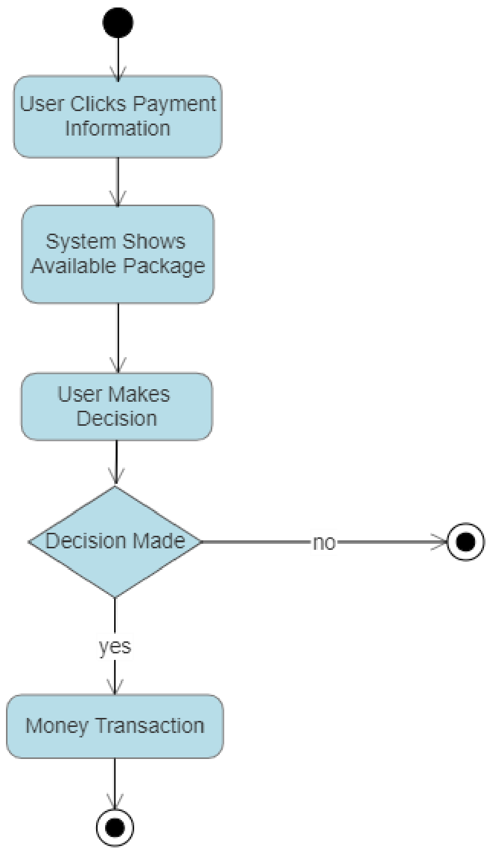

User Clicks Payment Information
System Shows Available Package
User Makes Decision
Decision Made
no
yes
Money Transaction

Figure 19: Payment Information

Activity Diagram (Money Transaction)

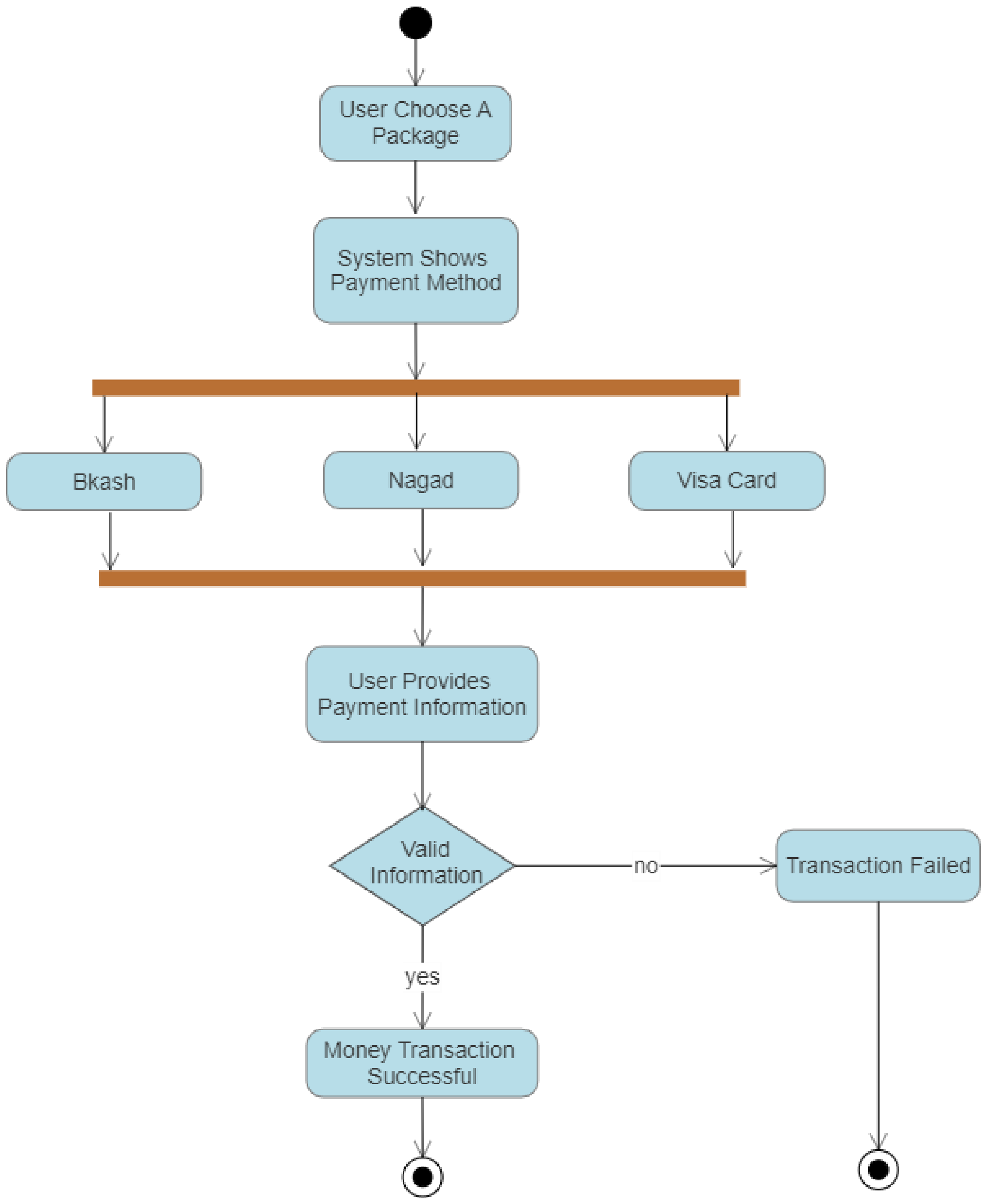

User Choose A Package
System Shows Payment Method
Bkash
Nagad
Visa Card
User Provides Payment Information
Valid Information
no
Transaction Failed
yes
Money Transaction Successful

Figure 20: Money Transaction

Activity Diagram (Update account storage)

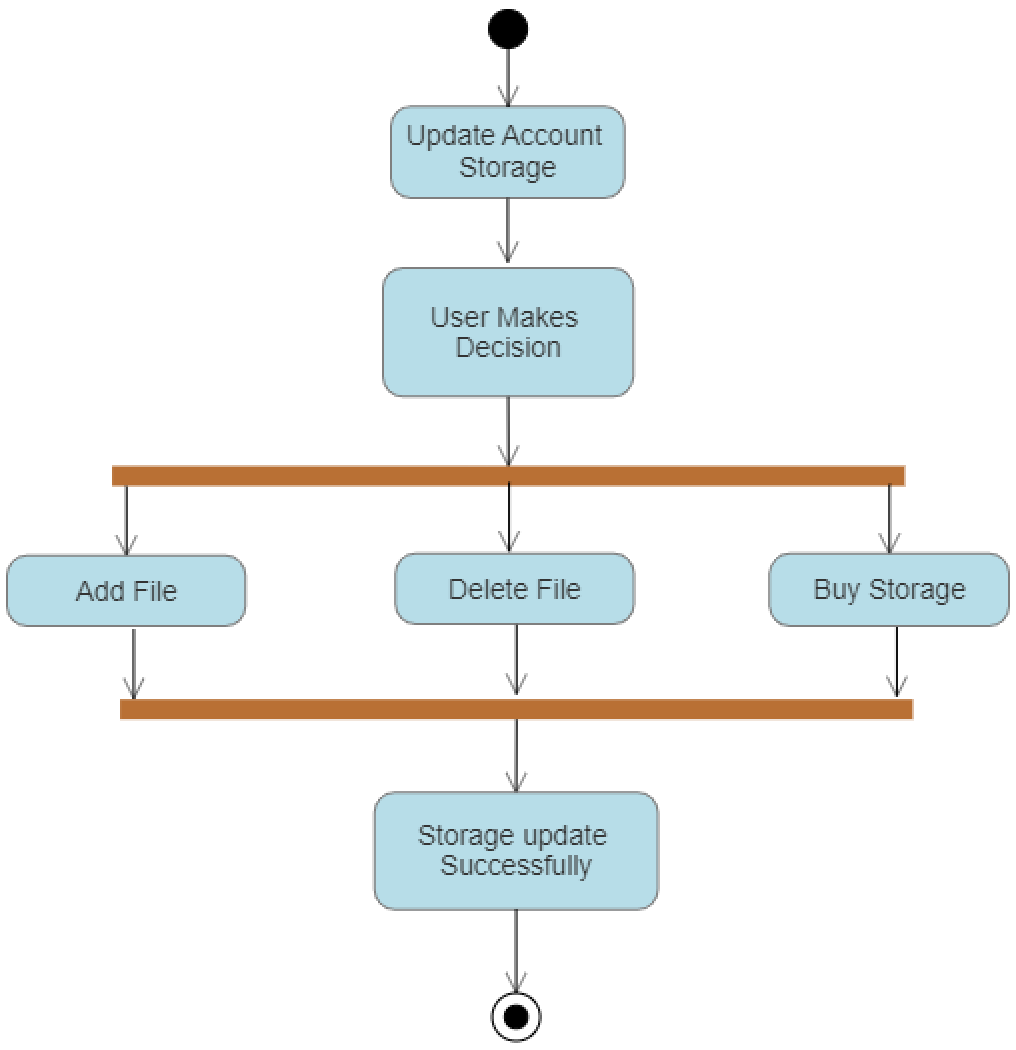


Figure 21: Update Account Storage

Activity Diagram (Play File)

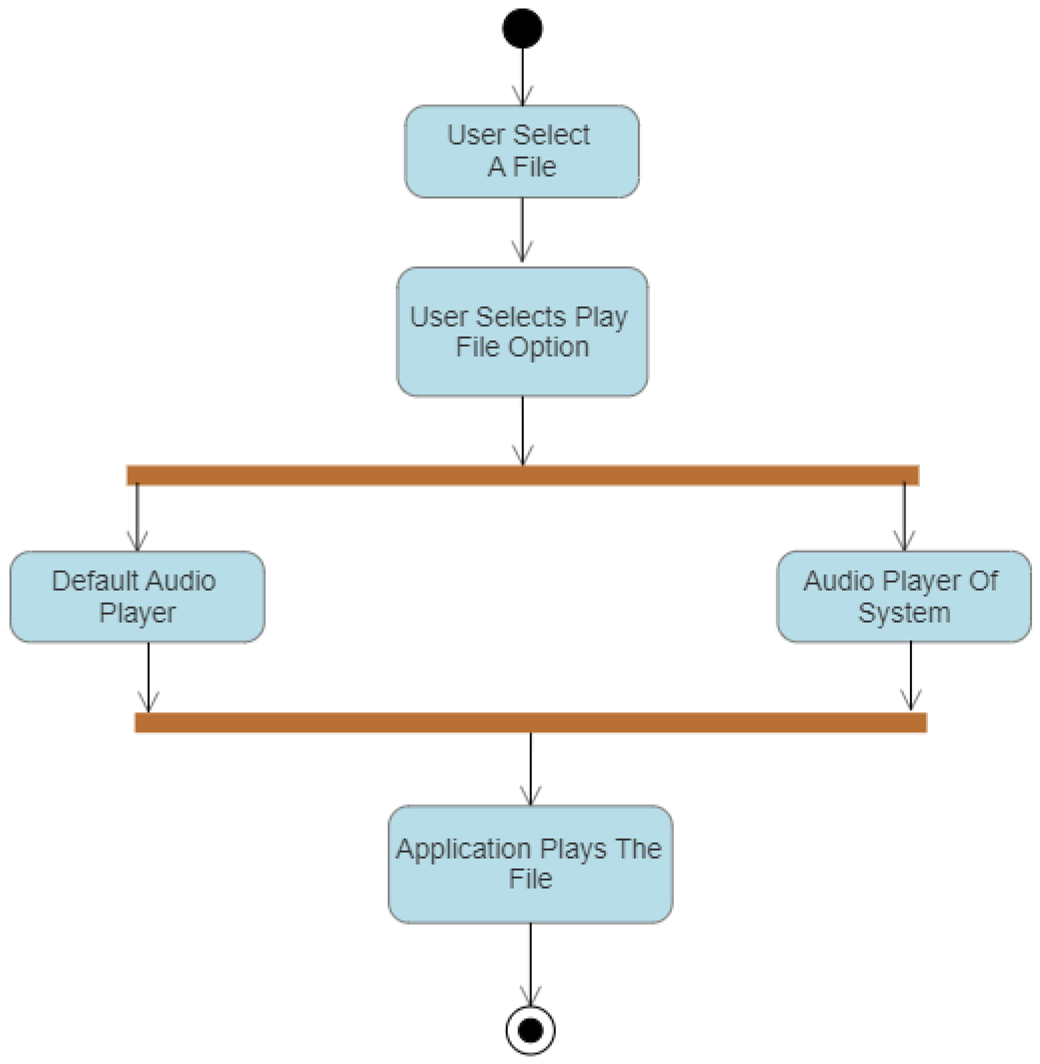


Figure 22: Play File

Activity Diagram (Run Audio Player)

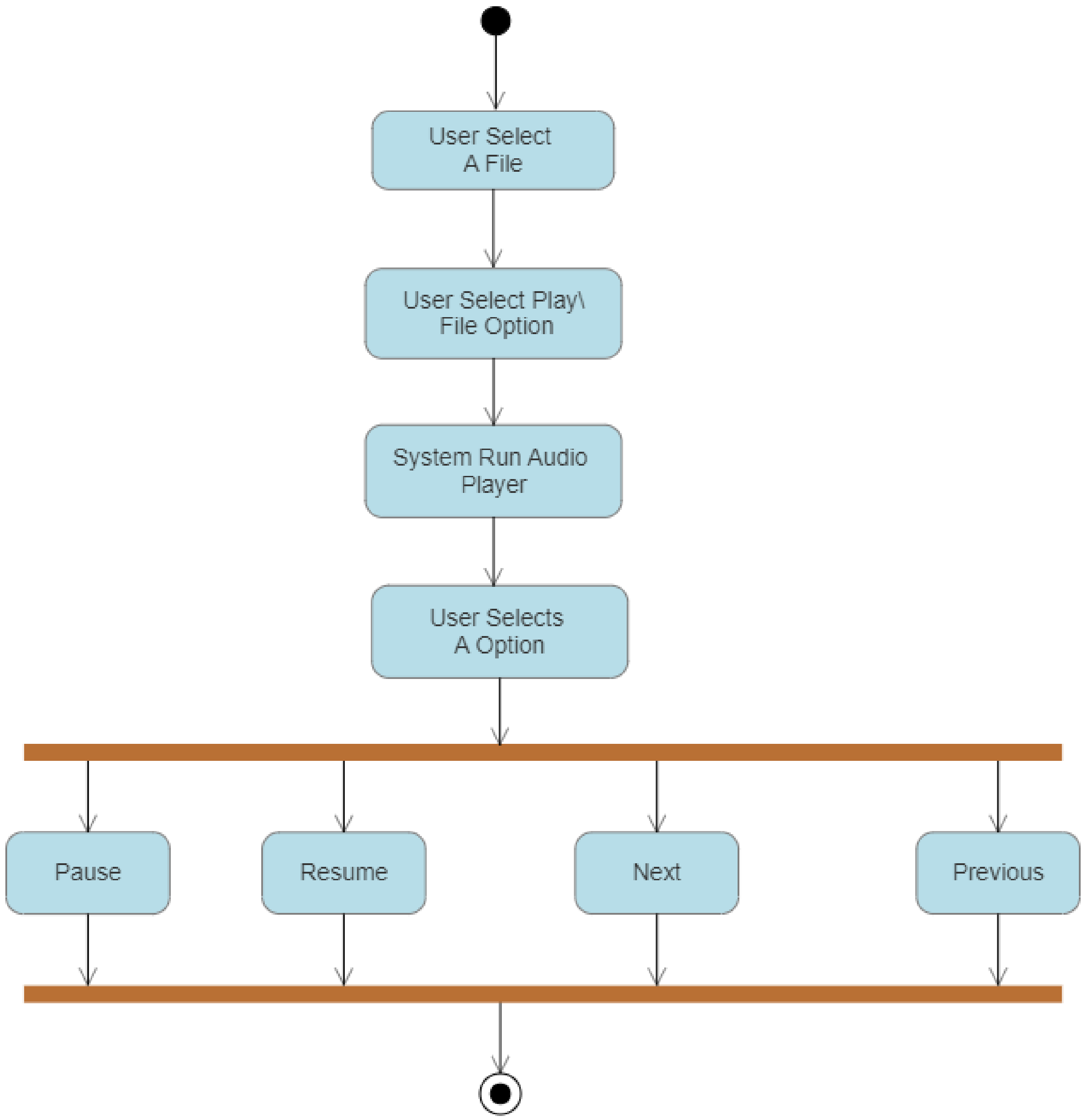


Figure 23: Run Audio Player

Activity Diagram (Log out)

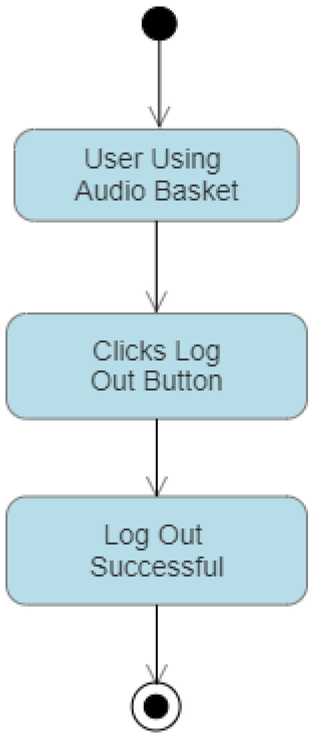


Figure 24: Log out

9. 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	**E-Farm**	**Business Area**		**Local**	
Functional Activity	**Use Case Reference**	**Design Document Reference**	**Code Module/ Reference**	**User Acceptance Validation**	**Comments**
FR1	UC1, UC2			Pass	
FR2	UC16			Verified	
FR3	UC16			Verified	
FR4	UC8			Verified	
FR5	UC5			Verified	
FR6	UC5			Verified	
FR7	UC5,UC10			Pass	
FR8	UC6			Verified	

FR9	UC14			Verified	
FR10	UC15			Verified	
FR11	UC12,UC19			Pass	
FR12	UC4			Verified	
FR13	UC21,UC15			Pass	
FR14	UC4			Verified	
FR15	UC15			Verified	
FR16	UC20			Verified	
FR17	UC24			Verified	
FR18	UC25			Verified	
FR19	UC2			Verified	

10. Appendix

10.1 Prioritization of requirements

We've prioritized the functional requirements by following Three-level Scale technique.

10.1.1 Three-level Scale

When a Business Analyst categorizes the requirements in any of the ordering or ranking scale, it is subject to the analyst's understanding of the business. Many analysts suggest that this method has some drawbacks and advocate methods that have more than one scale.

10.1.2 Prioritization of the requirements of E-Farm

FR1 – High priority

FR2 – Medium priority

FR3 – High priority

FR4 – High priority

FR5 – High priority

FR6 – High priority

FR7 – High priority

FR8 – Medium priority

FR9 – High priority

FR10 – High priority

FR11 – Medium priority

FR12 – High priority

FR13 – Low priority

FR14 – High priority

FR15 – Medium priority

FR16 – High priority

FR17 – Medium priority

FR18 – High priority

FR19 – Medium priority

FR20 – High priority

FR21 – High priority

FR22 – High priority

FR23 – Medium priority

DR1 – High priority

DR2 – Medium priority

DR3 – High priority

DR4 – High priority

DR5 – High priority

DR6 – High priority

DR7 – High priority

DR8 – High priority

DR9 – Medium priority

DR10 –Medium priority

DR11 – Medium priority

DR12 – Medium priority

DR13 – Medium priority

DR14 – Medium priority

PR1 – High priority

PR2 – High priority

PR3 – High priority

PR4 – High priority

PR5 – High priority

PR6 – Medium priority

PR7 – Medium priority

PR8 – Low priority

PR9 – Low priority