# **CHAPTER THREE**

## **Introduction**

This chapter details the system analysis and design of the project. The system to be built will be described in this chapter along with a provision of the process model that will be used in the design and implementation of the Grant Management Portal. The requirements for the software are meticulously outlined in this chapter.

It also examines the development tools which we intend to make use of in designing the system.

A use case diagram which details the various actions that a user can perform on the system will be illustrated. A sequence diagram that illustrates the sequence of events between components in the system, would also be shown below.

## **System Requirements**

### **User Requirements (Admin)**

The activities that admin can engage in are described in this section:

1. Admins shall have the ability to add, modify, and deactivate user accounts.
2. Admins shall be able to assign roles and permissions to users.
3. Admins shall be responsible for reviewing and approving or rejecting grant applications.
4. Admins shall have access to comprehensive project and user data for oversight.
5. The system shall allow admins to edit or correct data entries made by users if necessary.
6. Admins shall have access to advanced analytics and reporting tools for a holistic view of grant management.
7. The system shall send automated reminders to users and admins about upcoming deadlines.

### **User Requirements (Regular user)**

The activities that regular users can engage in are described in this section:

1. Users shall be able to log in securely using unique credentials.
2. Users shall have the ability to update and maintain their profiles, including personal and contact information.
3. Users shall be able to initiate and submit grant applications.
4. The system shall allow users to track the status of their grant applications.
5. Users shall have access to a dashboard displaying the progress and status of their ongoing projects.
6. The system shall send notifications and reminders for upcoming deadlines related to the user's projects.
7. Users shall be able to input and update relevant data related to their projects.
8. The system shall provide an interface for users to upload project-related documents.

## **Functional and Non-functional Requirements**

Functional and non-functional requirements are essential aspects of system specifications, outlining what the system shall do (functional) and how it shall perform (non-functional). In the context of a grant management system, these requirements can be defined as follows:

Functional Requirements:

Functional requirements describe the specific behaviors and functions of the system, detailing what actions the system must perform. They are as follows:

1. Users shall be able to register accounts and log in securely.
2. Different user roles (applicants, reviewers, administrators) shall have distinct permissions and access levels.
3. Applicants shall be able to submit grant applications online, providing necessary details and documents.
4. Reviewers shall have tools to evaluate applications based on predefined criteria.
5. Applicants shall be able to track the status of their submitted applications.
6. The system shall automate the approval workflow, routing applications to appropriate reviewers and administrators.
7. Automated notifications (email, in-app) shall inform users about application status changes, required actions, and deadlines.
8. The system shall generate reports and analytics about application trends, approval rates, and other relevant metrics.
9. The system shall present data through interactive charts and graphs for easy interpretation.
10. The system shall allow users to upload and store documents securely, related to applications and project progress.

Non-Functional Requirements:

Non-functional requirements define the quality attributes, constraints, and performance expectations of the system. They are as follows:

1. The system shall respond to user interactions within a defined time frame.
2. The system shall handle an increasing number of users and applications without significant performance degradation.
3. User data, documents, and communications shall be encrypted to ensure confidentiality.
4. The system shall be available 24/7 with minimal downtime for maintenance.
5. The system’s data shall be regularly backed up to prevent data loss in case of system failure.
6. The user interface shall be intuitive and user-friendly, requiring minimal training for users.
7. The system shall ensure compliance with data protection laws (e.g., GDPR) regarding user data storage and processing.
8. The system shall maintain detailed audit logs for all user activities within the system for compliance and auditing purposes.
9. The system shall have compatibility with various web browsers and operating systems used by stakeholders.
10. The system shall be able to handle concurrent user interactions and application submissions without slowdowns or errors.
11. The system shall process and analyze data efficiently, generating reports in a timely manner.
12. The system shall be regularly updated and maintained to fix bugs, add new features, and enhance security.
13. The system shall provide clear and concise error messages to assist users in troubleshooting issues.

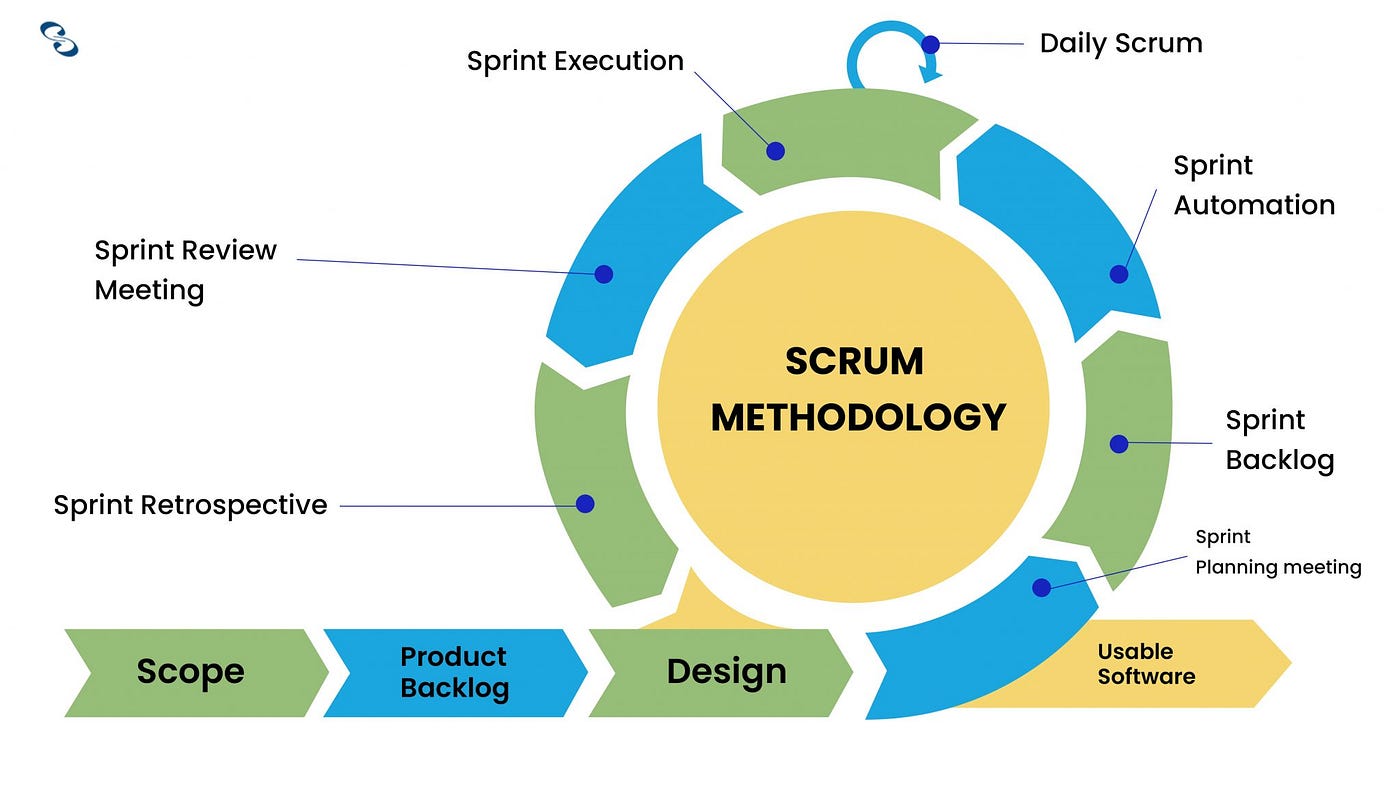
## System Architecture

The proposed Grant Management Portal is structured on a 3-tier architecture, a design framework that distinctly categorizes the application into three interconnected tiers: the presentation tier, the application tier, and the data tier. In the context of our project, the presentation tier functions as the user interface, facilitating interaction for researchers and grant administrators. This tier ensures an intuitive experience for initiating and tracking grant applications. The application tier, also known as the logic tier, houses the core functionalities of the system. It manages the processing of grant applications, and the communication and notification systems. Lastly, the data tier is responsible for storing and retrieving information related to grant applications, user details, and system configurations. This 3-tier architecture ensures a modular and scalable system, enabling independent development and modification of each tier. It enhances system reliability, security, and maintainability, critical for the dynamic nature of the system that demands continual adaptation and improvement. Overall, the 3-tier architecture provides a robust framework in alignment with the project's goals of efficiency, scalability, and user-friendly operation.

## **Software Development Process Model**

A software development process model outlines the processes involved in the design, development, and deployment of a software product. There are a variety of process models that can be adopted for any software development project. The agile process model will be used for the purpose of developing this Grant Management Portal.

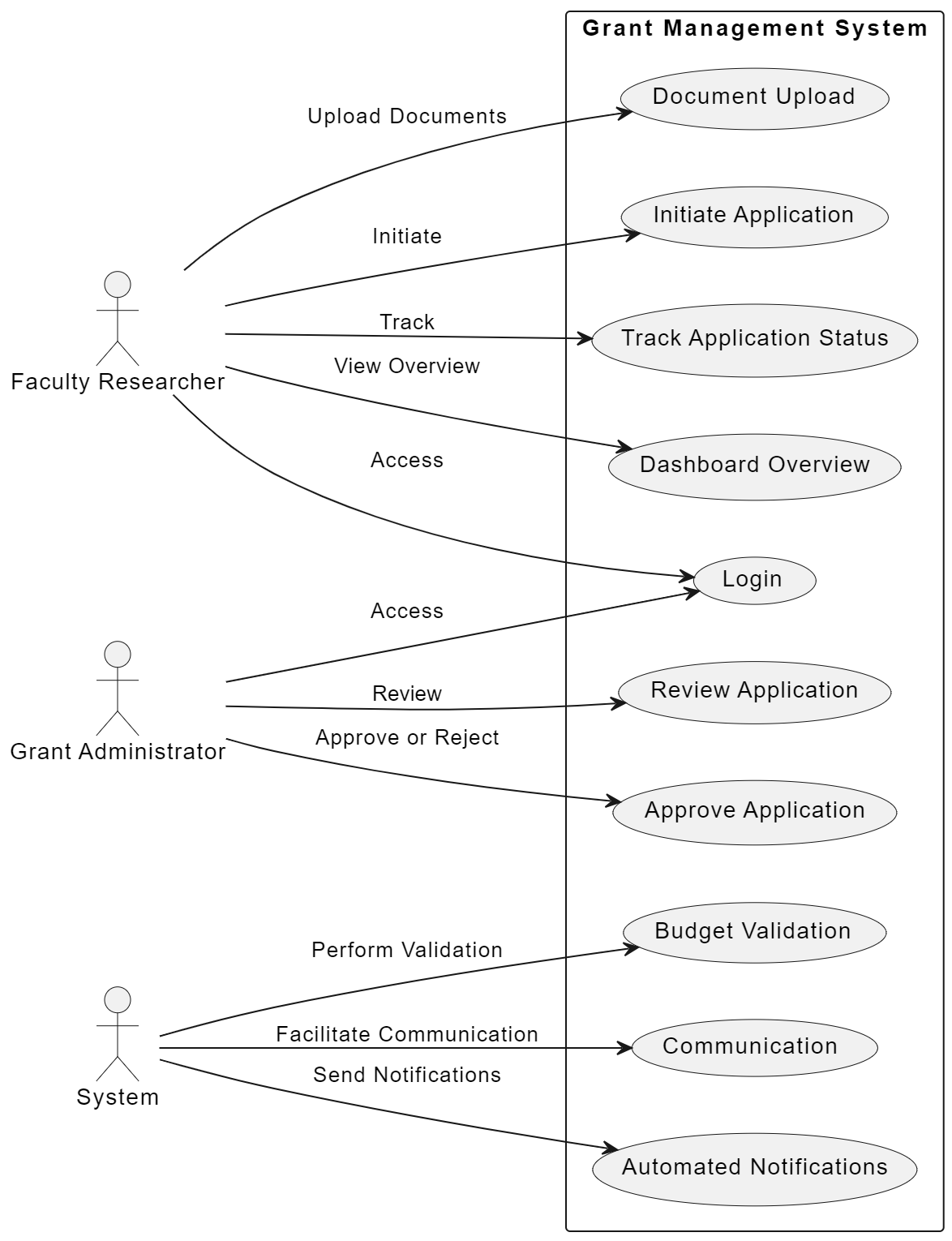
Agile process model is a software development approach based on iterative development. The phases of agile model include: Requirement gathering, requirement design, iteration, testing and feedback. The Agile method lets the team be flexible and spontaneous, so they can adjust and add new ideas as they go through the development process. Instead of sticking to a strict and rigid plan, Agile development embraces the uncertainty and unpredictability of the creative journey. The team works in short bursts, carefully shaping each feature and function. It promotes creative exploration and flexibility, permitting modifications to be seamlessly integrated during the development process. The approach also underscores the importance of regular communication and interactions among the client, developers, and other production team members, fostering a positive working relationship. Scrum is the agile framework that will be adopted in our software development process. Scrum is a lightweight yet effective approach that allows for adaptability and collaboration. It emphasizes iterative development, customer satisfaction, and the ability to respond to changing requirements. The Scrum process involves regular feedback loops, collaborative planning, and transparency, ensuring that the project aligns with client expectations and business objectives.



## **Proposed Model Diagram**

## **Use Case Diagram**

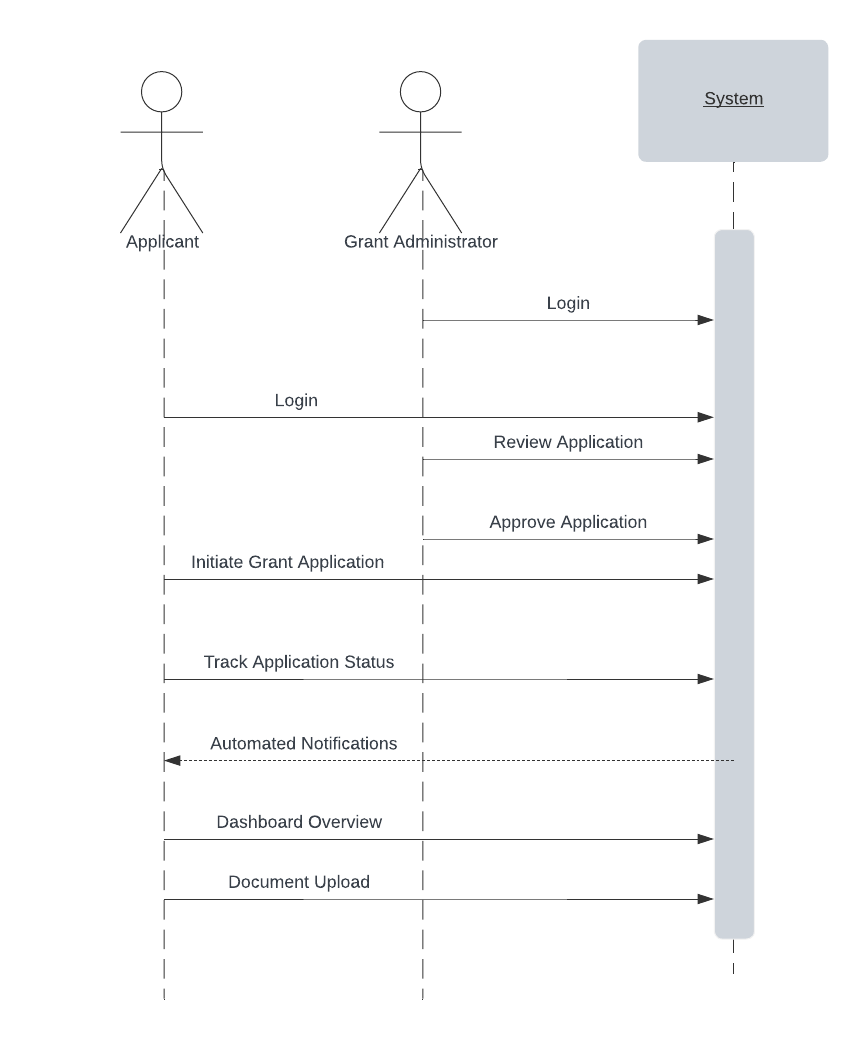
The dynamic behavior of a system is represented by a use case diagram. It incorporates use cases, actors, and their interactions to encapsulate the functionality of the system. It simulates the duties, services, and operations needed by a system or application subsystem. It shows a system's high-level functionality and describes how a user interacts with a system.

****

## **Sequence Diagram**

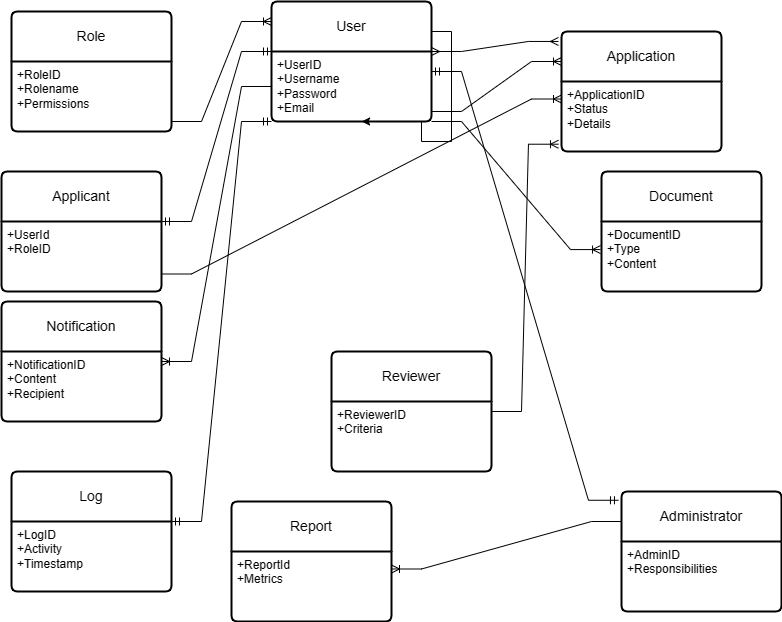
A sequence diagram is a Unified Modelling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. This diagram shows the activities in sequence that both admin and employees can carry out.

The sequence diagram for the proposed system is shown below:

****

## **Entity-Relationship Diagram**

An Entity-Relationship Diagram (ERD) is a visual representation of the entities (objects, concepts, or things) within a system, their attributes, and the relationships between them. It is used to model and design the structure of a database, illustrating how data entities relate to each other in a database schema.

****

## **Development Tools**

The development tools intended for the development and implementation of the Grant Management Portal are as follows:

1. Visual Studio and Visual Studio Code:

- Visual Studio: Microsoft's flagship IDE for Windows. It provides a comprehensive set of tools for .NET development, including ASP.NET Core applications. Visual Studio offers advanced debugging, profiling, and testing capabilities.

- Visual Studio Code: A lightweight, open-source code editor developed by Microsoft. It is also a lightweight, versatile code editor that supports a wide range of frontend technologies. It offers extensions for HTML, CSS, JavaScript, and popular frontend frameworks, providing an excellent development experience.

2. HTML and CSS:

- HTML, or Hypertext Markup Language, serves as the standard for crafting web pages. It utilizes tags to organize content, defining elements like headings, paragraphs, images, and links. Crucial for establishing the fundamental structure of web documents, HTML is foundational for web-based applications.

-CSS (Cascading Style Sheets) acts as a style sheet language complementing HTML. It dictates how HTML-marked elements shall be presented, allowing developers to separate content from presentation. Through selectors, CSS empowers developers to control layout, colors, fonts, and overall aesthetics, ensuring consistency and enhancing the visual appeal of the user interface.

3. ASP.NET Core SDK: The ASP.NET Core Software Development Kit (SDK) includes everything you need to build and run ASP.NET Core applications. It includes the .NET Core runtime, libraries, and tools. Entity Framework Core, SQL Server Management Studio (SSMS), Automapper, and SignalR are some of the popular libraries in the .NET framework.

4. Git and GitHub:

- Git: A distributed version control system widely used for collaborative development. Git allows you to track changes, merge code, and collaborate efficiently.

- GitHub: A web-based hosting service for Git repositories. GitHub provides features like issue tracking, pull requests, and collaboration tools, making it an excellent platform for team-based development.

5. Docker:

- Docker allows you to create, deploy, and run applications in containers. It simplifies the deployment process, ensures consistency across environments, and facilitates scalability. Visual Studio offers the opportunity to build .NET applications from the get-go through the use of its in-built Docker support.

6. Postman:

- A popular API development and testing tool that enables developers to test APIs quickly and document them effectively. It simplifies the process of building and testing APIs during development.

7. Swagger/OpenAPI:

- Swagger/OpenAPI tools enable you to document your API in a standardized way. They generate interactive API documentation, making it easier for developers to understand and consume your API endpoints.

8. React:

- React: A JavaScript library for building user interfaces, developed and maintained by Facebook. It allows developers to create reusable UI components and provides a virtual DOM for optimal performance.

9. Node Package Manager (NPM):

- npm (Node Package Manager): It allows developers to install, manage, and share JavaScript packages and libraries, making it essential for frontend development.

10. Bootstrap and Tailwind CSS:

- Bootstrap: A widely used front-end framework that simplifies the development of responsive and mobile-first websites. Bootstrap offers a collection of CSS classes and JavaScript plugins for building modern user interfaces.

- Tailwind CSS: A utility-first CSS framework that provides low-level utility classes to build custom designs. Tailwind CSS allows developers to create unique designs without leaving the HTML file.