RESEARCH GRANT FINDER

ISLAM MOHAMMED RUZHAN

UNIVERSITI TEKNOLOGI MALAYSIA

PSZ 19:16 (Pind. 1/13)

UNIVERSITI TEKNOLOGI MALAYSIA

DECLARATION OF THESIS / UNDERGRADUATE PROJECT REPORT AND COPYRIGHT

Author's full name : Islam Mohammed Ruzhan

Date of Birth : 05/06/2023

Title : RESEARCH GRANT FINDER

Academic Session : 2022/2023-2

I declare that this thesis is classified as:

☐	**CONFIDENTIAL**	(Contains confidential information under the Official Secret Act 1972)*
☐	**RESTRICTED**	(Contains restricted information as specified by the organization where research was done)*
✓	**OPEN ACCESS**	I agree that my thesis to be published as online open access (full text)

1. I acknowledged that Universiti Teknologi Malaysia reserves the right as follows:
2. The thesis is the property of Universiti Teknologi Malaysia
3. The Library of Universiti Teknologi Malaysia has the right to make copies for the purpose of research only.
4. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

SIGNATURE OF STUDENT

A20EC4028

MATRIX NUMBER

SIGNATURE OF SUPERVISOR

DR. MOHD SHAHIZAN BIN OTHMAN

NAME OF SUPERVISOR

NOTES : If the thesis is CONFIDENTIAL or RESTRICTED, please attach with the letter from the organization with period and reasons for confidentiality or restriction

“I hereby declare that we have read this thesis and in my
opinion this thesis is suffcient in term of scope and quality for the
award of the degree of Bachelor of Computer Science (Software Engineering)”

Signature	:	_____
Name of Supervisor	:	PROF. MADYA. TS. DR. MOHD SHAHIZAN BIN OTHMAN
Date	:	4 JULY 2024

Universiti Teknologi Malaysia

DECLARATION

I declare that this thesis entitled *" RESEARCH GRANT FINDER"* is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : .....  .....

Name : ISLAM MOHAMMED RUZHAN

Date : 4 JULY 2024

DEDICATION

This thesis is dedicated to my father, who taught me that the best kind of knowledge to have is that which is learned for its own sake. It is also dedicated to my mother, who taught me that even the largest task can be accomplished if it is done one step at a time.

ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians, and practitioners. They have contributed towards my understanding and thoughts. In particular, I wish to express my sincere appreciation to my main thesis supervisor, Professor Madya Dr. Mohd Shahizan bin Othman, for encouragement, guidance, critics and friendship. I am also very thankful to Dr Siti Zaiton for her guidance, advices and motivation. Without their continued support and interest, this thesis would not have been the same as presented here.

My sincere appreciation also extends to all my friends and others who have provided assistance at various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space. I am grateful to all my family member.

ABSTRACT

The proposed system that is going to be developed is the Research Grant Finder System. This system is web-based, and it uses web scraping technology to extract the data of the grant websites and present them in an attractive way by the help of data visualization. This system will reduce the hard efforts made by the researchers to look for suitable grants and apply for them. For developing the system, the requirements are first taken into consideration. After many meetings and interviews, the final requirements were set by the stakeholder and then the planning and documentation of the system is done. The total system was divided into six modules according to their functionalities. The proper methodology was chosen for this project which is Agile Methodology. The SRS contains a brief description of the functional and non-functional requirements of the system. The SDD consists of the architecture of the system which is Model-View-Controller. The STD consists of the test cases of each module of the system.

ABSTRAK

Sistem cadangan yang akan dibangunkan ialah Sistem Pencari Geran Penyelidikan . Sistem ini berasaskan web, dan ia menggunakan teknologi pengikisan web untuk mengekstrak data tapak web geran dan mempersembahkannya dengan cara yang menarik melalui bantuan visualisasi data. Sistem ini akan mengurangkan usaha keras yang dilakukan oleh penyelidik untuk mencari geran yang sesuai dan memohonnya. Untuk membangunkan sistem, keperluan terlebih dahulu diambil kira. Selepas banyak mesyuarat dan temu bual, keperluan akhir telah ditetapkan oleh pihak berkepentingan dan seterusnya perancangan dan dokumentasi sistem dilakukan. Jumlah sistem dibahagikan kepada enam modul mengikut fungsinya. Metodologi yang sesuai telah dipilih untuk projek ini iaitu Metodologi Agile. SRS mengandungi penerangan ringkas tentang keperluan fungsian dan bukan fungsian sistem. SDD terdiri daripada seni bina sistem iaitu Model-View-Controller. STD terdiri daripada kes ujian setiap modul sistem.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN	-	Artificial Neural Network
GA	-	Genetic Algorithm
PSO	-	Particle Swarm Optimization
MTS	-	Mahalanobis Taguchi System
MD	-	Mahalanobis Distance
TM	-	Taguchi Method
UTM	-	Universiti Teknologi Malaysia
XML	-	Extensible Markup Language
ANN	-	Artificial Neural Network
GA	-	Genetic Algorithm
PSO	-	Particle Swarm Optimization

LIST OF SYMBOLS

δ	-	Minimal error
D, d	-	Diameter
F	-	Force
v	-	Velocity
p	-	Pressure
I	-	Moment of Inertia
r	-	Radius
Re	-	Reynold Number

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Research refers to collection of data and then analyze it and come to a conclusion about any existing work or to propose any new idea. It is mainly done by university lecturers and professors and students under them. But research work is expensive. It requires a lot of money to publish research in the journals, to collect data and analyze data using various technologies and methods. The researchers also need to provide salary to the research assistants during the period of research. It is not possible for the researcher to bear such expenses. So, they have to take funding from various organizations. Furthermore, research funding may lead to continuous industry-science relations by making researchers more willing to collaborate and hence increase transfer of technological knowledge from science to industry which fosters and accelerates industrial innovations. (Bogler, 1994) There are many organizations that provide fundings for research for their own benefits. Researchers need to go through respective organizations' website to look for research grants. They have to check whether the grants are relevant to their research.

As a research-oriented university, the lecturers and professors of UTM also has to do research and look for funding. It is quite a hectic process, and it is not possible to look for each and every website for funding. So, a website that can gather all the grants provided by the funding organizations in a single system will be very beneficial for the lecturers and professors of UTM. For this reason, under the supervision of my supervisor, I plan to develop a system that will collect grant related information from the websites and store it in a single system using the technique of web scrapping. Through this, using a single system, the lecturers can find grants from different funding organisations. They can also filter and narrow their search and easily find the grants suitable for them. Web scrapping is the method of extracting data from a website. There is a JavaScript library that can be used for web scraping. It is called Puppeteer.

In this way, a system can be developed that will enable the researchers to find fundings easily.

1.2 Problem Background

Research grant is very important for a lecturer or a professor to start working on their research. Without research it is not possible to carry out research works properly. But it is very hard to find grants manually from various websites. It can be very time consuming and challenging because they have to search for grants that is suitable for their research. The lecturers of Faculty of Computing do research in various fields. For that, they need to search for grants from multiple websites. This causes them to lose a lot of their time. They cannot also go through all the websites. So they miss on important funding opportunities. They cannot filter their interests, grant types, and find grants according to grant deadlines. There is an existing Research Management website for the researchers of UTM, but it is completely manual, in that website researchers cannot search for grants according to their research topic, deadline, grant amount etc. There they have to choose from the available grants that are existing in the system. In this system, researchers cannot have access to the most recent information on funds that are available because of the lack of an automated mechanism to search for grants, which may result in loosing funding opportunities.The researchers are not notified of any upcoming grants, so they have to visit the website every now and then to look for new grants. And the researchers cannot customize the grants according to their needs. Sometimes they have no option but to choose a grant that does not satisfy their requirements properly. So, it causes hindrance in the regular process of research and development of the university.

1.3 Project Aim

The aim of this project is to develop a system that helps the researchers of faculty of computing to search for grants according to their specific requirements.

1.4 Project Objectives

The objectives of the project are:

a) To study the requirements of the Research Grant Finder system in terms of its usability and functionality.

b) To propose the design of the system by selecting the appropriate design pattern, to design the database structure and the user interface.

c) To develop a system that meets all the requirements of the different category of users and that follows the selected design pattern and models.

d) To test the system using software testing tools to check if the system is able to carry out the functionalities and meet the requirements of the stakeholder.

1.5 Project Scope

The scopes of the project are:

- The system will be a web-based system.
- It will have login authentication.
- The grant information will only be provided to authenticated users of the Faculty of Computing only.

- The system will use web scraping based on JavaScript libraries.

- An individual can only see his grant statistics related.

1.6 Project Importance

The system will assist the researchers in getting grants according to their requirements. Thus, the research process will be made more convenient. It will save them an ample amount of time. The researchers will get the most recent information about new upcoming grants through the system. The researchers will be able to filter the grant searching according to the type of grant he/she wants. A centralized system will be available to them, using which they can find the grants provided by various funding organizations. It will also be beneficial to the funding organizations, as their grant will be made more reachable to the various researchers. Overall, this system will provide better research opportunities and will benefit researchers, research organizations, and funding organizations to a great extent.

1.7 Report Organization

This chapter consists of the summary of the project, the problem statement, objectives, and scope of the project. Chapter 2 includes the literature review of the system. Chapter 3 consists of the methodology of the software development and the required hardware and software for this project. Chapter 4 consists of the requirement analysis of the system. And the coding and testing are discussed in chapter 5.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Research is a very time-consuming as well as an important task to do. Almost all the universities in this world have researchers who research in various topics of science, commerce and education. Many new ideas are introduced through this research which make an impact in various fields. But for research one of the most important things is fundings. Research cannot have an impact if there is no funding. A research process has a lot of things to fund for example, staffs, equipment, testing, data collection and many more. These are borne by some funding organizations, who need new ideas to work on and need scientists to develop new things. So, the research funding organizations fund various research, which they find promising. It is quite hectic for the researchers to search for funding according to the topic of their research. They have to manually look through hundreds of websites to find the most appropriate funding which is very time consuming, and it may cause a negative impact on the research quality.

To reduce the hassle of searching for research in a complex process, many websites are developed which contain the funding information of various funding organizations. In these websites various research grants are available in a single platform. These websites collect research grant data from various websites and take them as inputs and store them in the repository. The systems get the information form this repository when requested by the user. The user can search for grants by filtering according to his research. The grant data will be shown in a dashboard including the grant statistics and most popular grant types.

To develop a research grant finder for the Faculty of Computing, UTM, it is necessary to analyse and understand similar existing systems. By analysing the

existing systems, we can get an idea of the systems and we can know the limitations of the systems and solve those limitations in the proposed system. Finally, we can develop the system using Puppeteer of JavaScript for web scrapping and a database language to store the data in the system repository.

2.2 Current System Analysis

Currently there is no automated system for the researchers to search for research grants. The researchers who want grants have to go through the RMC website of UTM and look for grants. But RMC is a manual process, where grants are added manually, and there is less chance of getting the latest grants and the number of grants found is also less. Other ways the researchers use to look for grants is the websites of different funding organizations. But this is also a very hectic process. There are hundreds of grant funding organizations, with each different types of grants. So, it is not possible for the researchers to go through all the grants. It consumes a lot of their time. And it also reduces their chance to complete the research within the due time.

2.3 Comparison between existing systems

To develop a perfect system with the possibility of least errors, we have to analyze the existing systems that also use the same techniques and are used for the same application. In this way we can identify the shortcomings of the existing systems and try to solve them in the system we plan to develop. There are many websites that contains information about different types of grants. They use different techniques to store the grant data. Some use web scrapping, others store the data in the server using database. Some similar systems that enable researchers to search for grants are discussed below:

2.3.1 GrantForward

GrantForward is a grant searching tool that collects and arranges data on available grants from a variety of sources, including governmental bodies, foundations, and other organizations. GrantForward is a useful tool for academics,

scientists, and scholars looking for financing for their projects and studies since it gathers data on hundreds of grants and funding possibilities using web scraping technologies.

Features:

- Search engine
- Advanced searching option
- Ability to filter search based on different grant category.
- Allows users to track the grant of their interest and receive updates.

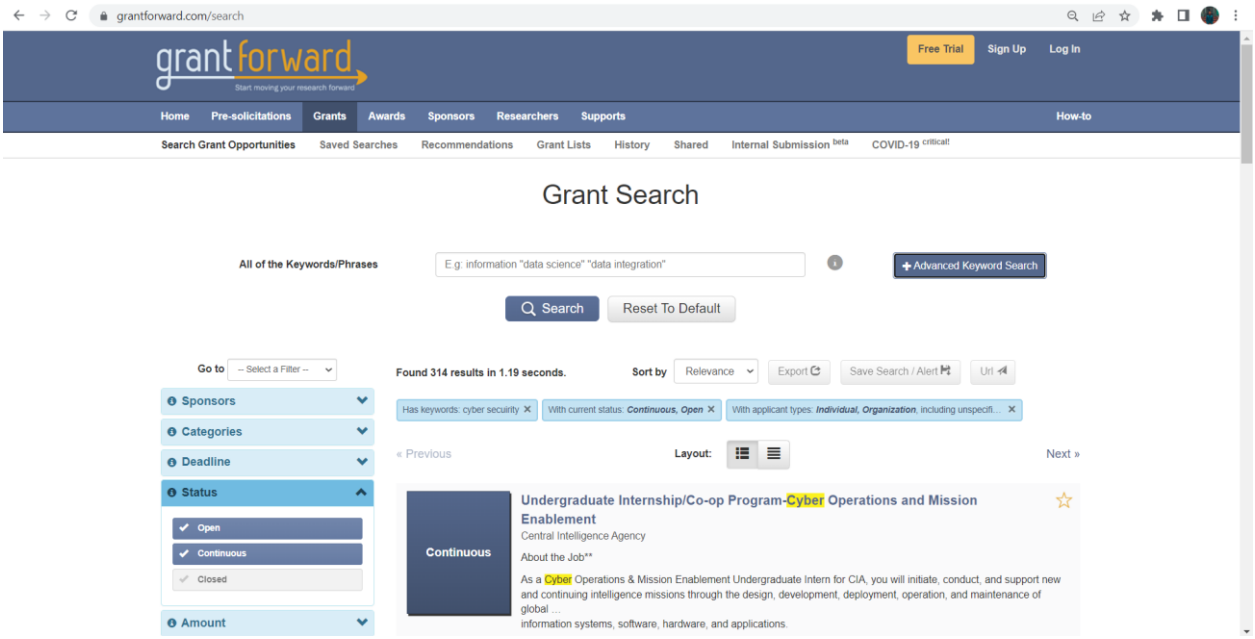


Figure 2.1: Interface of GrantForward

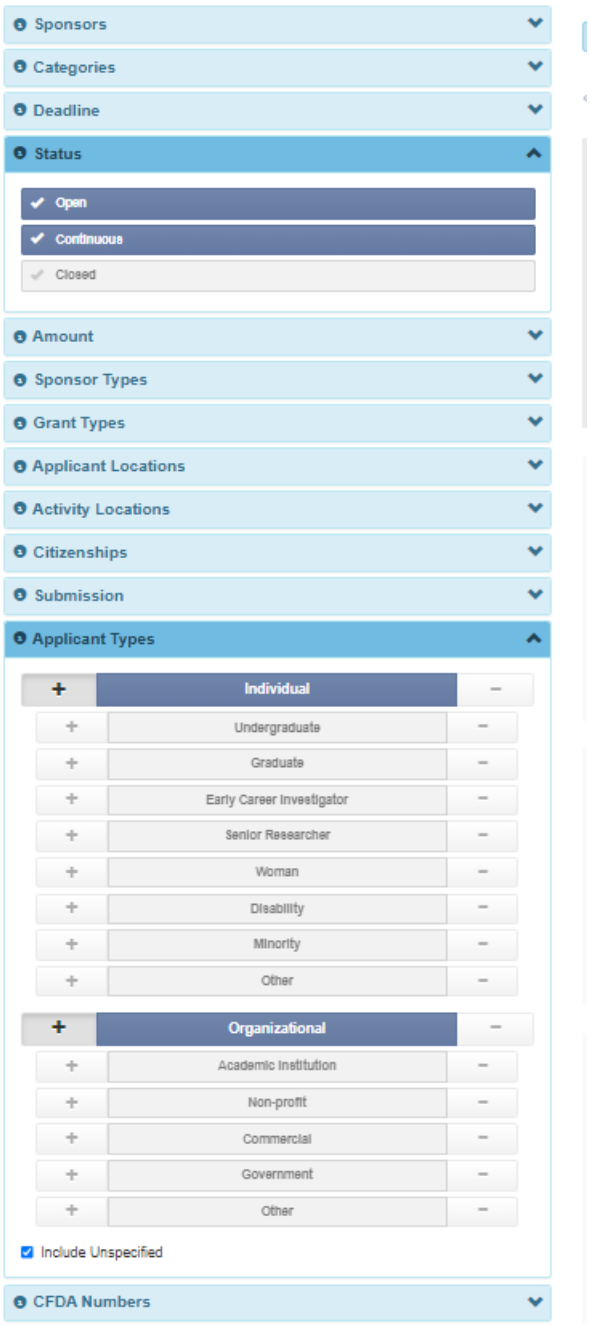


Figure 2.2: Filters of GrantForward

While GrantForward is a very useful tool for researchers and academics, it has a complex user interface, and the number of international grants is limited in this system. Also, it is a paid service so, it will cost some extra money for searching suitable grants. But if the School of Computing has their own research grant finder, the researchers of UTM need not to pay any extra money.

2.3.2 NIH Grants and Funding

The National Institutes of Health (NIH) grants website is a comprehensive resource for researchers, institutions, and organizations seeking funding for biomedical and health-related research projects. It provides detailed information on various grant programs, application processes, and funding opportunities offered by the NIH. The platform collects data from several sources, including governmental organizations, foundations, businesses, and organizations, using web scraping techniques and then presents it in a user-friendly manner. For a number of sectors including education, healthcare, the environment, the arts, and community development, the website provides a wide range of options for funding, such as grants, fellowships, scholarships, and prizes.

Features:

- NIH has a vast database.
- Database updated regularly with new grants.
- It has filter option in searching, through which the users can easily narrow down their search according to their research area.
- This system also gives email alerts to the users regarding important grants. The grants are categorized according to the fields.
- Easily navigable and easy to access information.

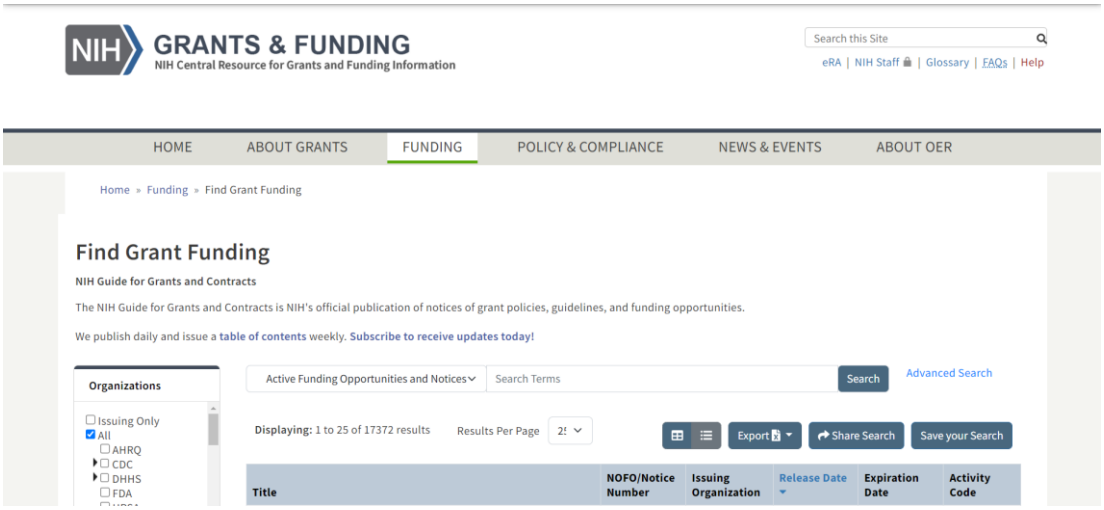


Figure 2.3: Interface of NIH Grants

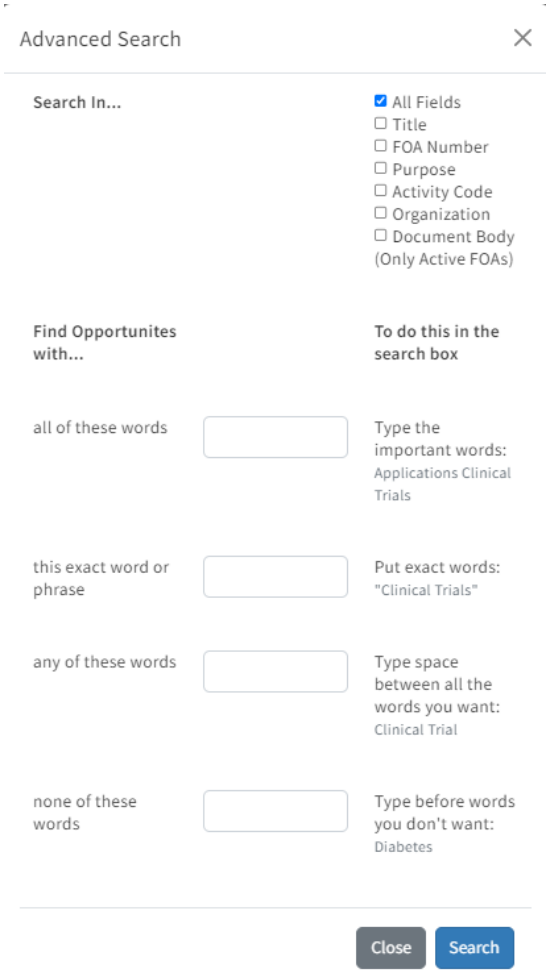


Figure 2.4: Advanced search options in NIH Grants

But the NIH website does not provide any analysis of the grants that it displays. It only shows a list of the grants and the user has to go thoroughly through the website to find for their desired grants.

2.3.3 Grants.gov

It is a website to search for and apply for the grants provided by the US government. It provides various grants inside and outside the United States. No subscription is required to look for grants in this system.

Features:

- Users of the website can do grant searching using keywords, sources of funding, grant categories, and application deadlines.

- Also, users can set up a personalized account to store searches, monitor applications, and get alerts about new grant possibilities.
- Provides guidelines on how to apply for grants.
- It is a grant provider of the US government.
- It provides grant for federal projects.

Though grants.gov is one of the biggest and oldest grant providing websites, it has some problems, which include unavailability of grants other than from USA, there is a filter search option but it is too complex and there has been reports that the filters are not much effective.

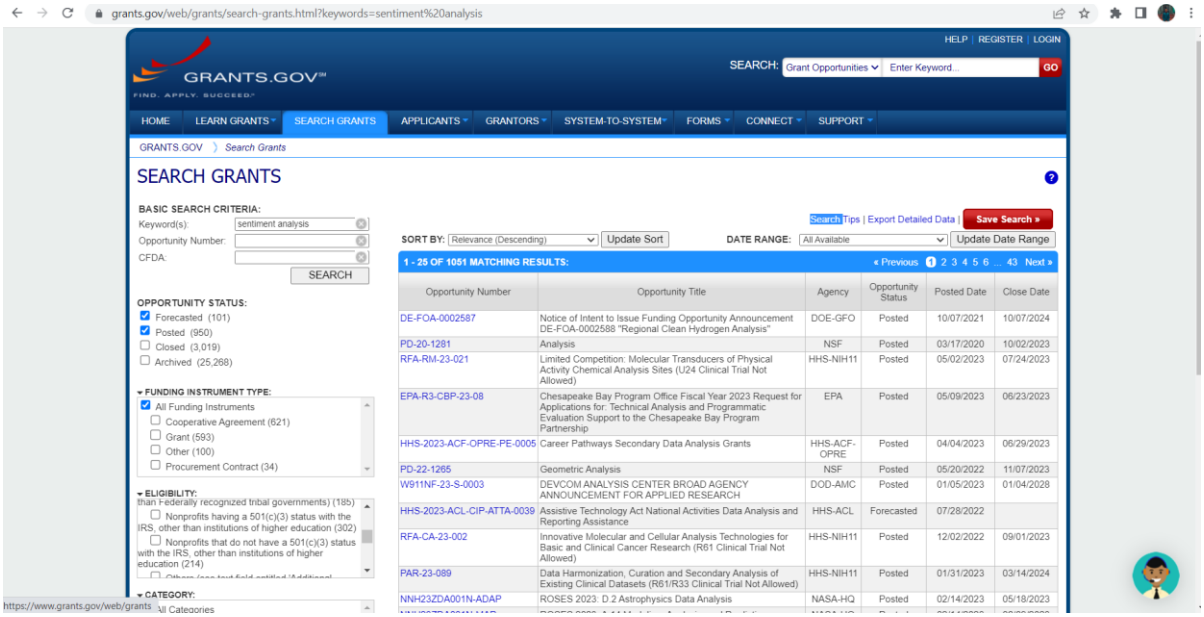


Figure 2.5: Search results in Grants.gov

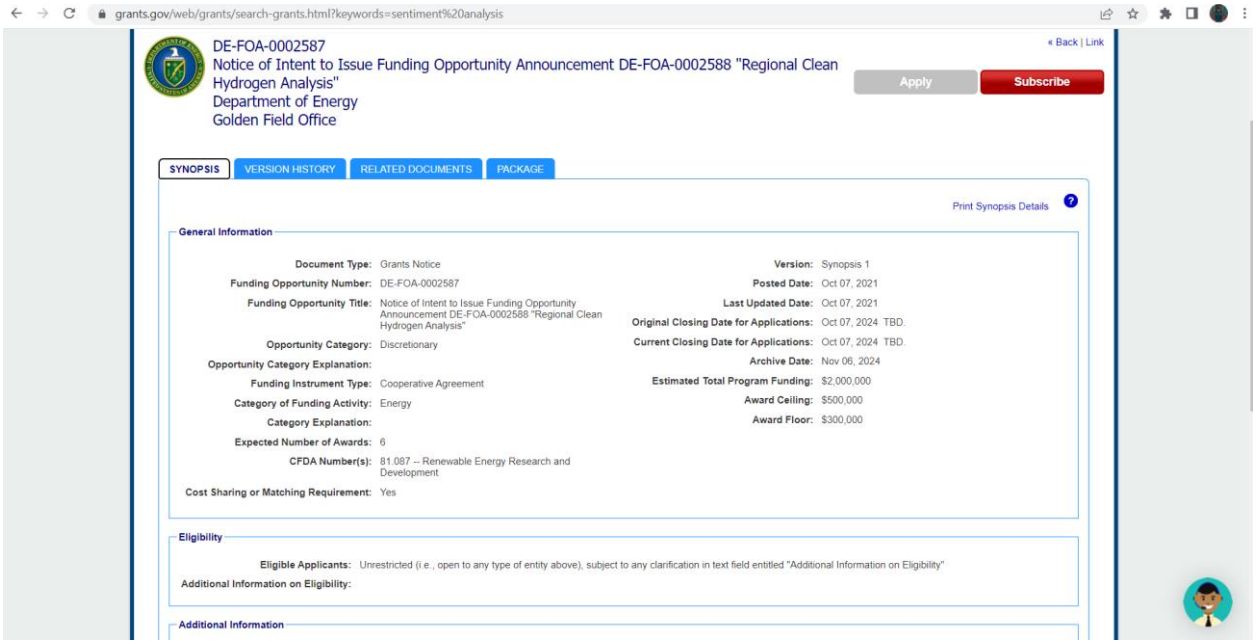


Figure 2.6: Grant details in Grants.gov

Table 2.1: Comparison Table between existing systems and Research Grant Finder

	GrantForward	**NIH Grants**	**Grants.gov**	**Research Grant Finder**
System Type	Web	Web	Web	Web
Web Scrapping	Full	Partial	Partial	Full
Repository	Yes			Yes
Dashboard	No	No	No	Yes
Data Analytics	Not Used	Not Used	Not Used	Used
Information	Less information, well organized	Organized information	Huge information, not organized	Well-structured and organized information
Subscription	Yes	Yes	No	No
Filter search	Yes	Yes	Yes	Yes
User Interface	Simple	Easy to navigate	Complex and hard to navigate	Simple and easy to navigate
Chatbot	No	No	Yes	No
Language	English	English	English	English
User friendly	Yes	Yes	No	Yes

All the systems use filtering methods to search for Grants. Not all websites use web scrapping to extract grant data. All of the existing systems and the to-be developed system are web-based systems. None of the existing systems contain dashboards for data analysis of the grants. The Research Grant Finder will analyze the grants data obtained from various sources.

2.4 Literature Review of Technology Used

Burgelman et al. (1996) refer technology as the theoretical and practical knowledge, skills, and artifacts that can be used to develop products and services as well as their production and delivery systems. To develop a system properly we need to use different kinds of technologies. In this age of advancement in science, technology is changing at a very fast pace. So, we need to use such technologies which is not easily replaceable, and which can adapt to changes. We will use the following technologies for the Research Grant Finder:

Frontend:

HTML:

Html bears the skeleton of a website. Since Research Grant Finder will be a web-based project, it is a must to use HTML.

CSS:

CSS will be used to beautify the front end and the dashboard of the system and to make some transitions and effects.

JavaScript:

JavaScript adds dynamic behaviour to the webpage. It is used for validation purposes and sending prompt messages to the user.

Backend:

Node Js:

Node Js is a widely regarded JavaScript runtime environment that allows to execute JavaScript code on the server, outside of a web browser.

Express Js:

Express.js is a simple and well-structured Node.js web application framework that provides a robust set of features for building web and mobile applications.

Web scrapping:

Puppeteer:

It is used to extract data from websites. It is a library based on the JavaScript programming language.

Database:

MongoDB

MongoDB is a database management system that can be used to modify, delete, create and store data. It is a must for every dynamic website.

Visual Studio Code:

Visual studio code is the editor for writing source codes. Codes in different languages are written and run using an integrated development in Visual Studio Code.

2.5 Chapter Summary

Chapter 2 provided detailed study of the research grant finder systems that exist around the globe. The advantages and shortcomings of the existing systems have been discussed and a comparison has been done with the proposed Research Grant Finder system so that, a proper and impactful system can be developed for the Faculty of Computing (FC).

CHAPTER 3

SYSTEM DEVELOPMENT METHODOLOGY

3.1 Introduction

Software development is a complex and long process. It is a challenging and complicated effort that requires careful planning in order to avoid financial losses and provide high-quality results. A small defect during software development can cause a vast amount of loss. If no planning is done before starting the development of a software, then many errors may arise during the development phase, it will result in the development of faulty software and also it will require a lot of time for its development. For this reason, planning and management are important steps of software development.

Specific methodologies are followed by developers to develop a software according to its type and requirements. The term "software development methodology" describes a planned, conducted out, and managed approach to software development initiatives. It provides a framework for organizing tasks, defining roles and responsibilities, and establishing a sequence of activities to be taken along the development process.

There are many methodologies that have been followed by developers to develop software. Such as Waterfall method, Incremental method, Agile development, Prototyping, Rational Unified Process etc. Each of these methodologies have their own characteristics and used according to the software being developed.

Software development is a complex and long process. It is a challenging and complicated effort that requires careful planning in order to avoid financial losses and provide high-quality results. A small defect during software development can cause a vast amount of loss. If no planning is done before starting the development of a

software, then many errors may arise during the development phase, it will result in the development of faulty software and also it will require a lot of time for its development. For this reason, planning and management are important steps of software development.

Specific methodologies are followed by developers to develop a software according to its type and requirements. The term "software development methodology" describes a planned, conducted out, and managed approach to software development initiatives. It provides a framework for organizing tasks, defining roles and responsibilities, and establishing a sequence of activities to be taken along the development process.

There are many methodologies that have been followed by developers to develop software. Such as Waterfall method, Incremental method, Agile development, Prototyping, Rational Unified Process etc. Each of these methodologies have their own characteristics and used according to the software being developed.

3.2 Methodology Choice and Justification

The methodology that will be most suitable for this system is the Agile Software Development methodology. Agile is a wide umbrella of software development beliefs. It is a conceptual frame-work for software engineering that begins with a starting planning phase and follows the road toward the deployment phase with iterative and incremental interactions throughout the life-cycle of the project. (Al-Saqqa et al., 2020) The traditional Waterfall method is not used because it is not suitable for developing automated systems using web scraping. Web scraping projects often require flexibility and adaptability due to the dynamic nature of websites and changes in grant criteria. The waterfall methodology is linear in nature which makes it challenging to respond to these changes effectively, as it can result in a significant amount of rework if requirements or design need to be modified. On the other hand, in Agile method, the changes can be made easily in the next step as it is an incremental process.

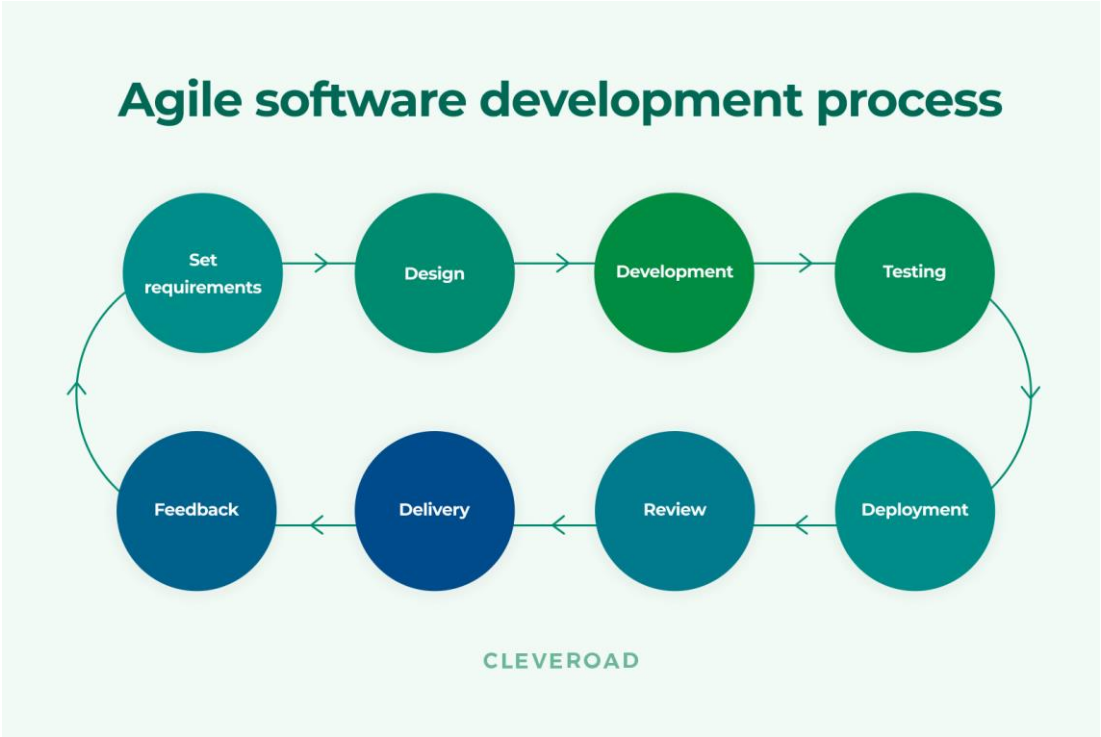


Figure 3.1: Agile methodology

The main advantage of agile method is its iterative approach. In this method, the project is divided into smaller increments or sprints. Each iteration of the grant-finding algorithm can focus on collecting information from a certain website or applying a particular filter. This makes it possible to release functionality early and continuously, which makes it simpler to validate and get feedback from users or stakeholders. By developing software using agile method it is possible to provide the software early than other methods. Because in this method, a small part of a code can be constructed first and it can be used and also the part of code can be enhanced later to make larger systems. In short, through developing the system with Agile Methodology, a working version of the system can be gotten early and also improvements can be done on it due to its incremental process of development.

3.3 Phases of the Chosen Methodology

Agile methodology focuses on delivering working software in small increments and emphasizes frequent communication with customers and stakeholders. Developers can respond to changing needs of stakeholder and provide value early in the development process due to the methodology's emphasis on flexibility, ongoing

planning, and openness. There are six phases of Agile software development which are concept inception, iteration, release, maintenance, and review. These are discussed below:

3.3.1 Concept Phase

The first phase of agile development is Concept phase. In this phase, the basis of the software development is completed. A layout of the system is created in this phase. The scopes and objectives are identified, the primary requirements are discussed with the stakeholder. The developer provides solutions to the problems of the stakeholders and an estimation of costs and probable date of completion through brainstorming, initial discussions, and gathering information.

For the Research Grant Finder system, it is necessary to define the objectives and vision of the system. The necessary documentation needs to be prepared and the features of the grant searching has to be identified with the stakeholder by discussions and brainstorming. An estimation of project completion date and coat will be provided to the stakeholders.

3.3.2 Inception Phase

This phase can start after the concept phase is completed. This phase marks the end of the planning stage of the project. The requirements, functionalities and objectives are researched and discussed in depth. The tasks are divided among the developers in a team and the tasks are prioritized. The necessary tools for the development are provided to the developers. It is important to build the project structure in this phase. The mockup user interface and the system development are designed in this phase.

For the system Grant Finder, this phase will be the last stage of planning. The requirements will be revised with the stakeholder. The scope and the objectives may be revised. The different tasks for the development such as grant searching, filtering, grant dashboard etc. will be identified and will be prioritized in an order. The mockup user interface will be designed according to the proposed system and the development phase will begin.

3.3.3 Iteration Phase

This is the longest and most important phase of agile development. This is the phase where the development of the system starts. The tasks are divided into sprints based on their priority. Each sprint has specific tasks and deadlines. The development progresses with the progress of the sprints. In this phase the coding starts, and the design is transformed to code. A sprint can consist of planning, execution, and delivery of the completed tasks. Throughout the iteration, the team holds daily stand-up meetings to provide an update on progress, discusses any challenges or impediments they are facing, and highlights their planned work for the day.

For Research Grant Finder system, all the tasks will be identified and divided into sprints. The sprints will contain planning and execution of tasks, such as designing the front end, building the structure of the system, including search function, filtering grants. Each sprint will contain extraction of grant information from specific websites. In this way through the progress of multiple sprints, the development of the system will be completed.

3.3.4 Release Phase

The release phase starts when the development of the system is almost done. That is the coding part is almost done. After the development of the system is done it

will be tested by the testers to ensure that it can perform its functionalities properly. If the testing goes well, then the product is set for release.

When all the coding of the Research Grant Finder will be completed, and the complete structure of the system is visible, the testing will be done to ensure that the system is working properly. A release will include scraping capabilities that will be done from multiple websites, filtering options, and a user interface. The release would undergo testing and quality assurance before being deployed or distributed to users.

3.3.5 Maintenance Phase

The maintenance phase begins when the software is released. Now that the system has been built completely, users may access it. The software development team will continue to offer help during this stage to keep the system operating efficiently and fix any new issues. They will also be available to provide consumers with further training and to make sure they are familiar with how to utilize the product. New iterations can be made over time to improve and add new features to the current version.

For the Research Grant Finder software, after its release, maintenance of the system will be a vital task for the operation of the software. The system has to work with a lot of data and for every search for grants by the user it will use web scraping to extract information. Besides, there may be many new websites that provide grants. So those websites need to be added to the system. To maintain smooth searching of grants including filtering and up-to-date grants, it is necessary to maintain the system to cope with the new changes.

3.3.6 Review Phase

The development team presents the functional features or software that have been developed throughout the iteration during the review phase. This demonstration allows stakeholders to assess the progress made and provide their input. The

developers present the working software or completed features to stakeholders, showcasing the functions that have been implemented in the system. Stakeholders have the opportunity to provide feedback during the review phase. They can express their ideas, concerns, and requests for any necessary modifications or improvements.

After release, the Research Grant Finder app needs to be reviewed by the stakeholders. They need to check whether the system is working properly or not. They need to check if the website could extract grant information from the sources and do the filtering according to the user criteria. After certain functionalities that have been developed at each sprint, the stakeholders do a review of the system, to ensure that the functionalities have met their requirements.

3.4 Sprints and their description

3.4.1 Sprint 1

The first sprint involved in implementing the system structure, inspecting the target website to be scrapped, installing necessary libraries and dependencies, implementing the scrapping module for extracting data and setting up the controllers for the scrapping. I also created some of the react components for the system and implemented user creation and login and run and test if the functions developed are working properly by presenting the completed tasks to get a review from the stakeholder.

3.4.2 Sprint 2

This sprint involves in creating the backend structure of the project, creating an admin exclusive page for scraping by introducing token for authentication and session, creating Mongoose Models for database integration, loading the grants from the database and show in the grants list, introducing the search functionality, creating the visualization of Grants Gov website.

3.4.3 Sprint 3

In the last sprint I completed designing the proper UI flow of the system, design the page to view the extracted grants. The scrapping process for the two websites were completed without any error and the data was saved in the database. A dashboard was created which comprised of all the visualizations of the different data obtained from scrapping. The system was tested by the stakeholder and their reviews were taken into count for further system improvements

3.5 Technology Used Description

The Research Finder system will be a web-based project. There are uncountable websites available on the internet. There are many different types of technologies that are available to design web-based software. Each web language has its own unique characteristics. So, the language is chosen according to the requirements of the system. Some of the languages for web development are Python, CSS, JavaScript, ReactJS, NodeJS etc.

Since the Research Grant Finder will be using web scraping technique and will have a dashboard to view the analysis of the data, the system will be developed using the Express framework of NodeJS and will use JavaScript library called Puppeteer since the language of the browsers is mostly JavaScript, it will be easier to get hold of different elements of the browser and extract data from it. As the system will have to store a large amount of data in the repository, a database language is needed to manage those data. The database system that will be used is MongoDB. By using these technologies, it is expected that the system can be built properly functional and adaptive.

3.6 System Requirement Analysis

This section includes the hardware and software that are needed to develop the system. By using the proper hardware and software a user can run the system properly and efficiently.

3.6.1 Hardware Requirements

This section gives an idea about the hardware needed to develop the Research Finder.

3.6.1.1 PC/ Laptop

A working PC or laptop is needed for the documentation of the system and also for doing the coding and testing the system.

Table 3.1: Hardware Requirements

Hardware	**Minimum Requirements**
Processor	I3 6th gen or equivalent
Type of Operating System	64-bit
Random Access Memory (RAM)	4GB
Disk space	256GB
Minimum free disk space	15GB
Input device	Keyboard, mouse
Output device	Monitor, Printer

3.6.2 Software Requirements

There are many software that are required to build a web application. This section will describe the software that are required to develop the Research Grant Finder system.

3.6.2.1 Visual Studio Code

Visual Studio Code is a source-code editor that will be used as the main platform for coding the application in this project. Visual studio code contains different plugins for different languages which help writing codes easily and nicely format them. It also has a code runner to run the code.

3.6.2.2 Enterprise architect

Enterprise architect provides a platform for creating, visualizing, and managing enterprise architecture artifacts and models. A variety of features and functions are available in enterprise architect software to help software designers with their work. It enables the creation and upkeep of architectural diagrams, including infrastructure designs, component models, architectural models, and package diagrams.

3.6.2.3 Windows 7/8/10 or equivalent MacOS

The operating system is necessary for the computer or laptop to function.

3.6.2.4 Google Chrome

Google chrome is the web browser that will be used to run the system and also to browse through various websites for resources and tools.

3.6.2.5 Microsoft Word

Microsoft Word will be used for documentation.

3.6.2.6 Microsoft Excel

Microsoft Excel will be used for arranging the extracted data from websites through web scraping.

3.7 Chapter Summary

Video provides a powerful way to help you prove your point. When you click Online Video, you can paste in the embed code for the video you want to add. You can also type a keyword to search online for the video that best fits your document. To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that complement each other. For example, you can add a matching cover page, header, and sidebar.

CHAPTER 4

REQUIREMENT ANALYSIS AND DESIGN

4.1 Introduction

This chapter discusses the requirement analysis and design of the Research Grant Finder System. This part consists of the functional and non-functional requirements of the system. This document will describe the Software Requirement Specification (SRS), Software Design Documentation (SDD) and Software Testing Document (STD) in short. Various diagram such as Use Case, Sequence diagram and activity diagram are drawn for each use case in details. Each use case also has its specification where the details description of the use cases are given. Finally the interface of the system is designed to provide a clear idea about the system.

4.2 Requirement Analysis

By doing consecutive meetings with the stakeholders, several functional and non-functional requirements of the system were identified which are discussed below:

4.2.1 Functional Requirements

I. The application allows user to login to the system.
II. The admin can insert user data into the system.
III. The admin can modify and delete user data.
IV. The admin can view the extracted data from the web scraping.
V. The admin can filter the extracted data.
VI. The admin can upload the data to database.
VII. The admin can transform and operate on the data.
VIII. The users can view the dashboard.

IX. The users can search for grants.

X. The users can save grants information.

4.2.1.1 Use Case Diagram

The use case consists of all the functionalities and the actors that are involved in the system. It gives a clear idea of how the system works and what are the roles of each actors in the system. The figure below represents the use case diagram of the Research Grant Finder system.

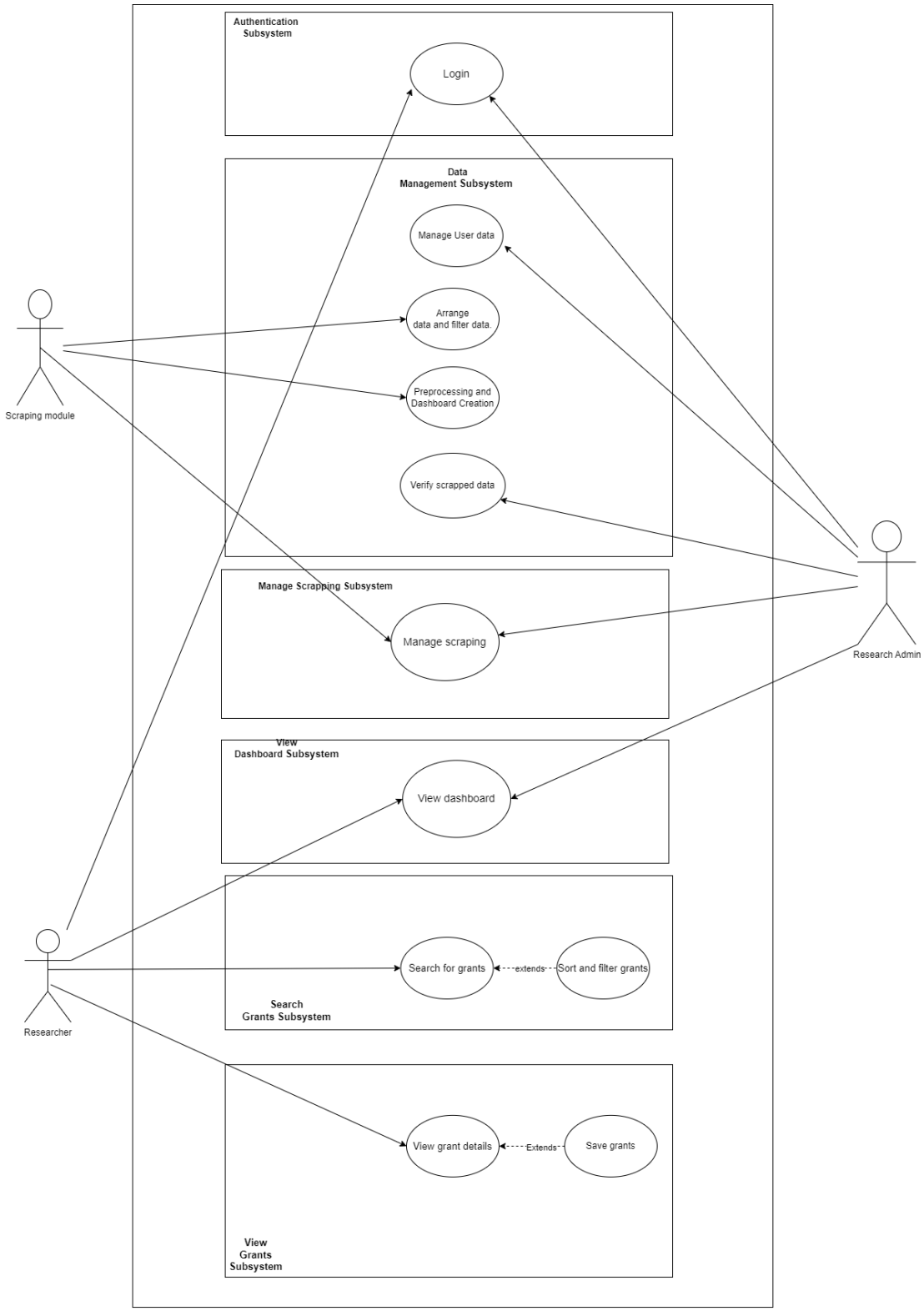


Figure 4.1 Use Case Diagram of Research Grant Finder system.

4.2.1.2 Use Case Actor Description

Table 4.1: Actor Description

User	Characteristics
Research Admin	Oversees the system functionalities, create user, delete users, manage scrapping, view dashboard and other maintenance tasks
Scrapping Module	Clean the data, process the data into a file.
Researcher/ End user	Views the dashboard, access the data, search for grants, view and save them.

4.2.1.3 Use Case Description

Fig 4.2: Use Case Description

Module	**Use cases**	**Description**
Authentication	Login	Allows the users to access the system by entering their valid details.
Data Management	Manage User Data	Allows the admin to insert, create and update the user information or delete them.
	Arrange and filter data	Allows the research heads and manager to reorganize the extracted data, filter it according to the necessary categories ant make it ready for preprocessing.
	Data Preprocessing & Dashboard Creation	Allows the system to run the scrapping functions,

		to process the data and then upload it to the database for visual representation in the form of a dashboard.
	Verify the scrapped data	This use case allows the research admin to delete the grants which are irrelevant and with wrong information.
Manage Scraping	Manage Scraping	Allows research admin to manage the web scraping process by letting him/her modify the different parameters that control the scrapping.
View Dashboard	View Dashboard	Allows all the actors in the system to view the final product which contains the visualization of all the extracted data.
Search Grants	Search for grants	Allows the user to search for grant using specific keywords.
	Filter and sort grants	Allows to narrow the search by selecting specific categories form the data.
View Grants	View Grant Details	Allows the user to view each grants information in details.

	Save Grants	Allows user to save grants and integrate their data with their profile.

4.2.1.4 Use Case Specification

Table 4.3: Use Case specification

Use Case ID	UC
Use Case Name	**Data Preprocessing and Dashboard Creation**
Description	The scrapping functions get the data and then it is filtered and processed by removing unwanted data.
Actor(s)	Scrapping Module
Pre-conditions	The scrapping function has been completed successfully.
Normal Flow	1. Get the extracted data. 1.1. System looks for extracted data in the web scraping module. 1.2. System retrieves the data for preprocessing. AF1 will be executed. 1.3. System does not find any data, EF1 will be executed. 2. Filter the data. 2.1. The module filters the data that is extracted. 2.2. The module excludes the data that is undefined or have null value. 2.3. Separate different properties that are extracted according to categories.

	2.4. Transform data. 3. Modify the data. 3.1. The scrapping module changes the data that is not understandable into an easier form. 3.2. Gets the data ready for visualization. 4. Confirm the changes. 4.1. The data is made ready for visualization, 4.2. The data is uploaded to the database.
Alternative Flow	1. If the data is not found by the system return error to the admin. 2. If the network is changed go to NF1.
Exception Flow	1. The system shows an error message, and the process ends. 2. Admin clicks on discard changes. 2.1. System displays error message. 2.2. Data not saved in database.
Post Conditions	The modified pre-processed data is stored in the system database.
Related Requirements	

4.2.2 Non-Functional Requirements

The non-functional requirements of the system include Usability, reliability, maintainability, and security. The SRS provides a clearer description of the non-functional requirements of the system.

4.2.3 Sequence Diagrams

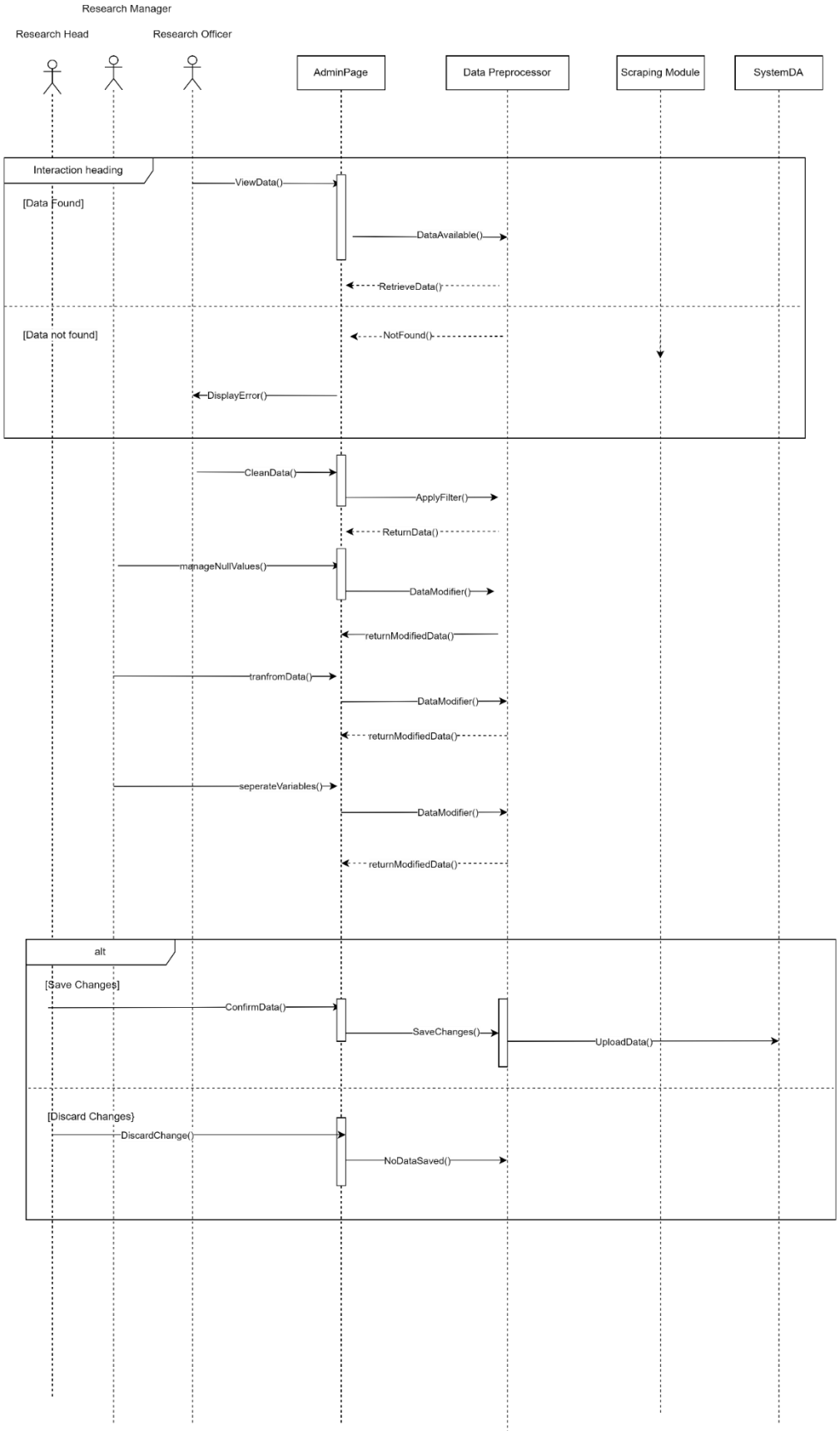


Figure 4.2: One of the sequence diagrams of Research Grant Finder system

4.2.4 Activity Diagram

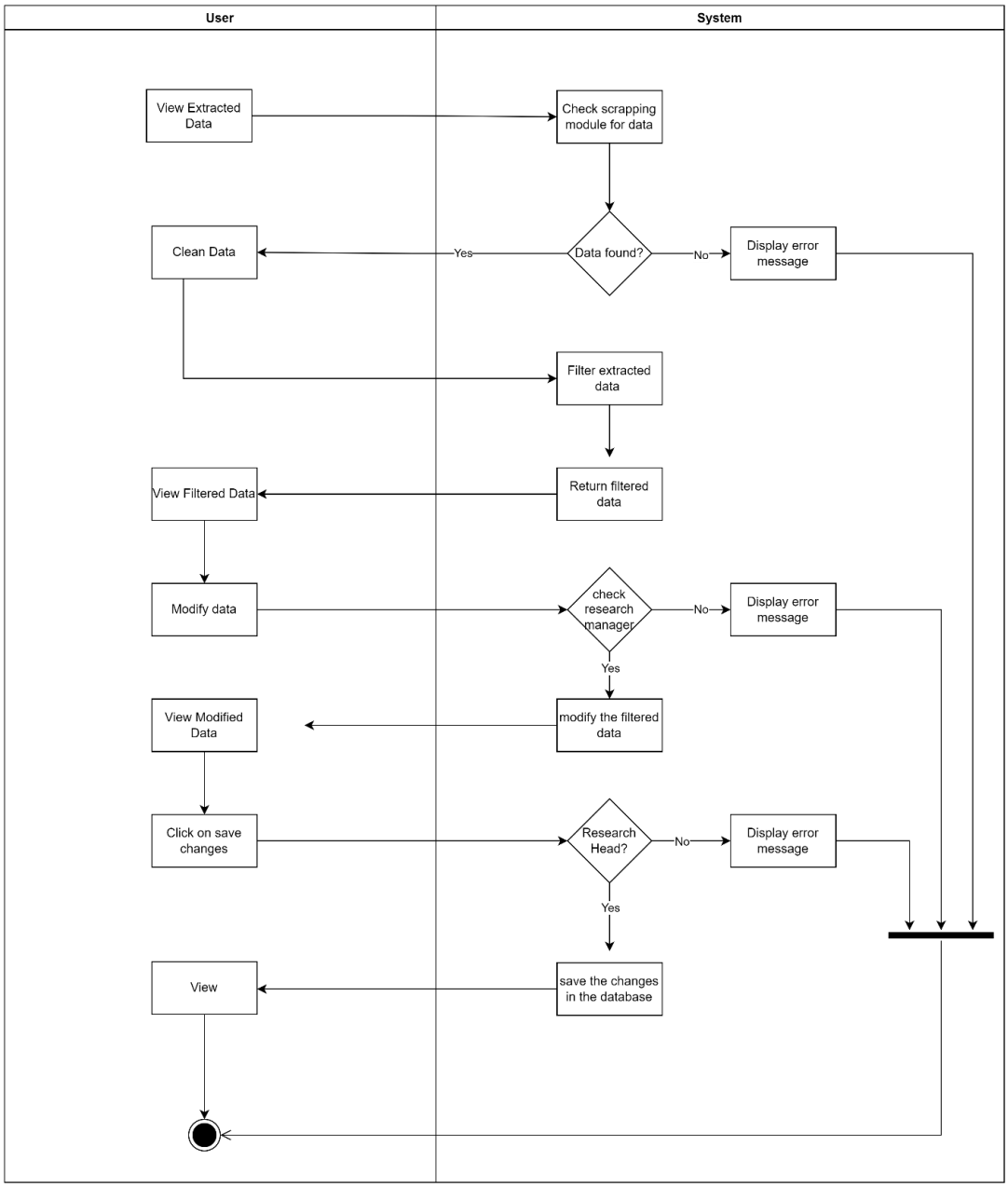


Figure 4.3: One of the activity diagrams of Research Grant Finder system.

4.3 Project Design

4.3.1 Architecture Model

The system architecture that will be used for this system will be MVC or Model-View-Controller. This architecture contains 3 layers which are model, view and controller.

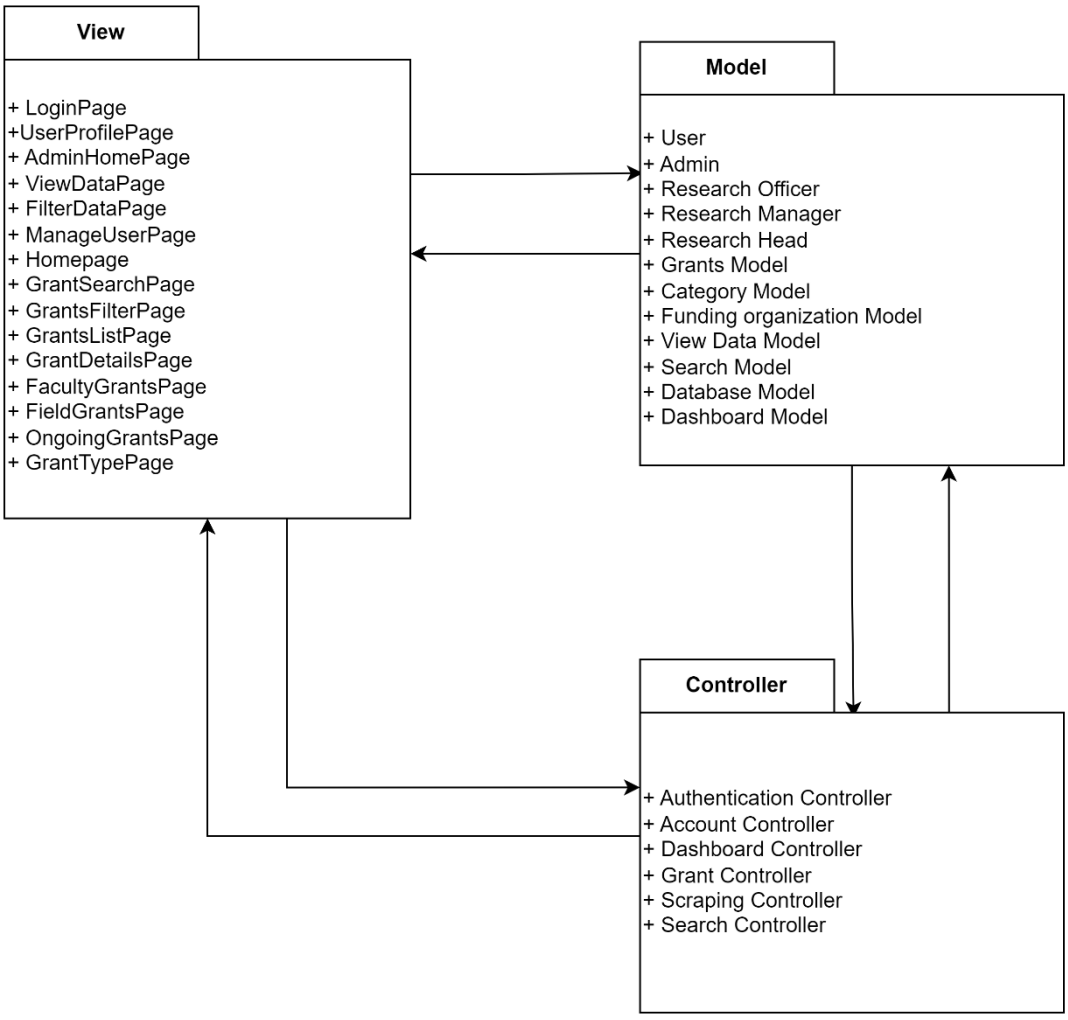


Figure 4.4: Architecture model of Research Grant Finder System

.

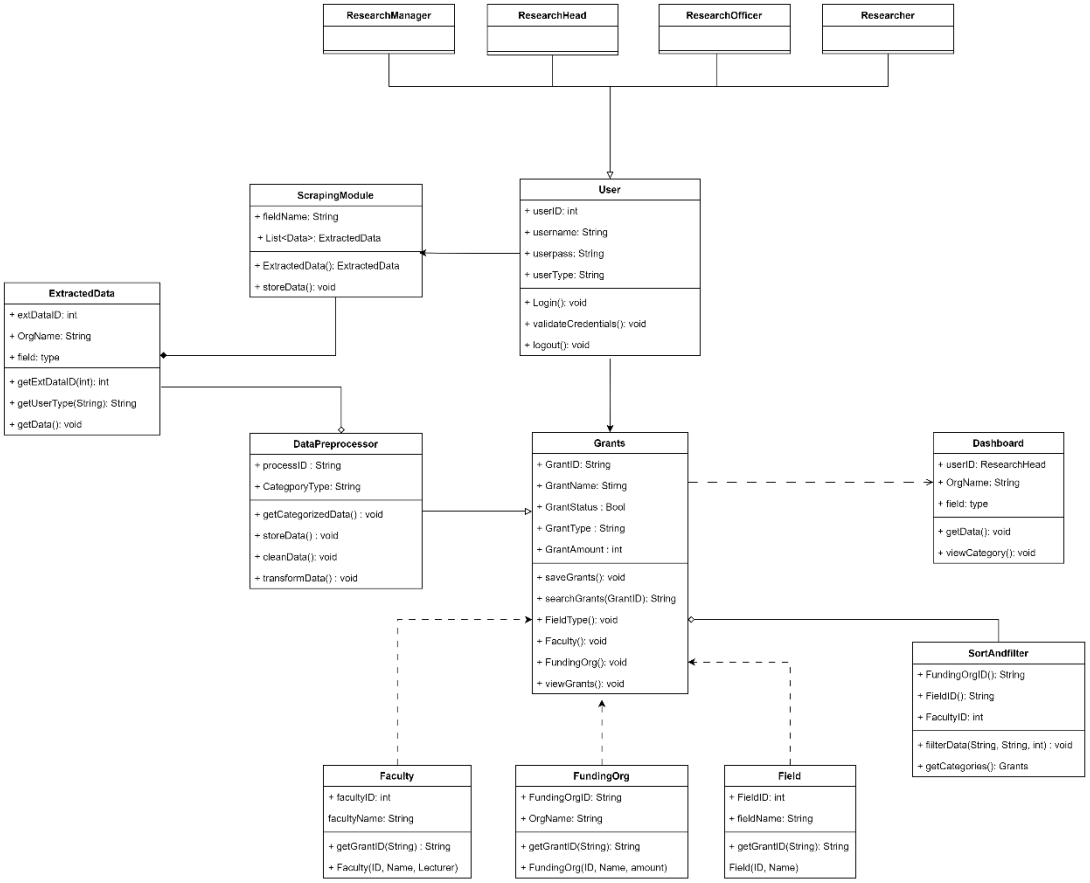
Model: The Model represents the application's data and business logic. It contains the data of the application and defines how it may be accessed, edited, and

validated. It's in charge of data manipulation, storage, and retrieval. The Model frequently interacts with the database to fetch or update data in the context of a database-driven application. It should, however, be independent of the specific data storage implementation.

View: The View is in charge of displaying the application's user interface to end users. It specifies how data will be displayed and how users will interact with it. The Model sends data to the View, which formats it for presentation by producing HTML templates or showing GUI elements. It should not include any business logic and should instead concentrate on visualizing the data provided by the Model.

Controller: The Controller functions as a go-between for the Model and the View. It processes user input and events, changes the Model, and picks the appropriate View to present the changed data. The Controller takes user input, such as button clicks or form submissions, and converts it into actions that change the state of the Model. It also connects with the View in order to adjust the user interface based on Model changes.

4.3.2 Class Diagram



4.4 Database Design

This section describes how the database of the Research Finder System was designed. Database design is very important in developing and implementing the system and for the system to work properly.

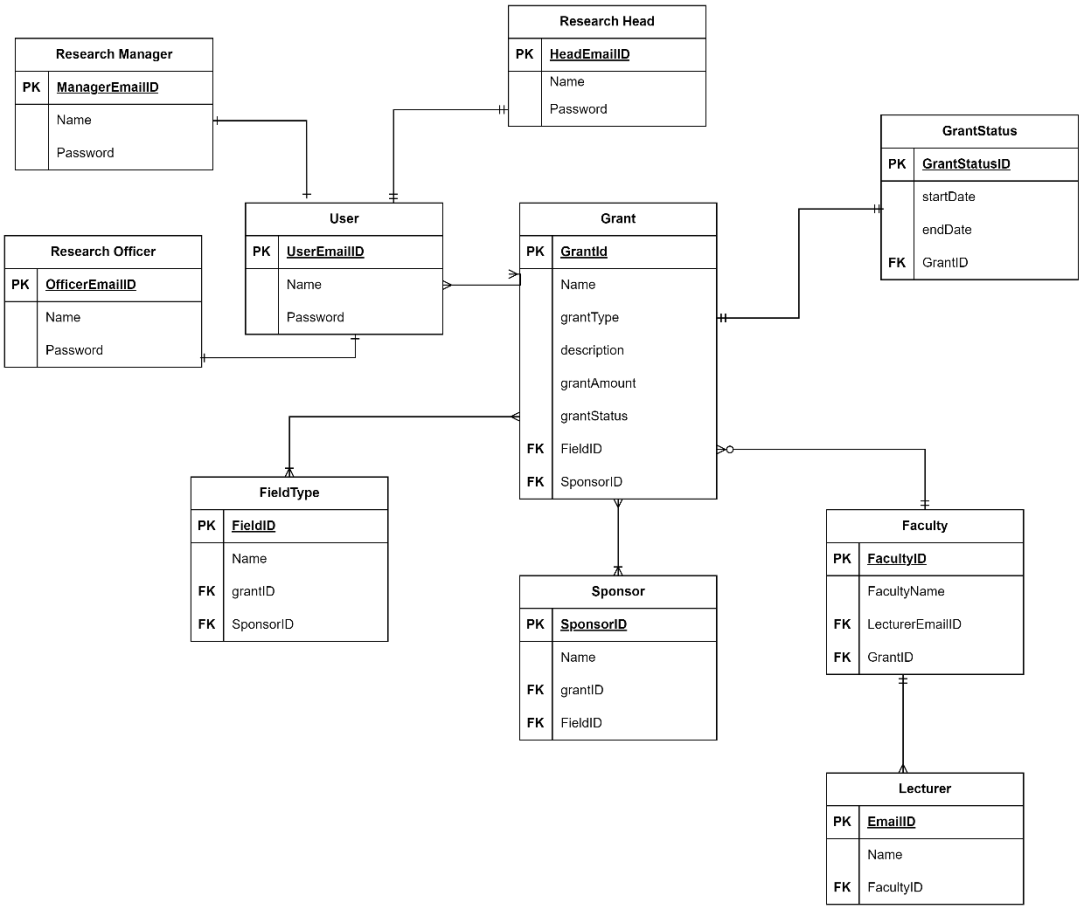


Figure 4.5: Database Diagram of Research Grant Finder system.

4.4.1 Data Dictionary

A data dictionary provides information about the data that are stored in the database. The data are discussed below:

Table 4.4: Data dictionary of the system

Field Name	Datatype	Constraints	Description
Research Manager			
ManagerEmailID	String	Primary Key	Unique ID for research manager
Name	String	Not Null	Name of the research Manager

Password	String	Not Null	Password that the research manager uses to log into the system.
Research Head			
HeadEmailID	String	Primary Key	Unique ID for research head
Name	String	Not Null	Name of the research Head
Password	String	Not Null	Password that the research head uses to log into the system.
Research Officer			
OfficerEmailID	String	Primary Key	Unique ID for research officer
Name	String	Not Null	Name of the research officer
Password	String	Not Null	Password that the research officer uses to log into the system.
Field Type			
FieldID	Int	Primary Key	Unique ID for each field of research.
Name	String	Not Null	The name of the fields.
GrantID	Int	Foreign Key	The grantID to get the grants

			related to the field.
FundingOrgID	Int	Foreign Key	The FundingOrgID to get the sponsors related to the field.
Grant			
GrantID	Int	Primary Key	Unique ID for each grants.
Name	String	Not Null	The title of each grants.
grantType	String	Nullable	The type of grants.
Description	String	Nullable	The details of each grant.
grantAmount	Int	Not Null	The amount offered in each grants
grantStatus	Bool	Not Null	Determines if the grant is over or ongoing.
FieldID	Int	Foreign Key	Access the field of research related to grants
FundingOrganizationID	Int	Foreign Key	Access the funding organizations that provide grants.

Funding Organization			
FundingOrganizationID	Int	Primary Key	Unique ID for representing the funding organization.
Name	String	Not Null	Name of the funding organization
GrantID	Int	Foreign Key	Access the grant information that have the same funding organization.
FieldID	Int	Foreign Key	Access the number of fields an organization is funding.
Faculty			
FacultyID	Int	Primary Key	Unique ID for representing faculty.
FacultyName	String	Not Null	Provided the name of the faculty.
LecturerID	Int	Foreign Key	Access the lecturers in a faculty
GrantID	Int	Foreign Key	Access the grants by each faculty.
Grant Status			

GrantStatusID	Bool	Primary Key	Stores the true or false value if the grant exist or not.
startDate	Date	Not Null	The start date of the grant
endDate	Date	Not Null	The ending date of grants.
GrantID	Int	Foreign Key	Access the grant information to update the status.
Lecturer			
LecturerID		Primary Key	Unique ID to represent the lecturer of faculty.
Name	String	Nullable	Name of the lecturer.
FacultyID	Int	Foreign Key	Access the faculty details the lecturer belongs to.
FieldID	Int	Foreign Key	Access the field of research of the grants
GrantID	Int	Foreign Key	Access the grant information.

4.5 Interface Design

The interface design helps us to get a blueprint of how the system works and how it will look like once it is developed. It is a very important part. It shows the stakeholders about the proposed system and gives them more room for suggestions during development.

4.6 Chapter Summary

Finally, the full architecture and design of the system have been demonstrated and planned in this chapter. Each use case, class, and component diagram has been thoroughly described. All of the demands for the Research Grant Finder system, including functional and non-functional requirements, have been outlined in the use case diagram for the complete system. Furthermore, the full explanation will be included in the Software Requirements Specification (SRS) and Software Detailed Design (SDD)

CHAPTER 5

SYSTEM DEVELOPMENT METHODOLOGY

5.1 Introduction

This chapter mainly discusses on how the system has been developed and the processes and the methods that have been implemented for developing the Research Grant Finder system. The implementation includes the frontend interface design, setting up the backend, database and also implementing the coding to develop the functionalities of the website according to the requirements. This chapter includes the code snippets of the important functions of the system, the implementation of architecture of the system to provide an idea of the system implementation of Research Grant Finder. In addition, this chapter also discusses the testing procedures used to test if the different functionalities are working properly and there is error validation.

5.2 Development Environment

The Research Grant Finder is a system that depends on data extraction from the target websites and then analysis them. As the stack that is being used for development is MongoDB, Express Js, React, Node Js (MERN), so the most suitable tool that can be used for the extraction is Puppeteer library of Node Js. The extracted data is stored in the MongoDB in the JSON format. Then the frontend calls the data from the backend after getting the data from the database. The backend contains the node files with the help of express framework. The front-end react components visualize the data according to the requirements by importing the react chart js library for creating interactive charts like bar chart, pie chart, doughnut, line chart etc. The front end and the back end are hosted in two different ports. Other libraries like Axios and middleware like CORS are used in the project so that the backend and the frontend can communicate with each other.

5.3 Implementation

Implementation is the phase where the system design and the architecture deviated from the requirement analysis are converted in codes so that the system can start working and perform the identified objectives. After the backend and frontend is initialized, the development of the system is started. The frontend contains of numerous React components which can be reused in different pages and provide different information. Some frontend components also have JavaScript logic to process data. The backend contains the main Puppeteer files which are the main source of the data of the whole system. These files are the backbone of the Research Grant Finder. The code implementation of different functionalities with a snapshot are given below:

5.3.1 Views Implementation

It consists of the frontend interfaces of the system including login, create user, dashboard, scrapping page.

The login page is the page where the user enters the credentials and then gets access to the system.

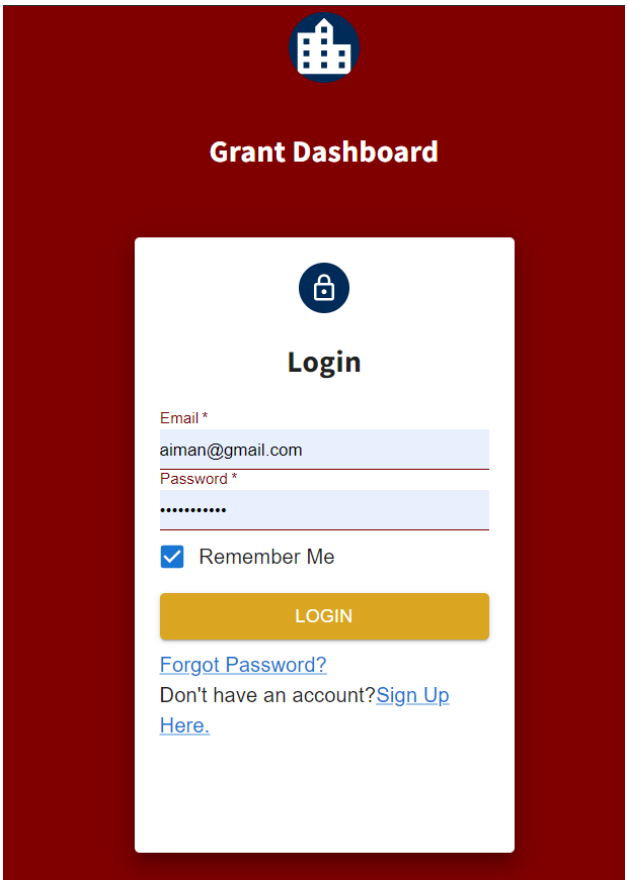


Figure 5.1: Login Screen of Research Grant Finder system.

The admin can create user and then the created user can login using the credentials provided by the admin.

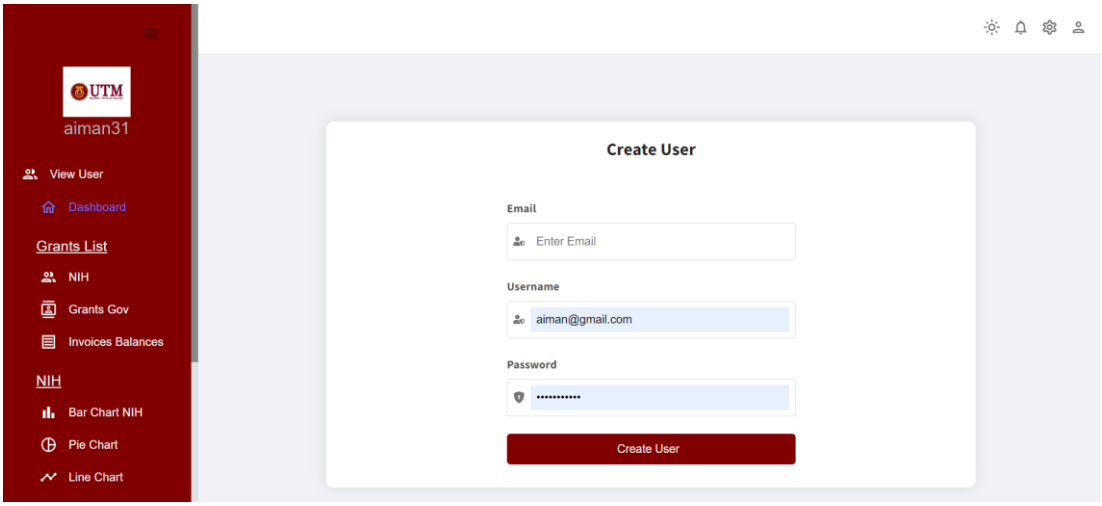


Figure 5.2: Create User Screen of Research Grant Finder system

The scraper screen is where the admin starts the scrapping. The puppeteer functions to extract the data are executed here when the scrape button is clicked.

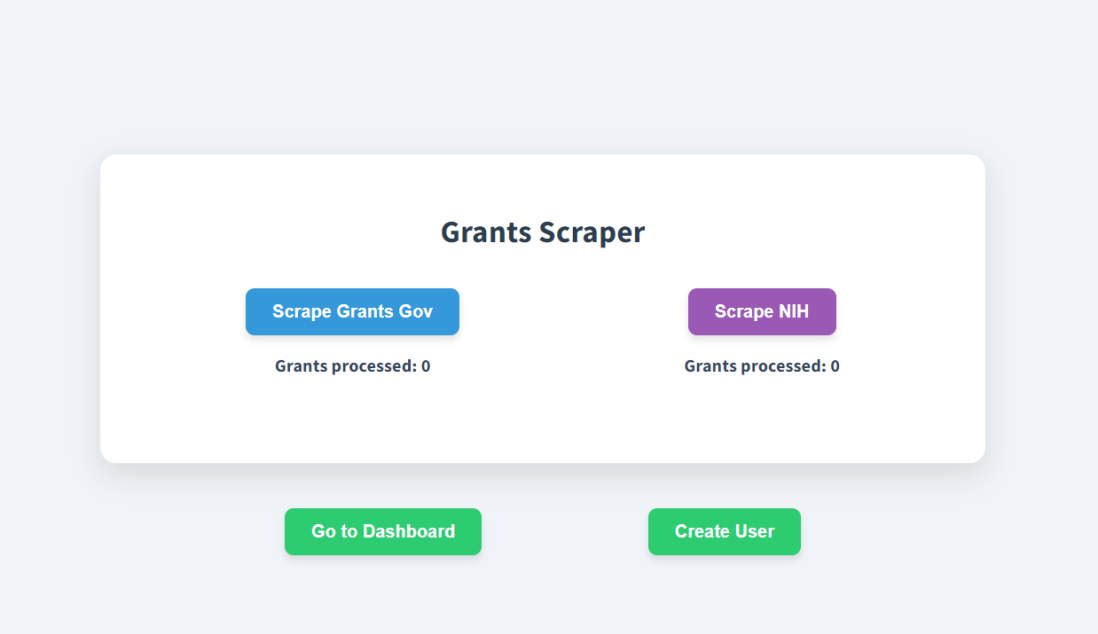


Figure 5.3: Scraper Screen of Research Grant Finder system

There are different pages in the system that have different visualizations of the data in charts.

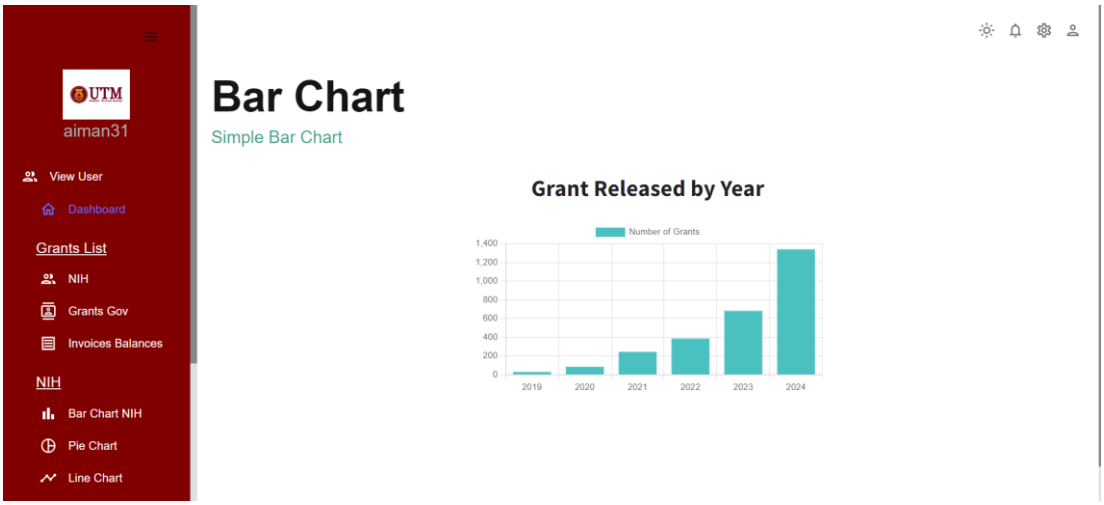


Figure 5.4: A bar chart showing data analysis

5.3.2 Coding Implementation

The coding implementation includes the coding structures of the system. Like the database integration starting express server, including the frontend components, backend controllers, database models.

```
app.use(express.json())
app.use(cors())

mongoose.connect("mongodb://localhost:27017/grants")
```

Figure 5.5: Initializing database connection

An important implementation was to declare the api endpoints in the backend responsible for the scrapping which is illustrated in Figure 5.6.

```
app.get('/scrape', async (req, res) => {
    try {
        await extractGrantGovData(); // having some problem extract only 96 page
        res.status(200).send('Scraping completed and data saved to JSON file.');
    } catch (error) {
        console.error(error);
        res.status(500).send('Error occurred during scraping.');
    }
});
```

Figure 5.6: Api endpoint for scrapping

Many frontend react components have been implemented for the system which calls the api from the backend to view the data in the form of a graph.

```jsx
import { Box } from "@mui/material";
import Header from "../../components/Header";
import BarChart from "../../components/BarChart";
import Sidebar from "../global/Sidebar";
import Topbar from "../global/Topbar";
import { useNavigate } from "react-router-dom";
import { useState, useEffect } from "react";
const Bar = () => {
  const [isSidebar, setIsSidebar] = useState(true);
  const token = localStorage.getItem("token");
  const navigate = useNavigate();
  useEffect(() => {
    if (!token) {

      navigate("/login");
      alert("You need to login first")
    }
  }, [token, navigate]);
  return (
    <div className="app">
      <Sidebar isSidebar={isSidebar} />
      <main className="content">
        <Topbar setIsSidebar={setIsSidebar} />
        <Box m="20px">
          <Header title="Bar Chart" subtitle="Simple Bar Chart" />
          <Box height="75vh">
            <BarChart />
          </Box>
        </Box>
      </main>
    </div>
  );
};

export default Bar;
```

Figure 5.7 Frontend react component

5.4 Architecture Implementation

The architecture that the Research Grant Finder follows is the Model-View-Controller architecture. The model contains the database models of the system. The controller contains the logic that help to communicate the frontend with backend. The views consist of the react components.

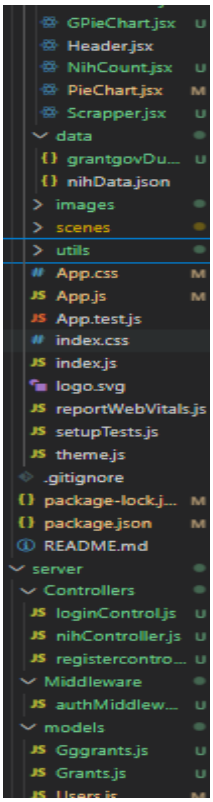


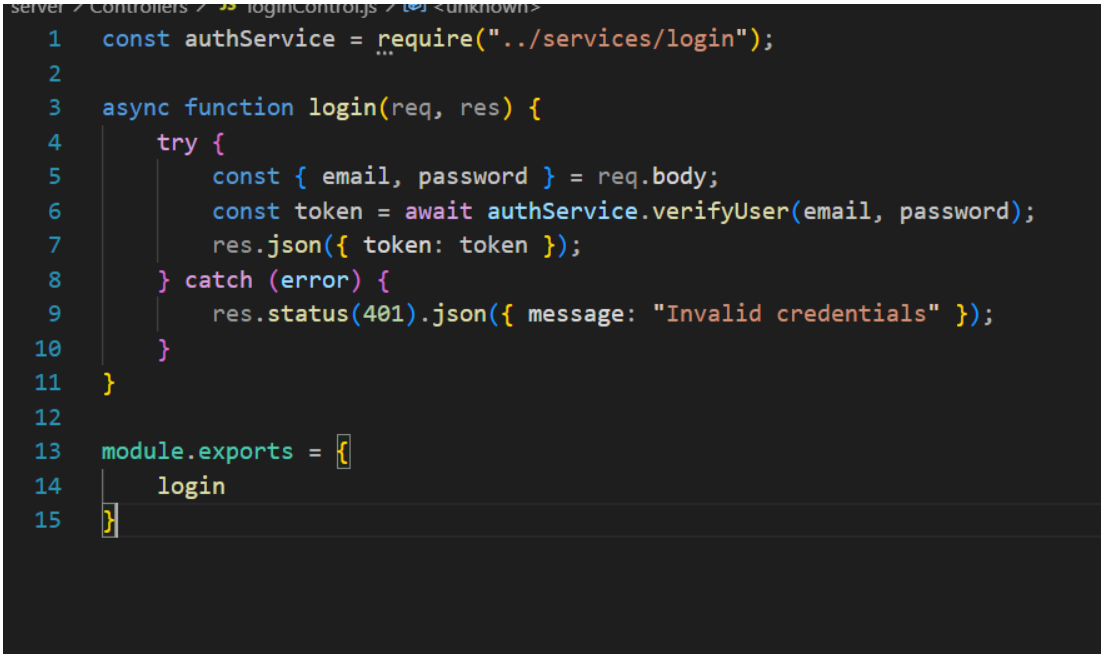
Figure 5.8 MVC structure

The model contains the database models of the system.

```
const mongoose = require('mongoose')
//This is a comment
const UsersSchema = new mongoose.Schema({
    email: {
        type: String,
        required: true
    },
    name: {
        type: String
    },
    password: {
        type: String,
        required: true
    },
    isAdmin: {
        type: Boolean,
        required: false
    },
})
const UserModel = mongoose.model("users", UsersSchema)
module.exports = UserModel
```

Figure 5.9 structure of User Model

The controller contains the logic that helps to communicate the frontend with backend. The views consist of the react components.



```
server > Controllers > loginControl.js > <unknown>
1   const authService = require("../services/login");
2
3   async function login(req, res) {
4       try {
5           const { email, password } = req.body;
6           const token = await authService.verifyUser(email, password);
7           res.json({ token: token });
8       } catch (error) {
9           res.status(401).json({ message: "Invalid credentials" });
10      }
11  }
12
13  module.exports = {
14      login
15  }
```

Figure 5.10 Controller that controls authentication of users

5.5 System Testing

After the implementation of the Research Grant Finder, the most important task is to test if the system is working as per the requirements and the design. Different types of testing can be conducted on a system that has been developed. Such as: black box testing, white box testing, user acceptance testing. Each of these techniques has their unique features and this testing helps the system to become more robust. The testing that will be carried out for the Research Grant Finder are Black Box Testing and User Acceptance Testing.

5.5.1 Black Box Testing

Black-box testing is a software testing technique which involves reviewing the functionality of an application by testing. It does not need the code of the system to do black box testing. Mainly for a test case the inputs and outputs of the software system

are examined. This testing ensures that the system behaves as it has been described in the documentations.

5.5.1.1 Test Case Design

Test TC001 for Module <Authentication>: <Login (UC001)>

This test contains the following test cases:

UC001_01: Successful login

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the login page.	-	System displays the login page.	The login page is displayed	Pass
2	The user enters the correct user details	Email: abdur@gmail.com Password: a@34ruz	The system successfully logs in according to the user credentials.	Login successful	Pass
3	The user clicks on the login button.	-	The system verifies the user successfully.	Only verified users can access the system	Pass

UC001_02: Invalid Login details

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the login page.	-	System displays the login page.	The login page is displayed	Pass
2	The user enters the details	Email: abdur2 Password: a@34ruz	The system displays the user input.	Successful display of user input	Pass
3	The user clicks on the login button.	-	The system displays error messages.	Error message is displayed as alert	Pass

UC001_03: Incorrect Login details

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the login page.	-	System displays the login page.	The login page is displayed	Pass
2	The user enters the details that is not in the database	Email: ruz@gmail.com Password: a@34ruz	The system displays the user input.	Successful display of user input	Pass
3	The user clicks on the login button.	-	The system displays error message because the data did not match with the database.	Error message is displayed for bad credentials	Pass

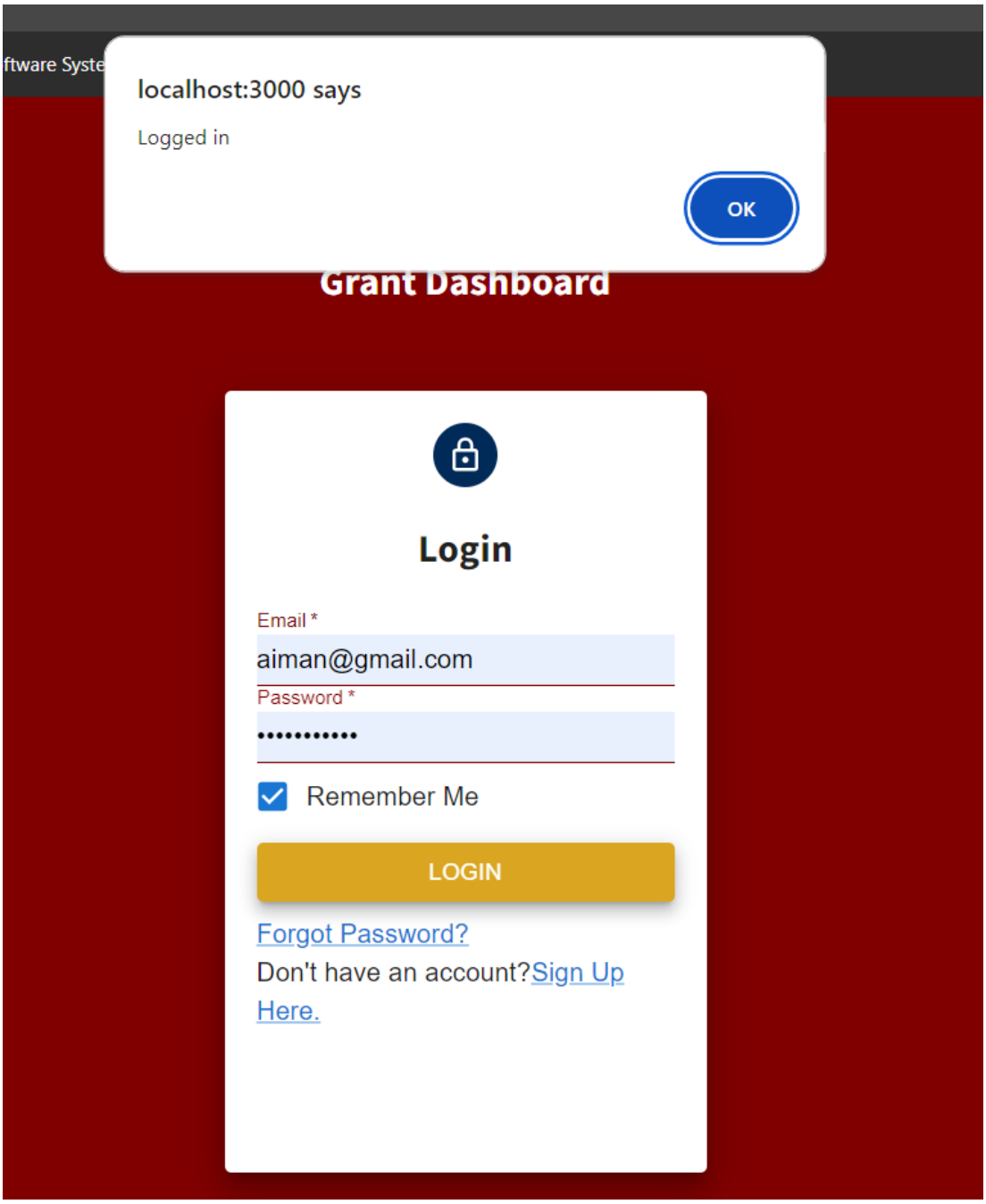


Figure: Successful Login

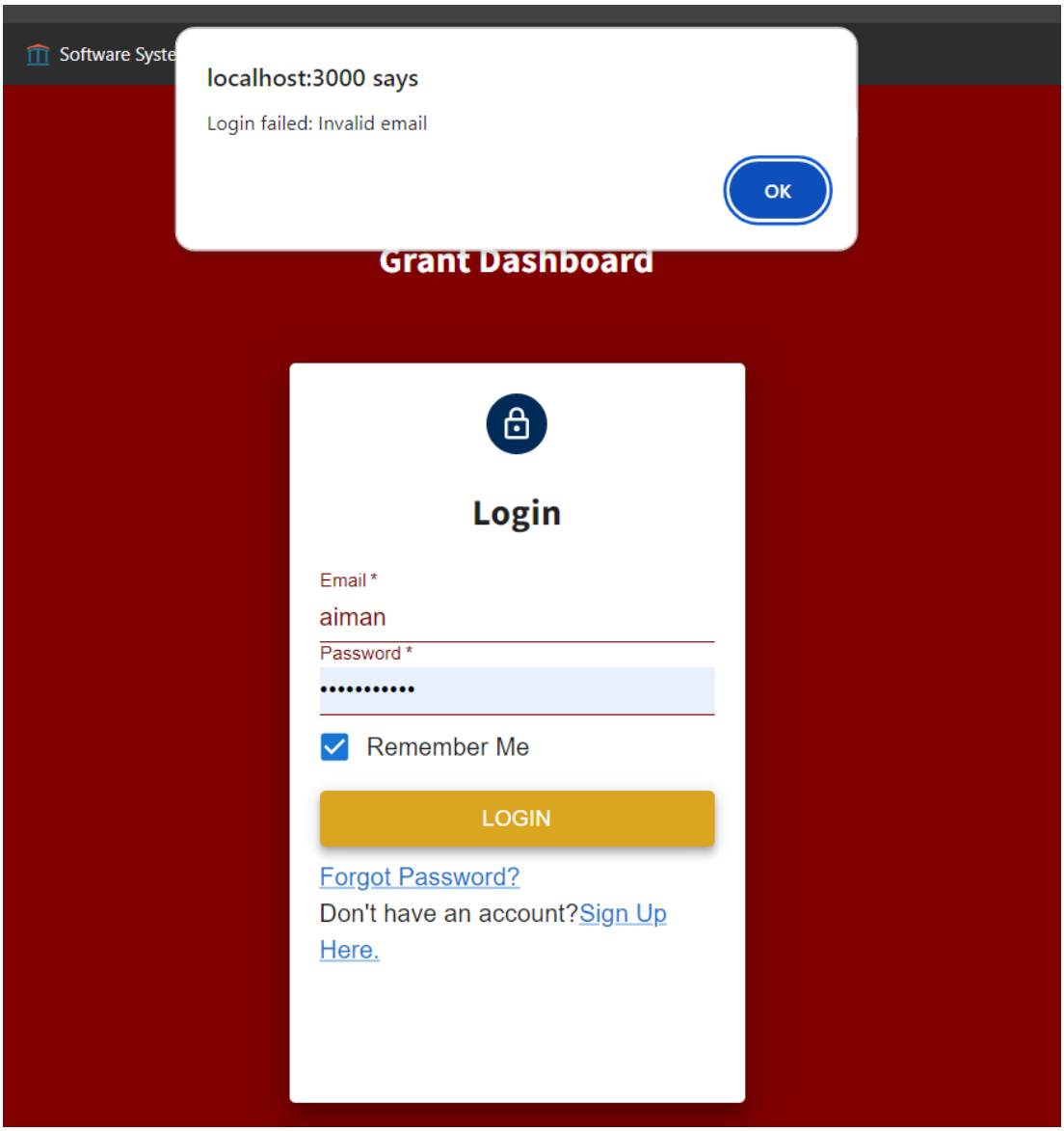
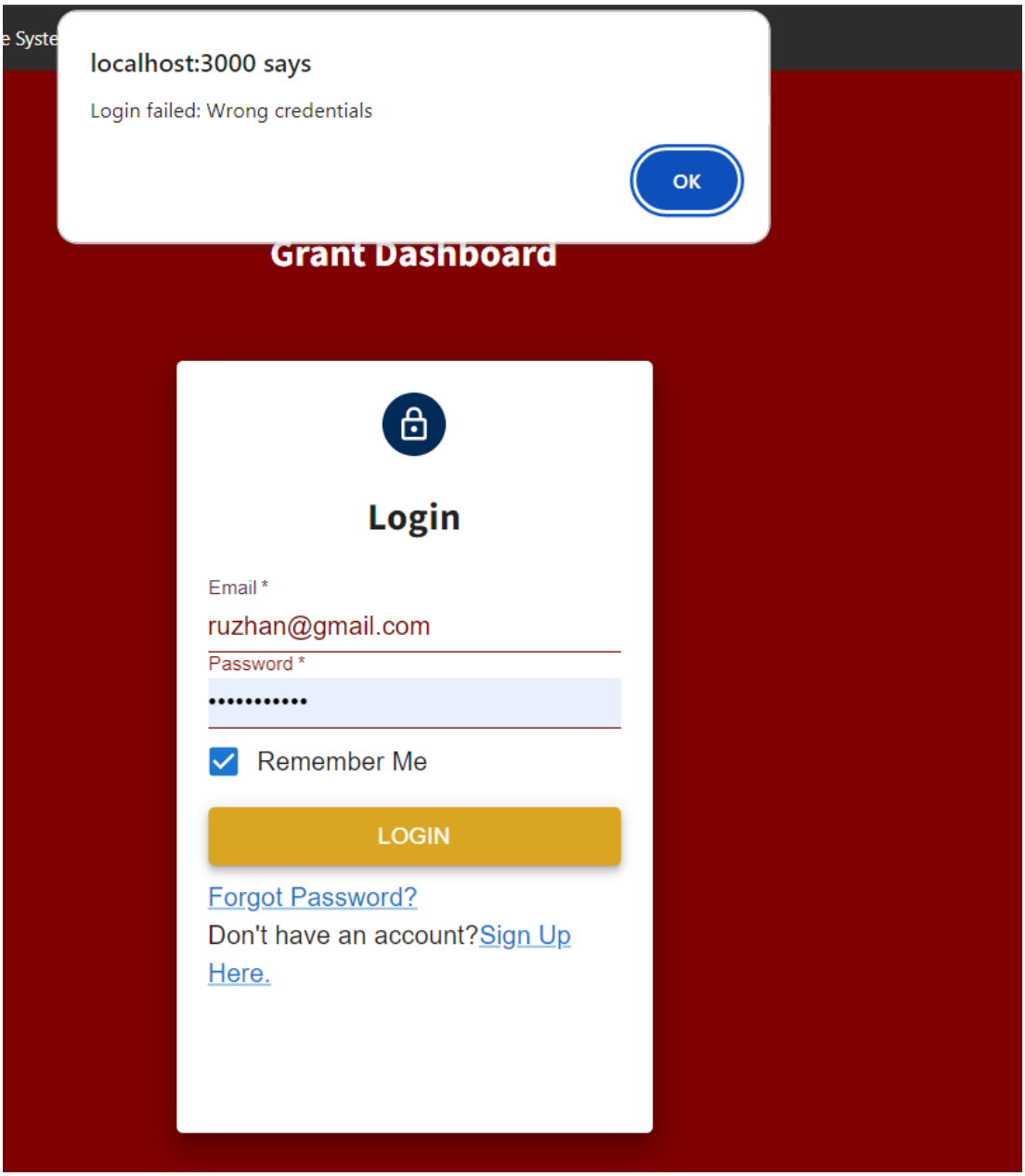


Figure: Error Validation Login



5.5.2 User Acceptance Testing

User Acceptance Testing or UAT is a type of testing where the stakeholder himself tests the desired functionalities of the system and check if it has met their requirements. For the UAT of Grants Research Finder, the stakeholder was asked to test the functionality of scrapping.

5.5.2.1 Scraping Functionality

No.	Function	Description	Performed By
1	Scraping the websites	This function allows the user to scrape the grants information and see the progress of extraction from the target websites.	Admin

No.	Actions	Expected Results	Pass/ Fail	Comment
1	1. The admin logs in 2. The scrapping page is viewed 3. The admin clicks on the button according to the website he wants to extract	The scraping starts and after some time the scraping is completed with a success message.	Pass	

CHAPTER 6

CONCLUSION

6.1 Introduction

The achievements, outcomes and the future improvements to the Research Grant Finder system is dicussed in this chapter. The primary objective of developing this system is to fulfill the requirements of the stakeholders and develop a system that will be able to ease the grant searching process of the researchers. This system provided an indepth analysis of the extracted grants from multiple websites, which will be very beneficial to its users. By the help of this system, the researchers can look for grants in a much more less hectic and easy way and can get more information by spending less time comparatively than before. Moreover, this system has immense potential for future updates by adding more grant websites for extraction and analysis. The next step will be designing and implementing new changes to the system so that the functionality of the system is increased a lot more in the future.

6.2 Achievement of Project Objective

In PSM 1, the requirements for the system were discussed with the stakeholders. A literature review of existing systems similar to the proposed system is conducted, and the comparisons between the proposed system and the existing systems are outlined in Chapter 2. The procedure that was used in the system's creation is briefly explained and discussed, as well as provided in Chapter 3. The system requirements are broken down into digestible portions and thoroughly specified within the SRS. SDD contains details regarding the general design of the system. STD also includes a listing of the system's test cases.

The documentations of PSM1 were a key resource in the successful development of the system for PSM2. Following the design methods and the requirements stated in the PSM 1 documentation, the Research Grant Finder has been developed and all the core functionalities have been implemented.
During the agile development stage, the development of the system was tested in every iteration and the progress was discussed with the stakeholders. And after all the implementations and corrections, the result is a complete Grant Finder system which can analyze the grant data and allow users to know about the grants information from multiple websites

6.3 Suggestions for Future Improvement

To keep up with the rapidly changing software industry, every app or system needs to be updated with time. So definitely there maybe suggestions and room for improvements and enhancing functionalities. The functionalities that can be introduced to Research Grant Finder are:

- Including more websites to extract the data from
- Check for a possibility of integrating AI to the system to cope with the present trend.
- Introducing custom filters according to user preferences.
- Improve the scrapping efficiency of the web scrapping.
- Increase the interoperability of the system so that it can be switched to other technologies easily and with less cost and maintenance.
- Release this system for the whole world for large scale use.

REFERENCES

Bogler, R. (1994). The impact of past experience on people's preference: the case of university researchers' dependency on funding sources. *Higher Education*. https://doi.org/10.1007/bf01383727

Burgelman, R. A., Maidique, M. A., & Wheelwright, S. C. (1996). Strategic Management of Technology andInnovation. (2nd ed.). Chicago: I. L, Irwin.

APPENDIX A: SOFTWARE REQUIREMENTS SPECIFICATION

Software Requirements Specification

Research Grant Finder System

Version 1.0

25/06/2023

School Of Computing

Prepared by: Islam Mohammed Ruzhan

Revision Page

a. Overview

Describe the content of the current version.

b. Target Audience

State the targeted audience.

c. Project Team Members

List the team members and respective assigned module.

d. Version Control History

Version	Primary Author(s)	Description of Version	Date Completed
<1.00 >	**Islam Mohammed Ruzhan**		**25/06/23**

Note:
This Software Requirements Specification (SRS) template is based on IEEE Std 830-1998, organized by modules according to system features (Appendix A.5 of the IEEE Std, 830-1998, Section 5) and customized to meet the need of SCSJ2203 course at Faculty of Computing, UTM. Compiled and checked by Shahida Sulaiman, PhD on 20 March 2016. Examples of models are from Satzinger (2011).

Table of Contents

3.1.4 Communication Interfaces

3.2 System Features

3.2.1 Module <Name of Module1>

3.2.1.1 UC001: Use Case <Name of Use Case1>

3.2.1.2 UC002: Use Case <Name of Use Case2>

3.2.1.3 UC003: Use Case <Name of Use Case3>

3.2.2 Module <Name of Module2>

3.2.1.1 UC004: Use Case <Name of Use Case4>

3.2.1.2 UC005: Use Case <Name of Use Case5>

3.2.n Module <Name of the *n* Module>

3.3 Performance Requirements

3.4 Design Constraints

3.5 Software System Attributes

3.6 Other Requirements

Appendices (if any)

1. Introduction

The Software Requirements Specification describes the Research Grant Finder in depth. This document includes a brief description of the functional and non-functional requirements. This document consists of the use case diagram which contains the subsystems involved in the system. It depicts the interaction between system and its users. Each subsystem has multiple use cases which describe the functionality of the subsystems. Each use case also consists of sequence and activity diagram which helps to show the overall flow of the system.

1.1 Purpose

The purpose of SRS are:

- To describe the system functionalities to the stakeholders.
- To give a clear idea about the flow of the system.
- Describes how the system will function.
- Serves as an input for System Design Document (SDD).

1.2 Scope

The system that will be build is Research Grant Finder. It is a web-based dashboard that will be created by using data visualization on the extracted data that will be obtained by web scraping from the target websites.

With the help of this system, the researchers can view the grant statistics of the faculty and other information such as numbers of grants received by respective faculty, Highest grant providing organization and amount of grant received by respective researcher. A researcher can also look for grants from the extracted data from the websites and filter his/her search. A researcher can also save the grants he wish to apply for in future. Overall, this web application will be very helpful for the research works of the university as it provides opportunities to researchers to apply for grants more easily than it is at present.

1.3 Definitions, Acronyms and Abbreviation

Acronyms	Definitions
	Faculty of Computing
SRS	Software Requirements Specification
UTM	Universiti Teknologi Malaysia

1.4 References

I. What are system requirements specifications? (2022) United States Headquarters. Available at: https://www.inflectra.com/Ideas/Topic/Requirements-Definition.aspx#:~:text=System%20Requirements%20Specification%20(SRS)%2C,behavior%20of%20a%20software%20application. (Accessed: 24 June 2023).

II. IEEE. IEEE Std. 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998

1.5 Overview

This document is divided into 3 sections which are:

I. Introduction

This part provides an overview of the whole document and what to expect in it.

II. Overall Description

This part gives an outline of the system's features, how they work together, and how they function. Section 2 also talks about how the requirements, assumptions, and dependencies are divided up and the structure of the system.

III. Specific Requirements

This part consists of the requirements of the system which are interface requirements, performance requirements and design constraints.

2. Overall Description

This section will contain an overall description of the system features, functionality, interactions and system constraints of the Research Grant Finder System. This system will include the use case of the system and detailed description of each of them so that the overall system and its flow is made understood to its readers.

To make this system more understandable to the stakeholders and the readers, the system has been divided into 6 subsystems, as the entire system is not easy to explain. Each of these

subsystems are combined together to develop the whole system. The 6 subsystems are Authentication subsystem, Data Management subsystem, Manage Scraping subsystem, View Dashboard subsystem, Search Grants subsystem, View Grants subsystem. To interact within these subsystems there are 4 types of actors who are Research Head, Research Manager, Research Officer and Researcher respectively. The use case of the system is given below for more information.

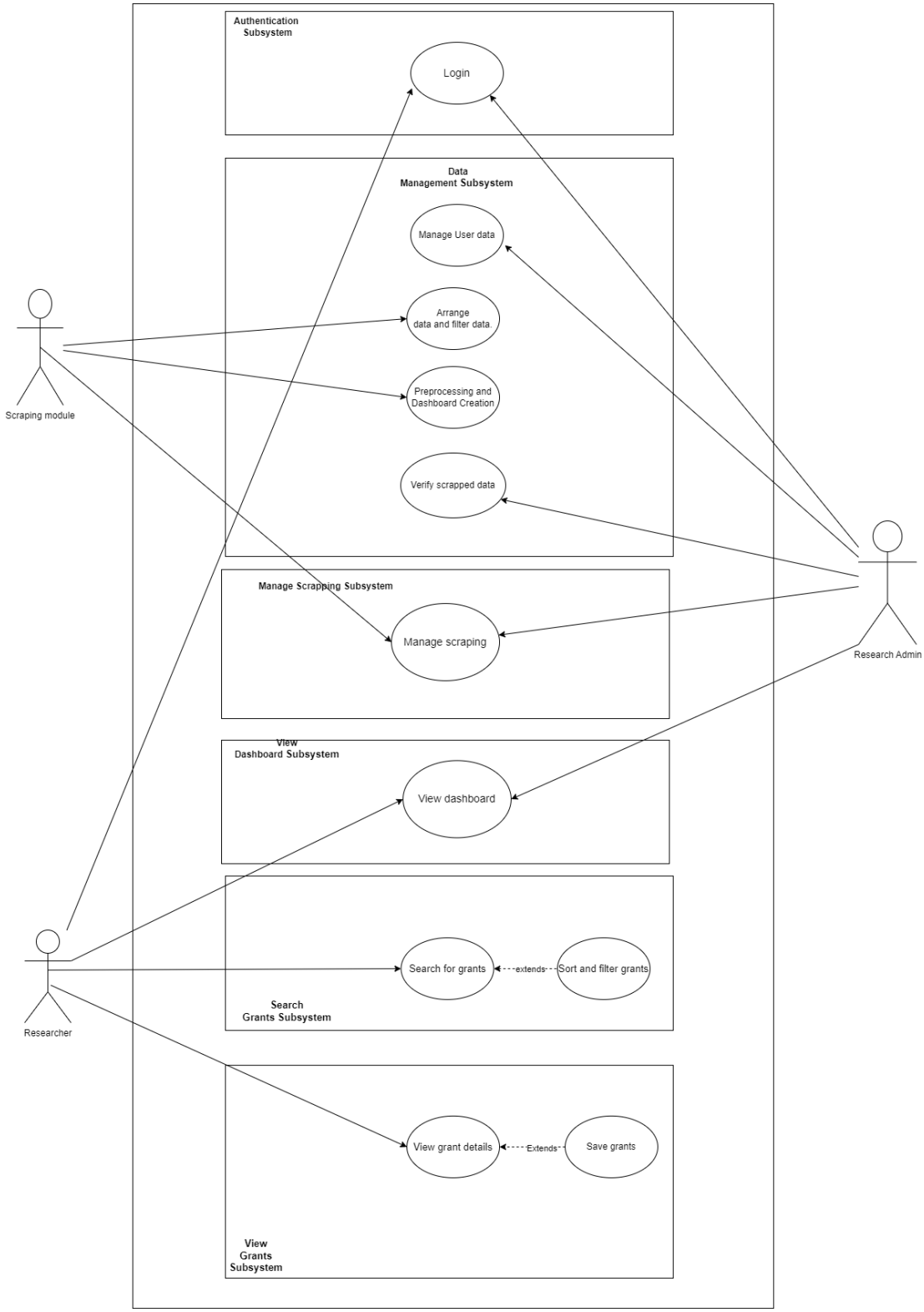


Figure 2.1: Use Case Diagram of <Name of the System>

2.1 Product Perspective

The system this SRS describes is the Research Grant Finder system. Most of the universities are sustaining because of their research works. A university is ranked on the availability of research and fund success rates. So every universities are in a competition for doing research

works. But there are many research works which cannot be done because of financial problems. So they have to take grants from different organizations. Bu they have to face a lot of problems for applying and looking for grants. The same problems are faced by the researchers of Universiti Teknologi Malaysia. The Research Grant Finder will extract the grant information from different grant providing websites, sort them according to different categories, preprocess the data, store it in a database and then create a dashboard of grants using data visualization. The researcher can view the grants analytics such as number of grants received according to respective funding organization, grants received by respective researcher, grants received according to different fields and top grant providing organizations. These data will help the researchers to choose grants. They can also search and save grants that they want to apply for through this system. Overall, this system will be beneficial to the university and researchers and improve their research works.

2.1.1 System Interfaces

The Research Grant Finder system is a web-based application that will use web scraping technology and data visualization techniques to display the data in an understandable manner. The front end will be developed using ReactJs and for the backend Django framework of Python will be used. For the connection between front end and backend, we will use MongoDB database language.

2.1.2 User Interfaces

Describe how the system will interact with its users.

The Research Finder will have interfaces so that the system is user-friendly and easy to use. There are separate interfaces in the system for different functions and actors. These are described below:

2.1.2.1 Login Interface

This interface is used to access the functionalities of the system. This interface identifies the type of user by reading the data entered by the users.

2.1.2.2 Admin Interface

This interface is only for the admins to work on the maintenance and performance of the system. The admins can modify the user data, make certain changes, modify the data of the system and manage the system properly through this interface.

2.1.2.3 Dashboard Interface

This is the final interface of the system. It will contain all the data that are extracted by the system and present them in an understandable and attractive format.

2.1.2.4 Search Grants Interface

This interface enables the user to search for their specific grants using keywords from the database where the extracted data is stored.

2.1.3 Hardware Interfaces

This system can be used from any desktop computers or laptops with the help of a web browser and a stable internet connection.

2.1.4 Software Interfaces

To run this system, it is needed to have an operating system notably Windows, MacOs and a web browser like Google Chrome, Microsoft Edge, Safari that are capable of running this system.

2.1.5 Communication Interfaces

Communication in this system is carried out over the internet makes use of dependable TCP/IP protocols like HTTP and FTP in order to ensure the highest level of compatibility and dependability possible.

2.1.6 Memory

Any computer having the standard memory configurations will be able to run this system.

2.1.7 Operations

The system consists of many operations. There are 3 admins with respective roles to conduct these operations properly. The initial login credentials of the user will be provided by the admin. The user needs to login using those credentials. The dashboard can be viewed by anyone, no login is needed for it. But to access the grant information, it is needed to login to the system.

2.1.8 Site Adaptation Requirements

This system will run on any web browser that the operating system of a computer supports. It does not need any platform-specific requirements.

2.2 Product Functions

The product functions are described in the form of table below.

Module	**Use cases**	**Description**
Authentication	Login	Allows the users to access the system by entering their valid details.
Data Management	Manage User Data	Allows the admin to insert, create and deke the user information or delete them.
	View the Extracted Data	This use case allows the admin to view the raw data that has been found by scraping the target websites.
	Arrange and filter data	Allows the research heads and manager to reorganize the extracted data, filter it according to the necessary categories ant make it ready for preprocessing.
	Data Preprocessing & Dashboard Creation	Allows user(admin) to process the data and then upload it to the database for visual representation in the form of a dashboard.
Manage Scraping	Manage Scraping	Allows research admin to manage the web scraping process by letting him/her modify the different parameters that control the scrapping.

View Dashboard	View Dashboard	Allows all the actors in the system to view the final product which contains the visualization of all the extracted data.
Search Grants	Search for grants	Allows the user to search for grant using specific keywords.
	Filter and sort grants	Allows to narrow the search by selecting specific categories form the data.
View Grants	View Grant Details	Allows the user to view each grants information in details.
	Save Grants	Allows user to save grants and integrate their data with their profile.

2.3 User Characteristics

The different types of users involved in this system are described in the table below

User	Characteristics
Research Admin	Oversees the system functionalities, create user, delete users, manage scrapping, view dashboard and other maintenance tasks
Scrapping Module	Clean the data, process the data into a file.
Researcher/ End user	Views the dashboard, access the data, search for grants, view and save them.

2.4 Constraints

The constraints for the system are:

1. The system should be able to be accessed from different popular web brosers.

2. The system should have an effective communication channel for data transfer.
3. The system should be responsive and easy to use.
4. The data extraction from the target websites should be done effectively.
5. The system should use a language that is understandable by all.

2.5 Assumption and Dependencies

The system may not function properly if it is not run on suitable web-browser or does not have a stable internet connection or if it is run on a smartphone with different screen resolution. The admin also needs to update the system database to keep the system updated and regulate the scraping of data according to different situations.
The system can adjust to new developments, but it may affect the requirements and change some functionalities of the system.

2.6 Apportioning of Requirements

If this project's development is delayed, some needs may be carried over to the next edition of the software. These are some of the needs that will be addressed in the second release. The SDD of the system will contain more details of the functionalities.

3. Specific Requirements

This section discusses both the system's functional and non-functional requirements. This section also covers the description of use case specifications, as well as sequence and activity diagrams for each use case.

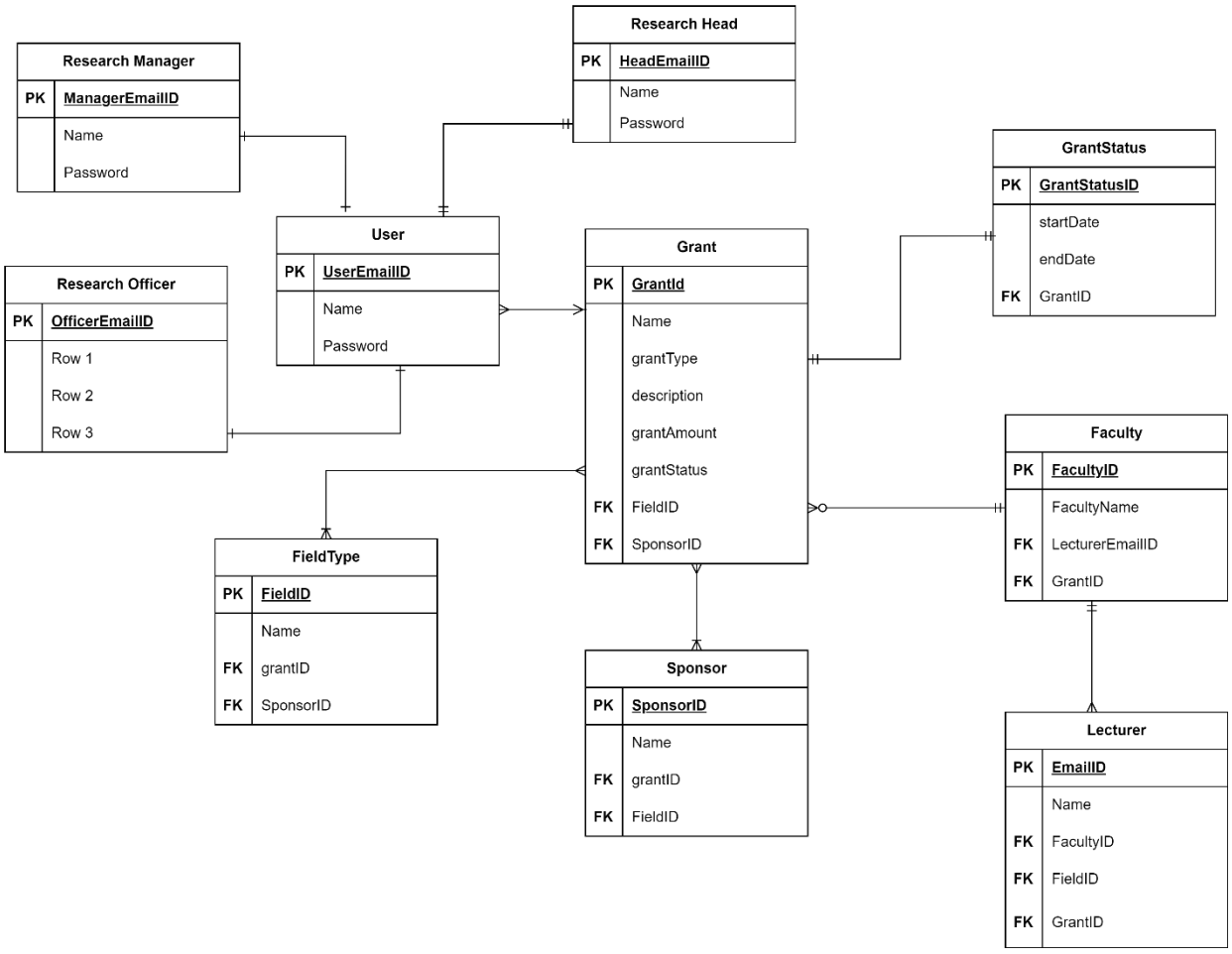


Figure 3.1: Domain Model of < Research Grant Finder System>

3.1 External Interface Requirements

3.1.1 User Interfaces

3.1.1.1 Login Interfaces

Through the login interface al the users get access tp the main features of the system. The admins after login can perform their activities in managing the system. The end users also need to login in order to save grants. The users need to enter valid email and password that is already present in the database. Otherwise, they cannot pass this interface.

3.1.1.2 Admin Interfaces

Through the admin interface the admins manage the system. The research head manages the scraping, confirms changes of the data and oversees the whole system through this interface. The research manager and officer also clean and modify the data for visualization through this interface.

3.1.1.3 Dashboard Interface

Through this interface all the users of this system can view the data that is visualized after being extracted from the websites. The data can also be seen in detail from the dashboard.

3.1.1.4 Search Grants Interface

Through this interface the end users/ researchers can search for their grants by inputting the keywords related to their research. He can then view the grants and see their details and also save them so that they don't lose it in future.

3.1.2 Hardware Interfaces

To use this type of system, one must have a web browser installed on a laptop or a desktop. The system is only accessible with a stable internet connection. Unstable internet connections or web browser faults can cause system failures.

3.1.3 Software Interfaces

To access this system, users must have a web browser and an internet connection in order for it to function effectively. This system can be accessed using a variety of web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, Apple Safari, and Opera. To get the most out of the computer system, it is critical to keep the web browser up to date.

3.1.4 Communication Interfaces

This system allows users to log in using a web browser connected to a local area network (LAN) in order to maintain consistent communication between its many components. Furthermore, trustworthy TCP/IP protocols such as HTTP and FTP are used for internet-based communication to ensure the highest level of compatibility and dependability.

3.2 System Features

3.2.1 Module <Authentication>

State briefly the functional requirements (use cases) that are available in this module. Better to include the diagram of the specific module (or the example of Customer Support System – by subsystem, see example below) from the overall use case diagram in Figure 2.1.
The functional requirements of this module are:

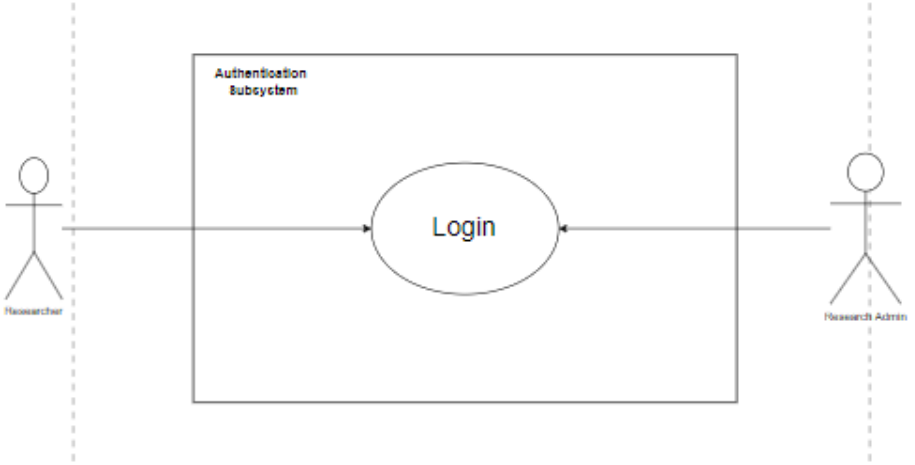


Figure 3.3: <Authentication>

3.2.1.1 UC001: Use Case <Login>

Table 3.1: Use Case Description for <Login>

Use Case ID	UC
Use Case Name	Login
Description	This use case allows the user to sign-in to the system. This use case differentiates a normal user and the admin.
Actor(s)	Researcher, research head, research manager, research officer.
Pre-conditions	User has an account in the system and has a stable internet connection.
Normal Flow	1. User enters the login credentials. 2. The user submits the login. EF1 will be executed. 3. The system validates the login credentials by matching them with the database. AF1 will be executed.

	4. If the data matches with the admin database, then the system redirect to admin page. 5. If the data matches the user, the system redirect to grant search page.
Alternative Flow	1. If the user data is not found, view error message. Else continue NF4.
Exception Flow	1. The input field is kept empty. 1.1. System displays error message.
Post Conditions	
Related Requirements	

Include system sequence diagram and activity diagram for each respective use case. See example below for Telephone Order scenario of Create New Order use case for both diagrams. Consider including different scenarios example telephone vs. Web order scenario in different diagrams if applicable.

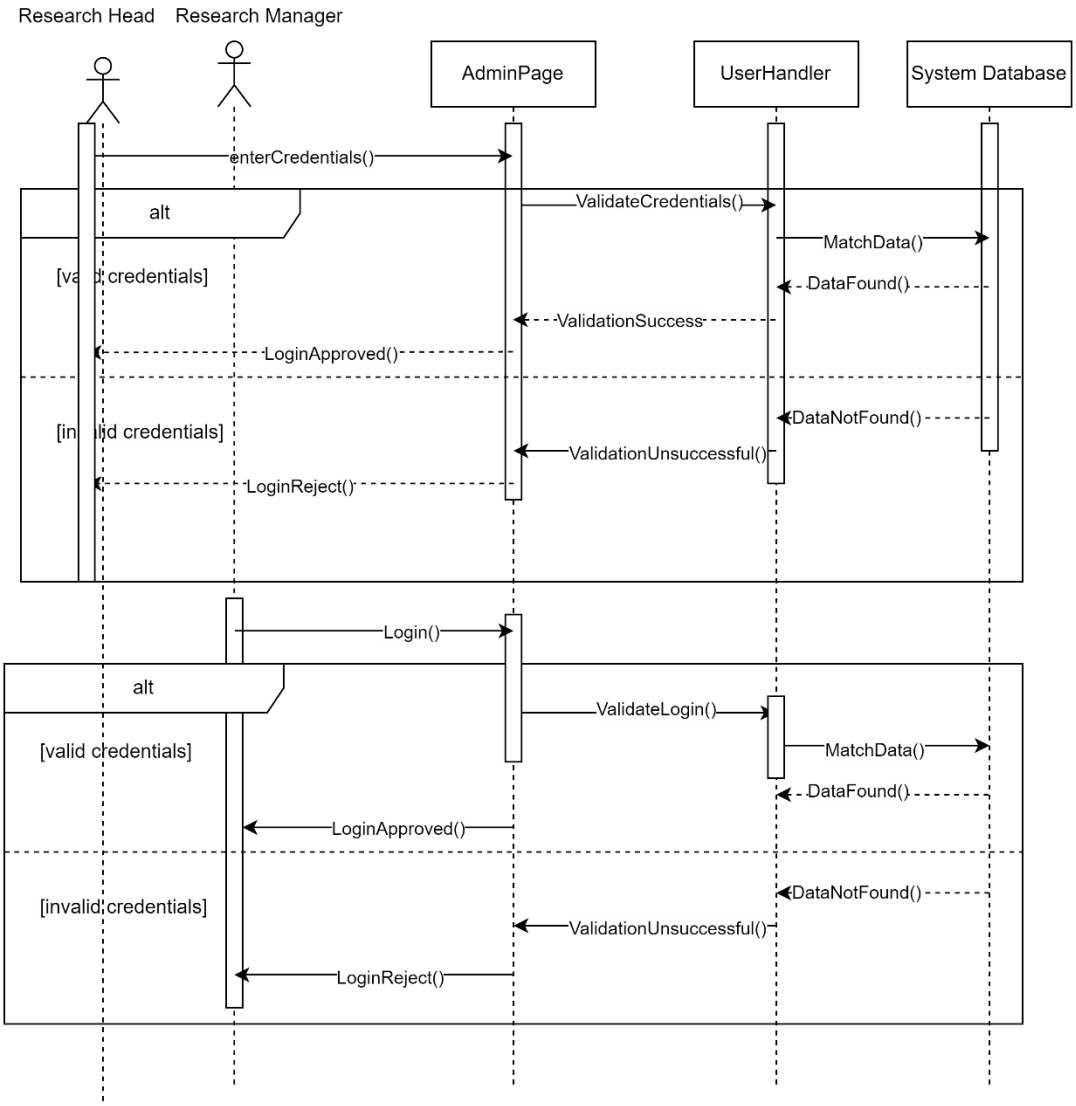


Figure 3.4: System Sequence Diagram of <Login>

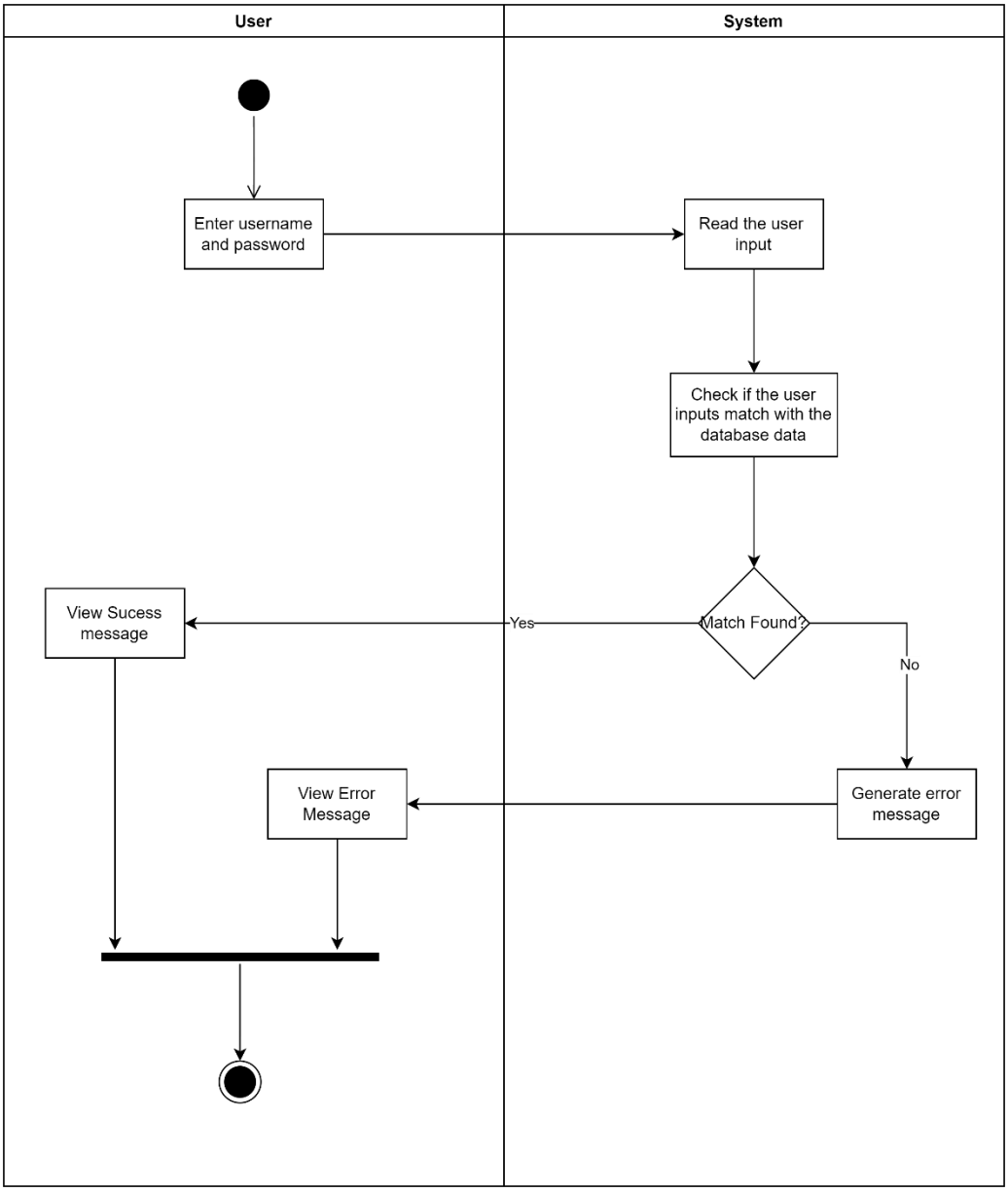


Figure 3.5: Activity Diagram of <Login>

3.2.2 Module <Data Management>

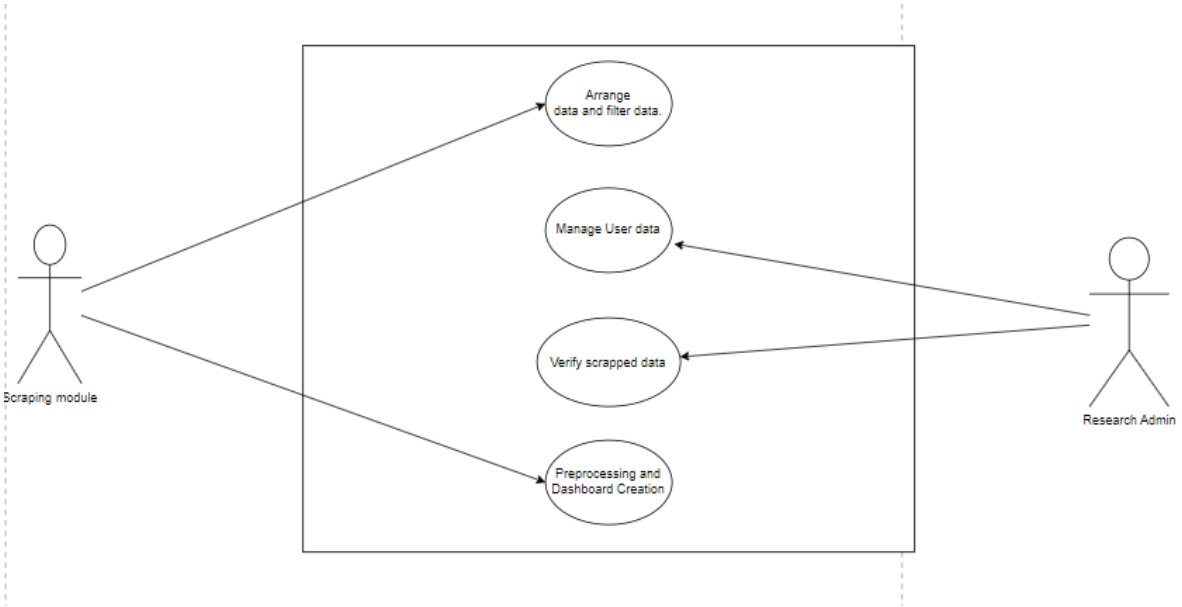


Figure 3.6: < Data Management >

The functional requirements of this module are:

3.2.2.1 UC002: Use Case <Manage User Data>

Table 3.2: Use Case Description for < Manage User Data >

Use Case ID	UC
Use Case Name	Manage User Data
Description	This use case allows the admins to insert, modify and update the user login information and their saved grants information.
Actor(s)	Research Head, Research Manager
Pre-conditions	The actors must be logged in to the admin page.
Normal Flow	1. Insert user data. 1.1. The admin inserts the user credentials that the user will use to login to the system.

	1.2. The admin does not fill the input fields, then EF1 will be performed. 1.3. The system stores the data in the user database. 2. Request user data 2.1. The admin requests to view the saved user data. 2.2. The system returns the data from the database. 2.3. The admin can view the data. 3. Modify user data. 3.1. The admin can edit the username and password of the users. 3.2. The admin waits for approval. AF1 will be executed. 4. Save changes. 4.1. Review the data. 4.2. The data is not properly changed. EF2 will be performed. 4.3. The admin saves the changes. 4.4. The modified data is stored in the database.
Alternative Flow	1. If the admin is research Manager, stop at NF3
Exception Flow	1. The admin keeps empty field. 1.1. The system shows error. Start NF1. 2. The admin clicks on discard changes. 2.1. The system shows error message. Continue NF4.
Post Conditions	The modified user data is stored in the system.
Related Requirements	

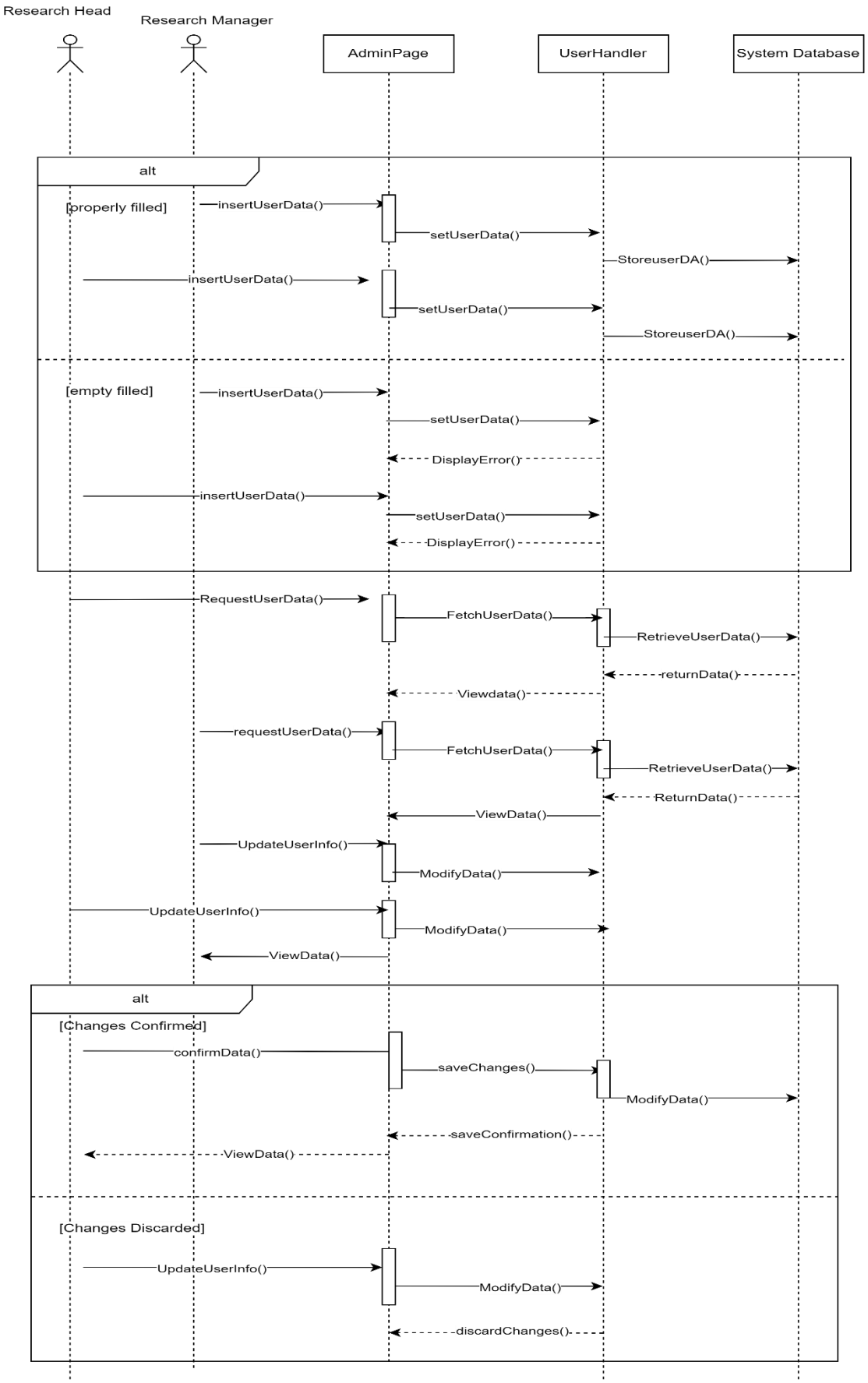


Fig 3.7: Sequence Diagram <Manage User>

Manage User Data

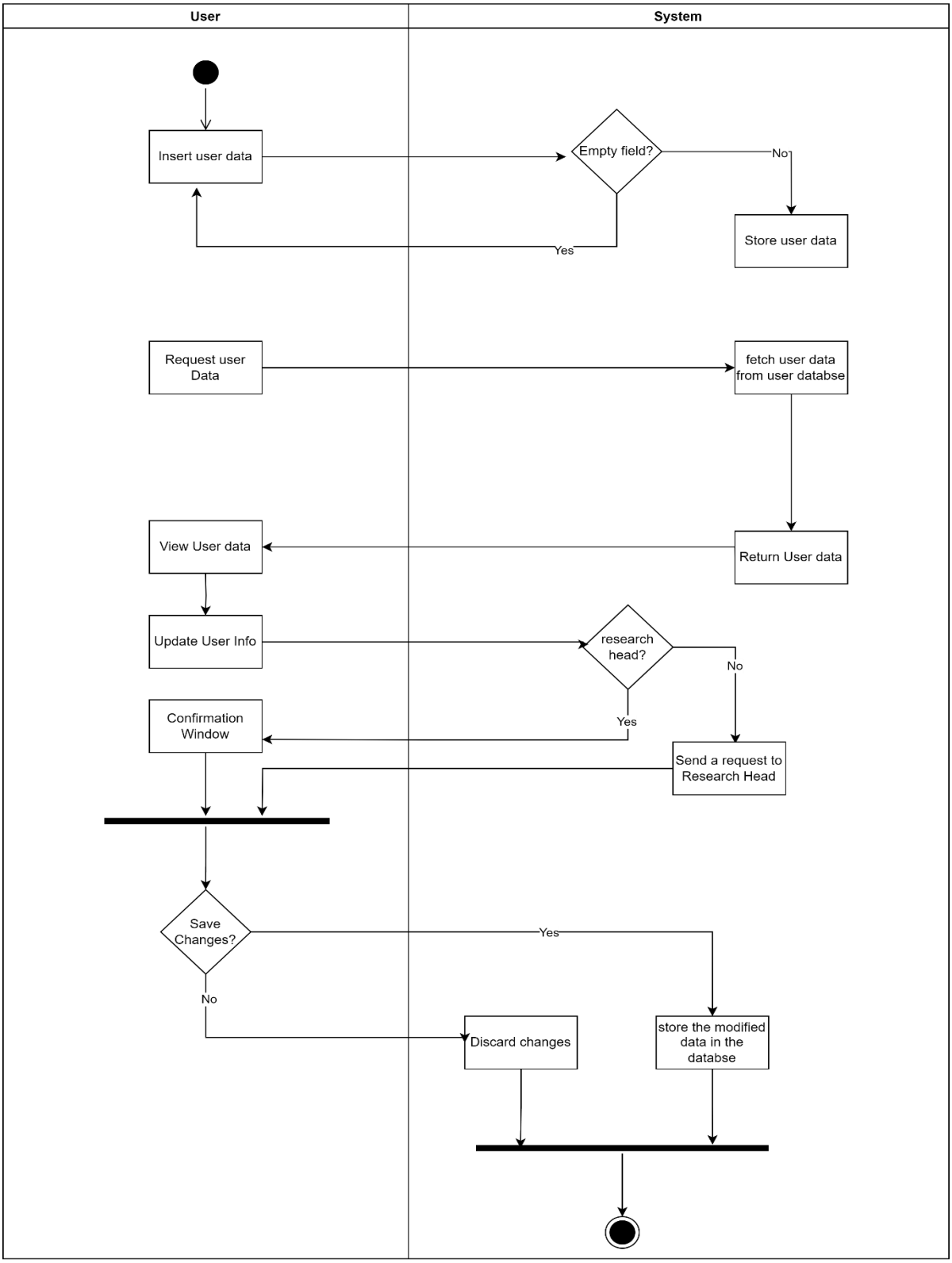


Figure 3.8: < Manage User Data >

3.2.2.2 UC003: Use Case <Arrange data and filter data.>

Table 3.5: Use Case Description for < Arrange data and filter data.>

Use Case ID	UC
Use Case Name	**Arrange data and filter data.**
Description	The admin loads the data from the scraping module and then arranges them with respect to different data fields. The data are sorted and made ready for preprocessing.
Actor(s)	Research admin, scrapping module
Pre-conditions	1. The admin is logged in to the system. 2. Data exists in the scraping module.
Normal Flow	1. Admin requests the data. 2. The system loads the data from the module. 3. The admin arranges the data to combine the grant information. 3.1. Identify data of funding organization. 3.2. Identify data of research fields. 3.3. Identify grant revenue. 3.4. Identify ongoing grants. AF1 will be executed. 4. Upload to the dashboard preprocessor for further processing.
Alternative Flow	If the admin is research head, go to NF4.
Exception Flow	

Post Conditions	The extracted data are organized and sorted according to different categories.
Related Requirements	

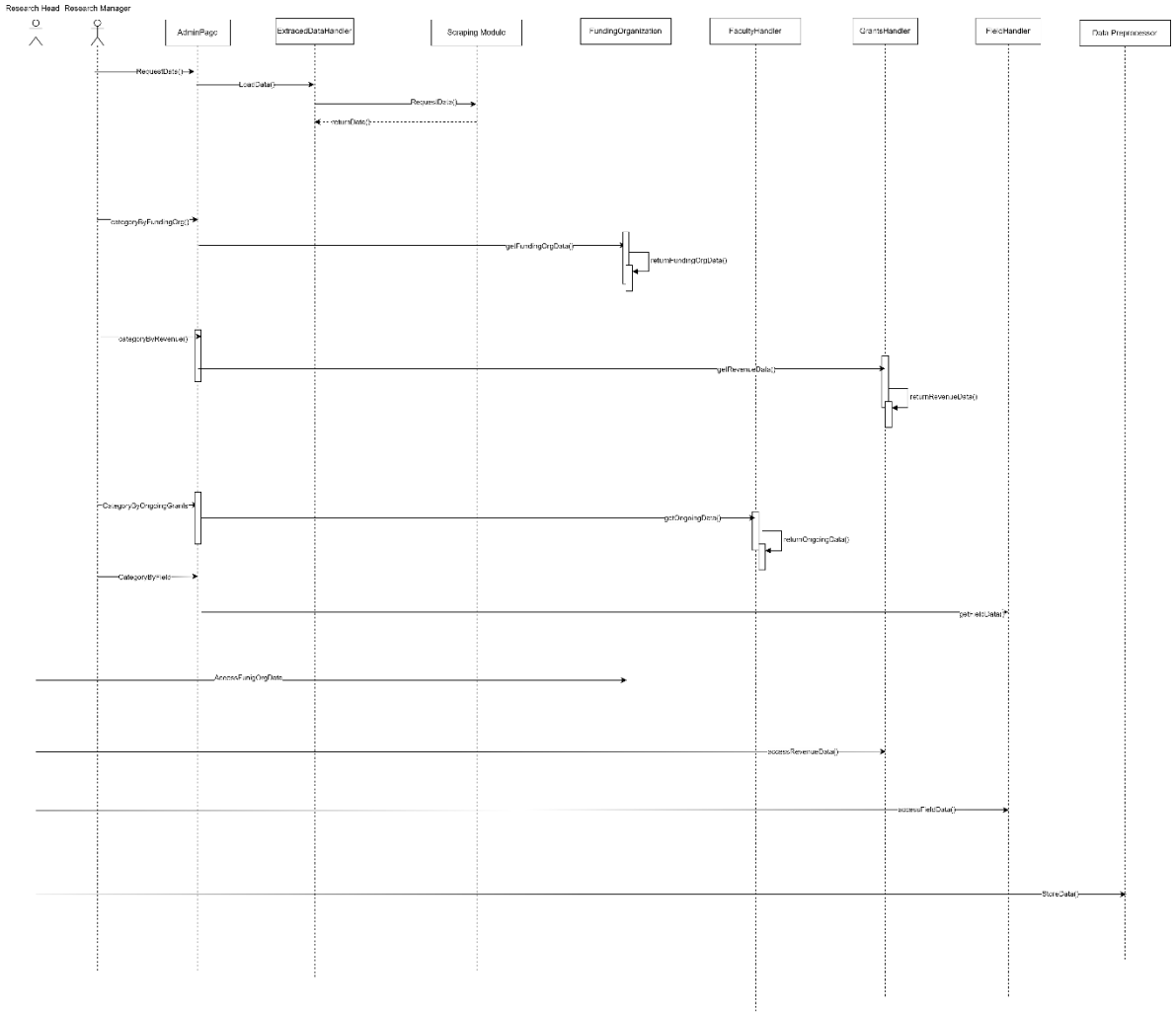


Figure 3.10: Sequence Diagram < Arrange data and filter data >

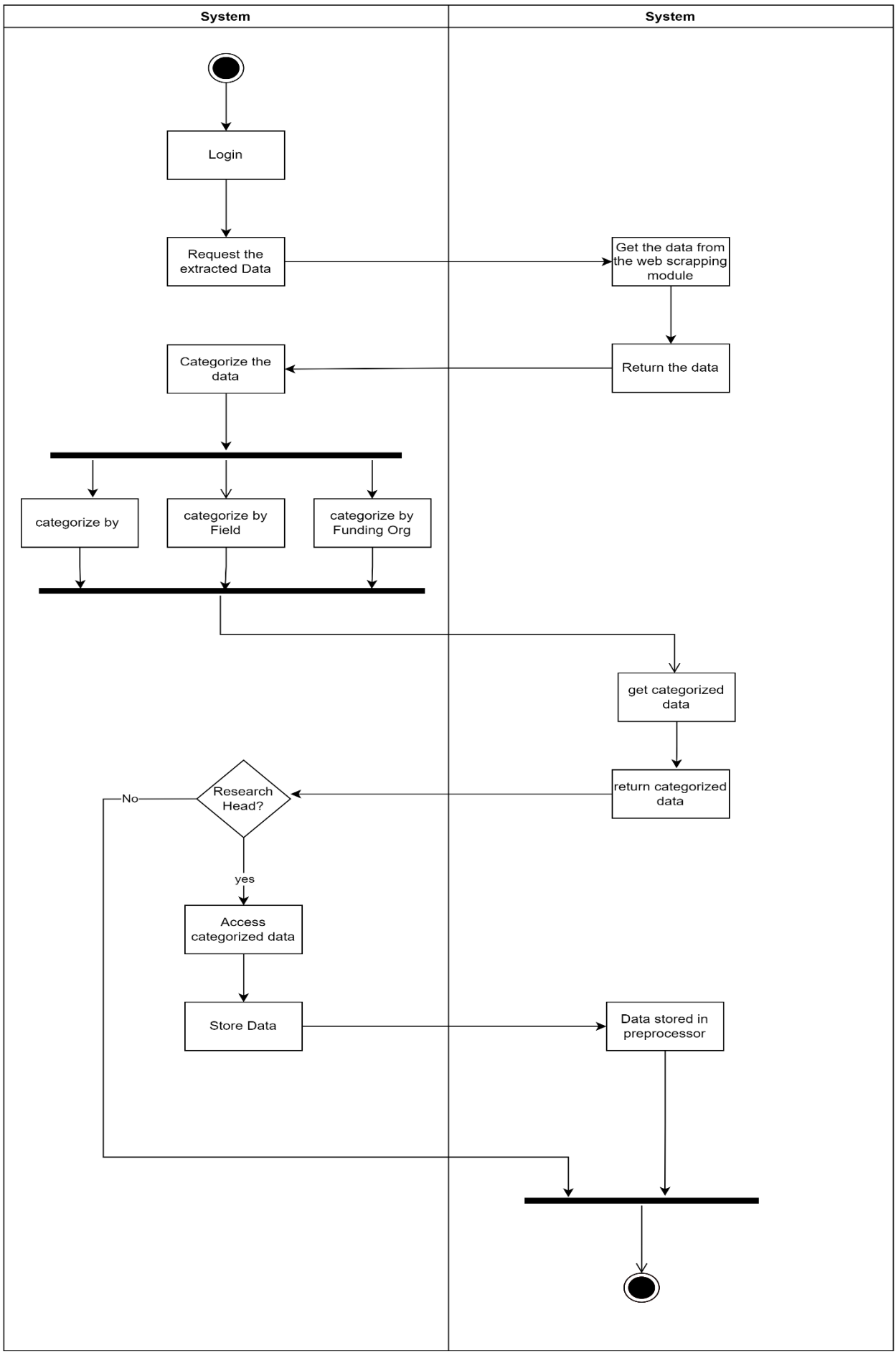

System
System
Login
Request the extracted Data
Get the data from the web scrapping module
Return the data
Categorize the data
categorize by
categorize by Field
categorize by Funding Org
get categorized data
return categorized data
Research Head?
No
yes
Access categorized data
Store Data
Data stored in preprocessor

Figure 3.11: Activity Diagram < Arrange data and filter data >

3.2.2.3 UC004: Use Case <Data Preprocessing and Dashboard Creation>

Table 3.4: Use Case Description for < Data Preprocessing and Dashboard Creation >

Use Case ID	UC
Use Case Name	**Data Preprocessing and Dashboard Creation**
Description	The scrapping module processes the data for visualization
Actor(s)	Scrapping Module, Research Admin
Pre-conditions	The actors must be logged in to the system.
Normal Flow	1. View the extracted data. 1.1. System looks for extracted data in the web scraping module. 1.2. System retrieves the data for preprocessing. AF1 will be executed. 1.3. System does not find any data, EF1 will be executed. 2. Clean the data. 2.1. The admin cleans the data that is extracted. 2.2. The admin removes the data that is not needed for the system. 2.3. Separate dependent and independent variables. 2.4. Transform data. 3. Modify the data. 3.1. Admin changes the data that is not understandable into an easier form. 3.2. Connect data with the independent variables. AF2 will be executed.

	4. Confirm the changes. 4.1. The admin views the changed data. 4.2. The data is not proper. EF2 will be performed. 4.3. The admin confirms the data and clicks on the save button. 4.4. The data is uploaded to the database.
Alternative Flow	1. If the admin is the research manager go to NF3. 2. If the admin is not Research Head, stop at NF3
Exception Flow	1. The system shows an error message, and the process ends. 2. Admin clicks on discard changes. 2.1. System displays error message. 2.2. Data not saved in database.
Post Conditions	The modified preprocessed data is stored in the system database.
Related Requirements	

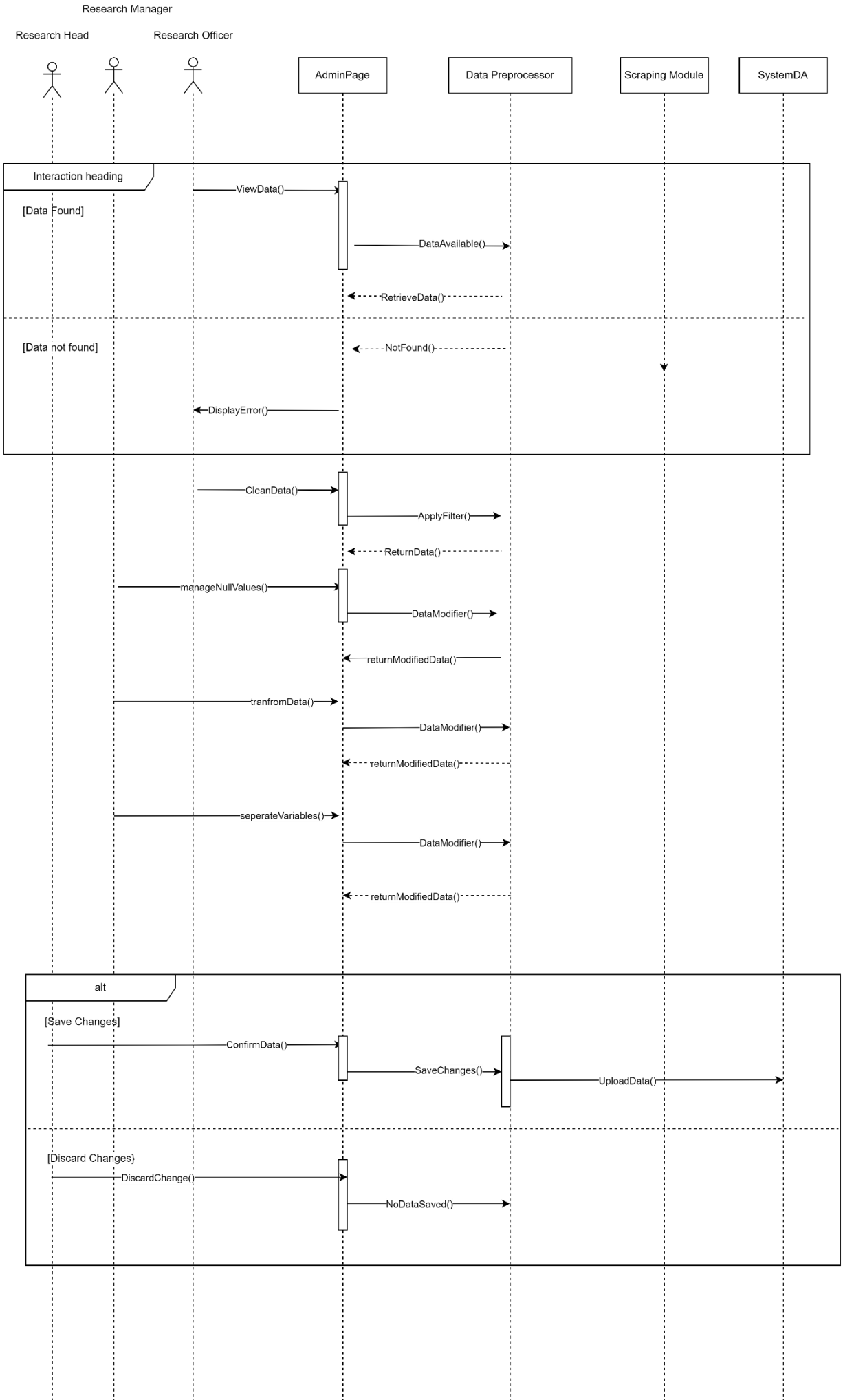

Research Manager
Research Head
Research Officer
AdminPage
Data Preprocessor
Scraping Module
SystemDA
Interaction heading
ViewData()
[Data Found]
DataAvailable()
RetrieveData()
[Data not found]
NotFound()
DisplayError()
CleanData()
ApplyFilter()
ReturnData()
manageNullValues()
DataModifier()
returnModifiedData()
tranfromData()
DataModifier()
returnModifiedData()
seperateVariables()
DataModifier()
returnModifiedData()
alt
[Save Changes]
ConfirmData()
SaveChanges()
UploadData()
[Discard Changes]
DiscardChange()
NoDataSaved()

Figure 3.12: Sequence Diagram < Dashboard Creation and Processing >

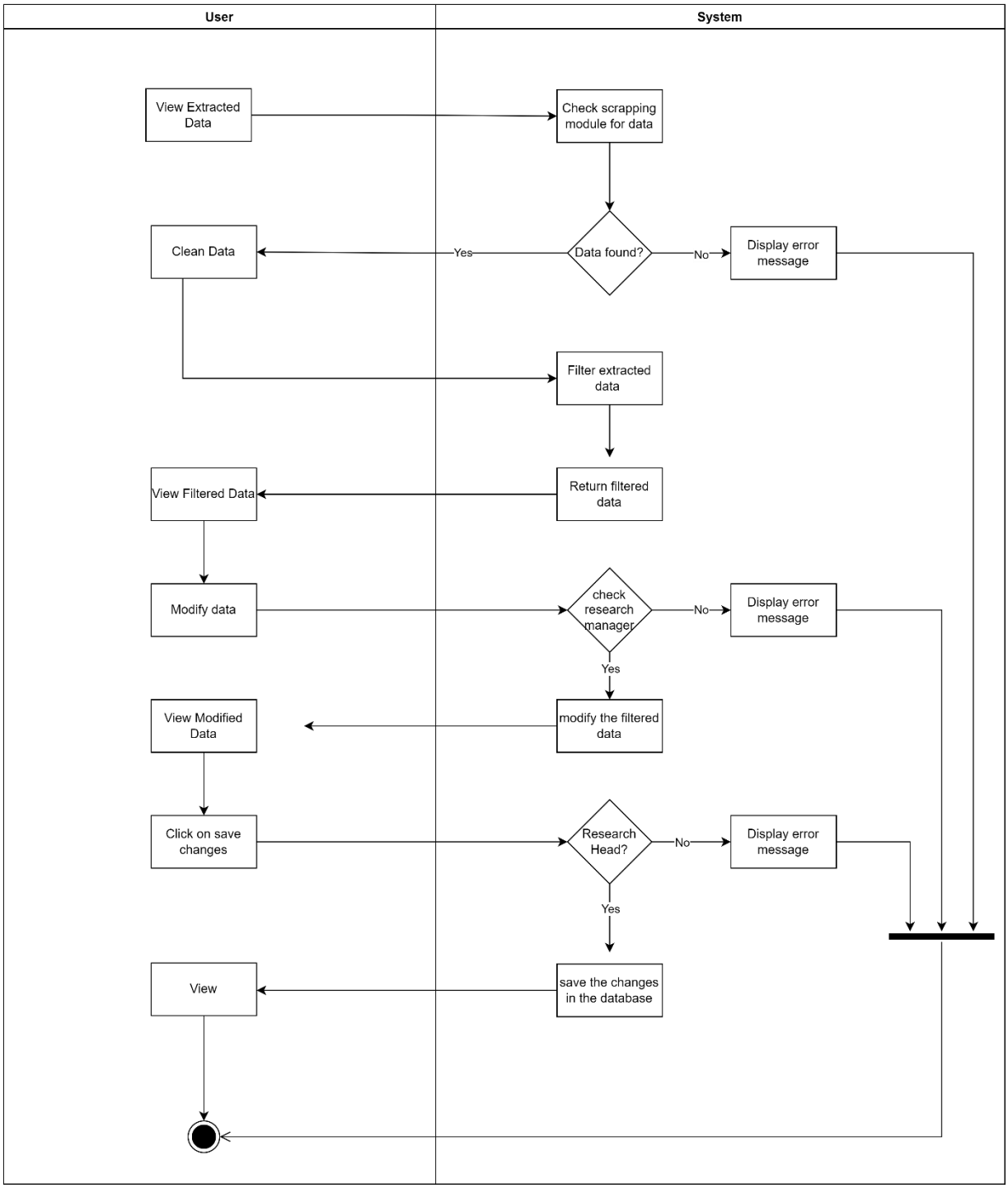


Figure 3.13: Activity Diagram < Dashboard Creation and Processing >

3.2.3 Module <Manage Scraping>

3.2.3.1 UC005: Use Case <Manage Scraping>

Table 3.5: Use Case Description for <Manage Scraping>

Use Case ID	UC
Use Case Name	Manage Scraping
Description	In this use case the research head can control the web scraping of the grants data.
Actor(s)	Research Admin
Pre-conditions	The Research Admin must be logged in to the system.
Normal Flow	1. The admin clicks on the manage scraping button. 2. The admin chooses options to manage. 3. The admin clicks on modify scrape frequency. 3.1. The admin changes the scraping frequency of the algorithm. 3.2. The admin saves the changes.EF1 will be executed. 4. The admin clicks on modify target URL. 4.1. The admin changes the URL target websites for scraping. 4.2. The admin saves the changes. EF1 will be executed. 5. The admin clicks on modify configuration files. 5.1. The admin modifies the configuration files. 5.2. The admin saves the changes. EF1 will be executed 6. The admin clicks on view scrape logs. 7. The admin saves the changes.

Alternative Flow	
Exception Flow	1. If the admin clicks on save changes but the changes are not saved, go to NF1.
Post Conditions	
Related Requirements	

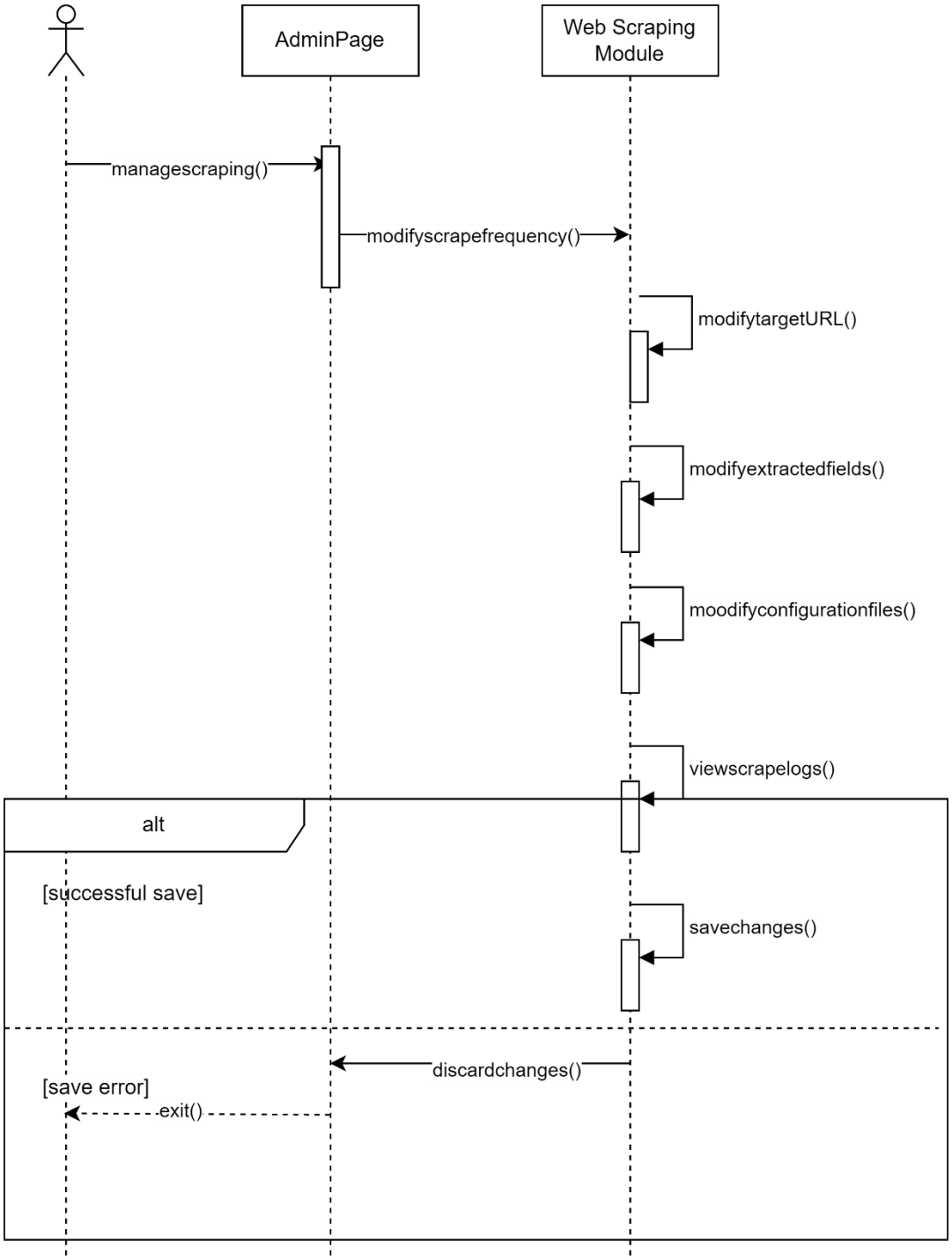


Figure 3.14: Sequence Diagram < Manage Scraping >

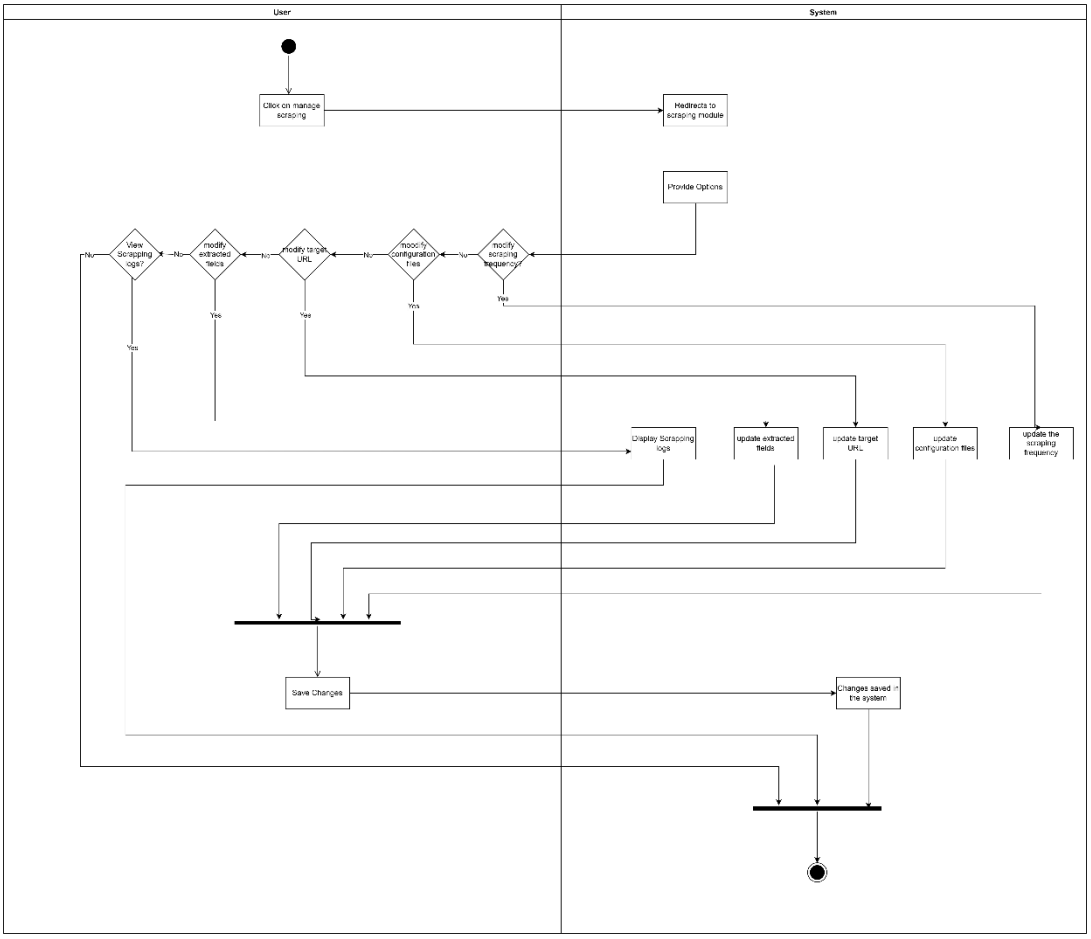


Figure 3.15: Activity Diagram < Manage Scraping >

3.2.4 Module <View Dashboard>

3.2.4.1 UC006: Use Case <View Dashboard>

Table 3.5: Use Case Description for < View Dashboard >

Use Case ID	UC
Use Case Name	**View Dashboard**
Description	This use case allows the users to view the dashboard.
Actor(s)	Researcher, research admin

Pre-conditions	The data is extracted, and the grant data is stored in the database.
Normal Flow	1. When the user enters the system, the system requests info of the dashboard. 2. The system requests info from the database in the form of different categories with respect to which the data has been uploaded to the database for visualization. 3. The system loads the dashboard. Execute NF2
Alternative Flow	1. If the system fails to load dashboard, go to NF2.
Exception Flow	-
Post Conditions	The user can look for the grant analytics and search for grants.
Related Requirements	

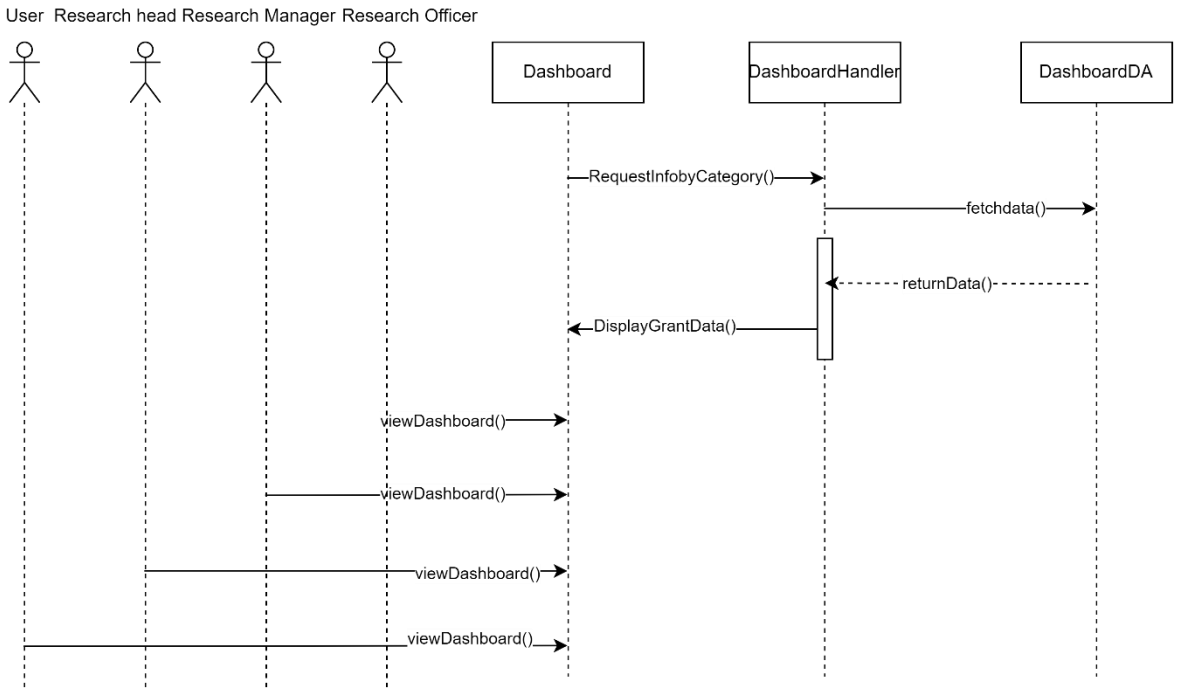


Figure 3.16: Sequence Diagram < View dashboard >

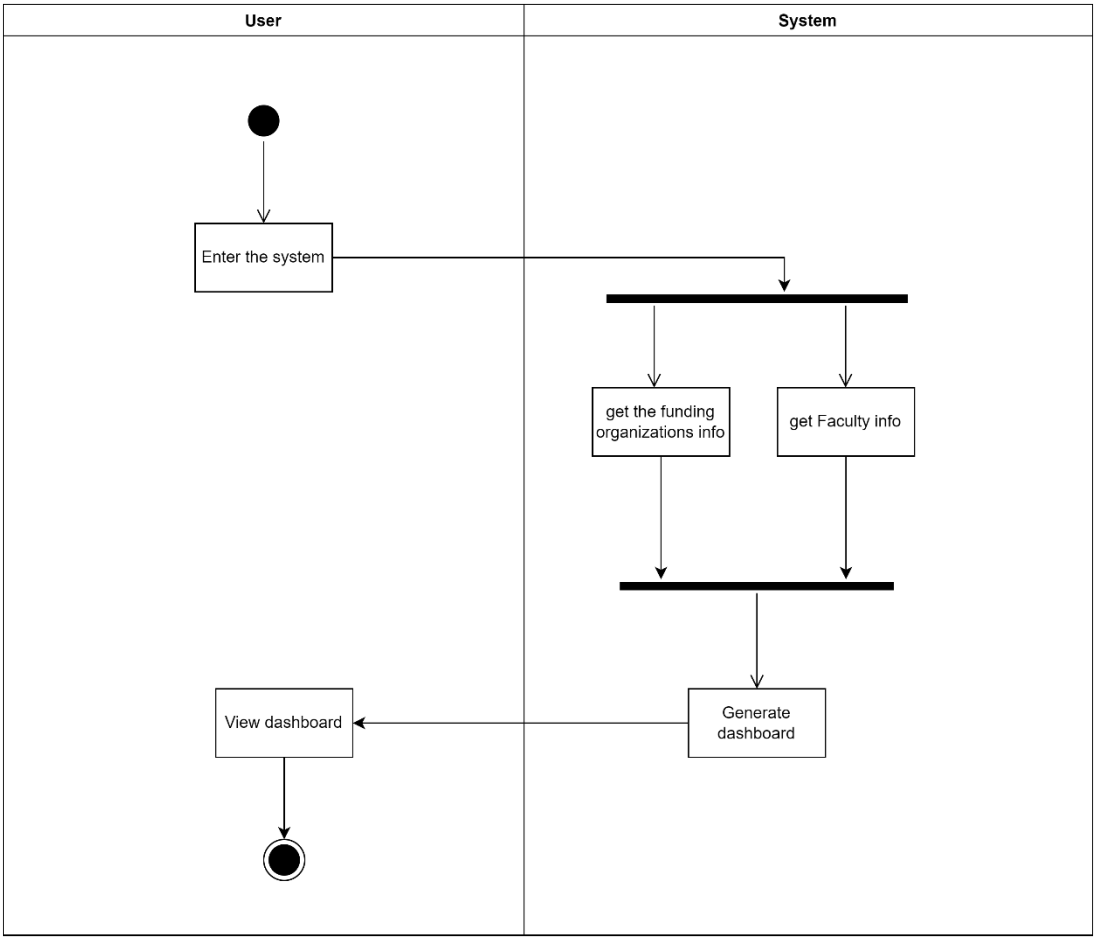


Figure 3.17: Activity Diagram < View dashboard >

3.2.5 Module <Search Grants>

3.2.5.1 UC007: Use Case <Search for Grants>

Table 3.5: Use Case Description for < Search for Grants >

Use Case ID	UC
Use Case Name	**Search for Grants**
Description	This use case allows the users to search for grants database.
Actor(s)	Researcher, research admin.
Pre-conditions	1. Grants data stored in the system. 2. The dashboard is ready to be viewed.

Normal Flow	1. The user enters the desired keyword. 2. The system fetches the keyword. 3. The system matches the keyword with the existing grants data in the system database. 4. The system loads the grants that match with the keywords. AF1 will be performed.
Alternative Flow	1. If the system does not find any matched data, it displays error. Continue from NF1.
Exception Flow	
Post Conditions	The user can view the grants list according to his keywords.
Related Requirements	

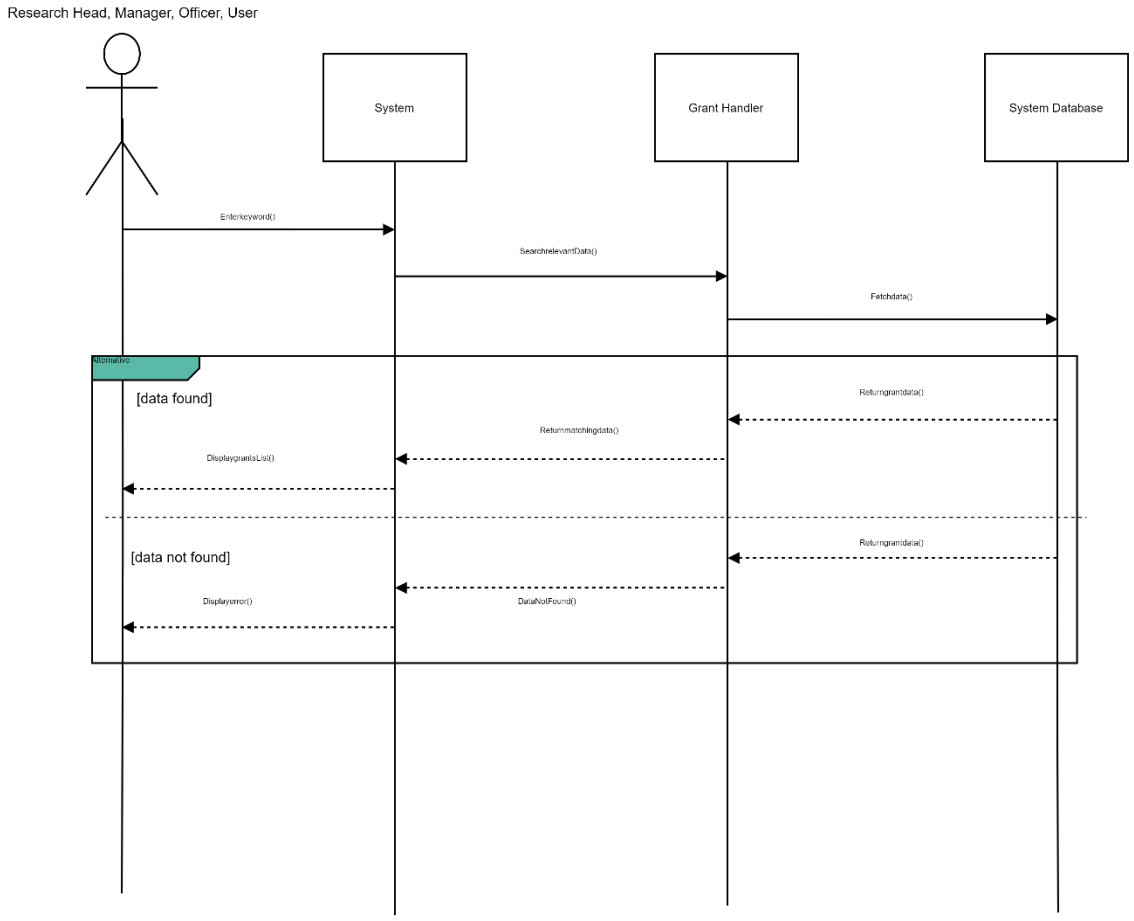


Figure 3.18: Sequence Diagram < Search for Grants >

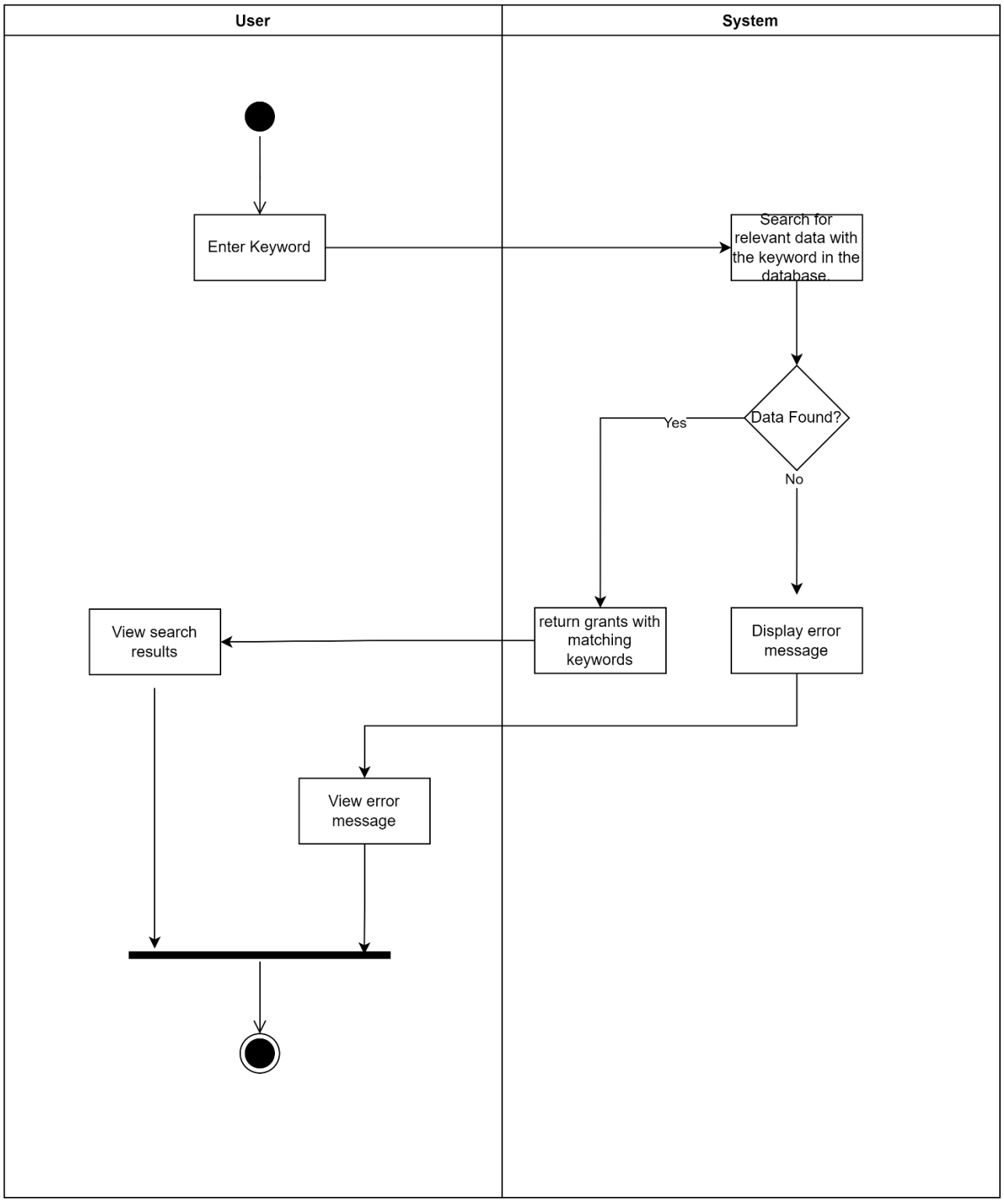


Figure 3.19: Activity Diagram < Search for Grants >

3.2.5.2 UC008: Use Case <Sort and Filter Grants>

Table 3.5: Use Case Description for < Sort and Filter Grants >

Use Case ID	UC
Use Case Name	**Sort and Filter Grants**
Description	This use case allows its user to filter the data according to different categories and also sort the data.
Actor(s)	Users
Pre-conditions	1. The database contains data. 2. The dashboard can be viewed.
Normal Flow	1. The user chooses the different filters. 2. The user chooses filter by different funding Organization. 2.1. The data is filtered according to the funding organization the user chooses. 3. The user chooses filtering according to the field of research. 3.1. Data is filtered according to the field selected by the user. 4. The user chooses filtering by deadline. 4.1. Data is filtered according to the grants deadline. 5. User clicks on search button. 6. The filtered data is displayed. 7. The user chooses sorting by name. 7.1. Data is sorted by the name of the grants in alphabetical order. 8. The user chooses sorting by Year. 8.1. Data is sorted by year of the grants.

	9. The user chooses sorting by field of research. 9.1. Data is sorted by fields of the grants in an order.
Alternative Flow	-
Exception Flow	-
Post Conditions	The sorted and filtered data will be visible to the user.
Related Requirements	

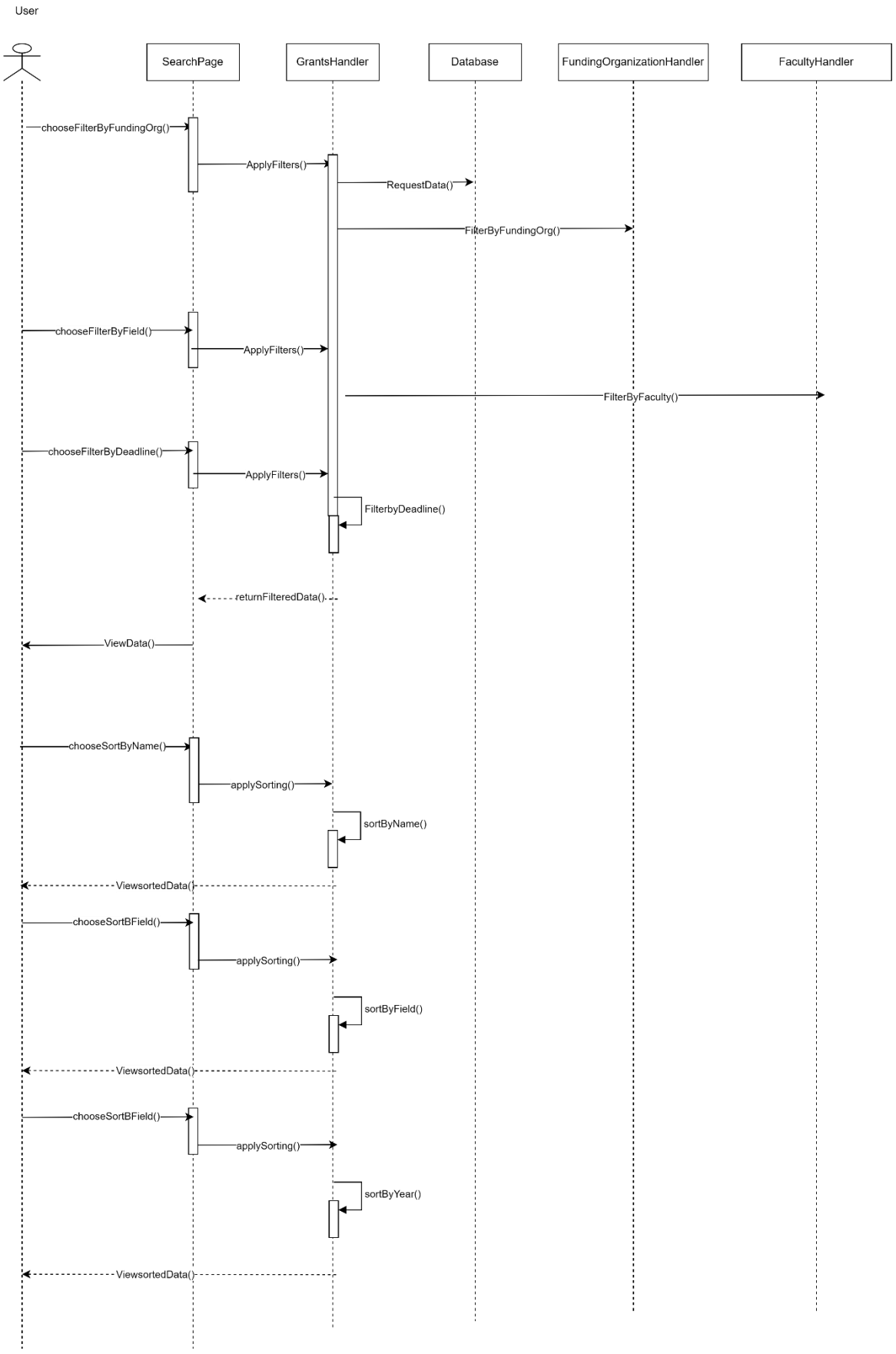


Figure 3.20: Sequence Diagram < Sort and Filter Grants >

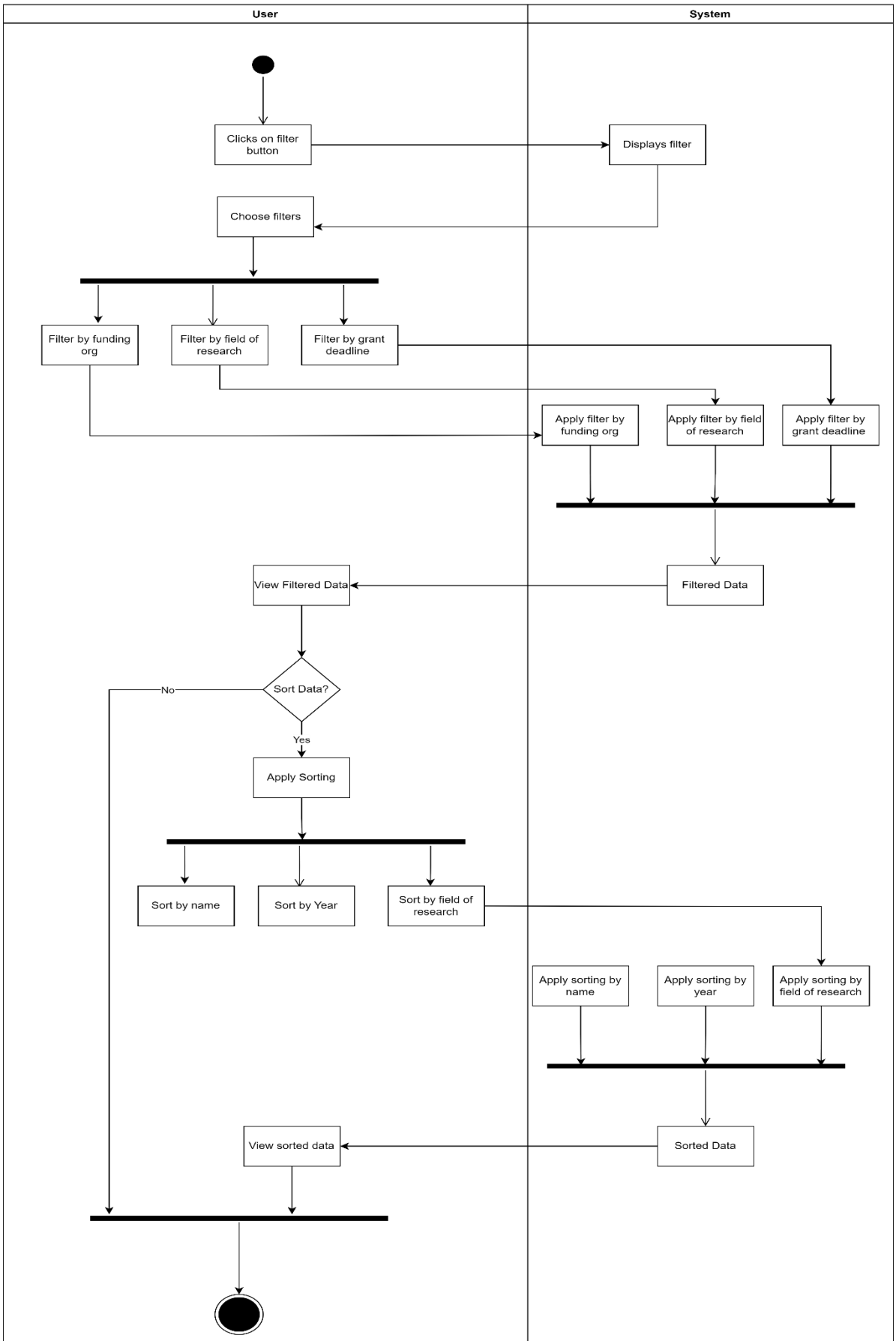

User
System
Clicks on filter button
Displays filter
Choose filters
Filter by funding org
Filter by field of research
Filter by grant deadline
Apply filter by funding org
Apply filter by field of research
Apply filter by grant deadline
Filtered Data
View Filtered Data
Sort Data?
No
Yes
Apply Sorting
Sort by name
Sort by Year
Sort by field of research
Apply sorting by name
Apply sorting by year
Apply sorting by field of research
Sorted Data
View sorted data

Figure 3.21: Activity Diagram < Sort and Filter Grants >

3.2.6 Module <View Grants>

3.2.6.1 UC009: Use Case <View Grant Details>

Table 3.5: Use Case Description for < View Grant Details >

Use Case ID	UC
Use Case Name	**View Grant Details**
Description	The use case allows the user to view the grant details after the user click on specific grant.
Actor(s)	User
Pre-conditions	1. The grants data are stored in the system. 2. The dashboard can be viewed. 3. The user searched for grants.
Normal Flow	1. The user selects a grant from the search result. 2. The system retrieves the information of the specific grant. 2.1. The system gets the funding organization and the grant data of each grant. AF1 will be performed. 3. The data is viewed in the grant details page.
Alternative Flow	1. If the system does not get data about the details view error message.
Exception Flow	
Post Conditions	The user can view the grant details including the amount, type and funding organization.

Related Requirements	

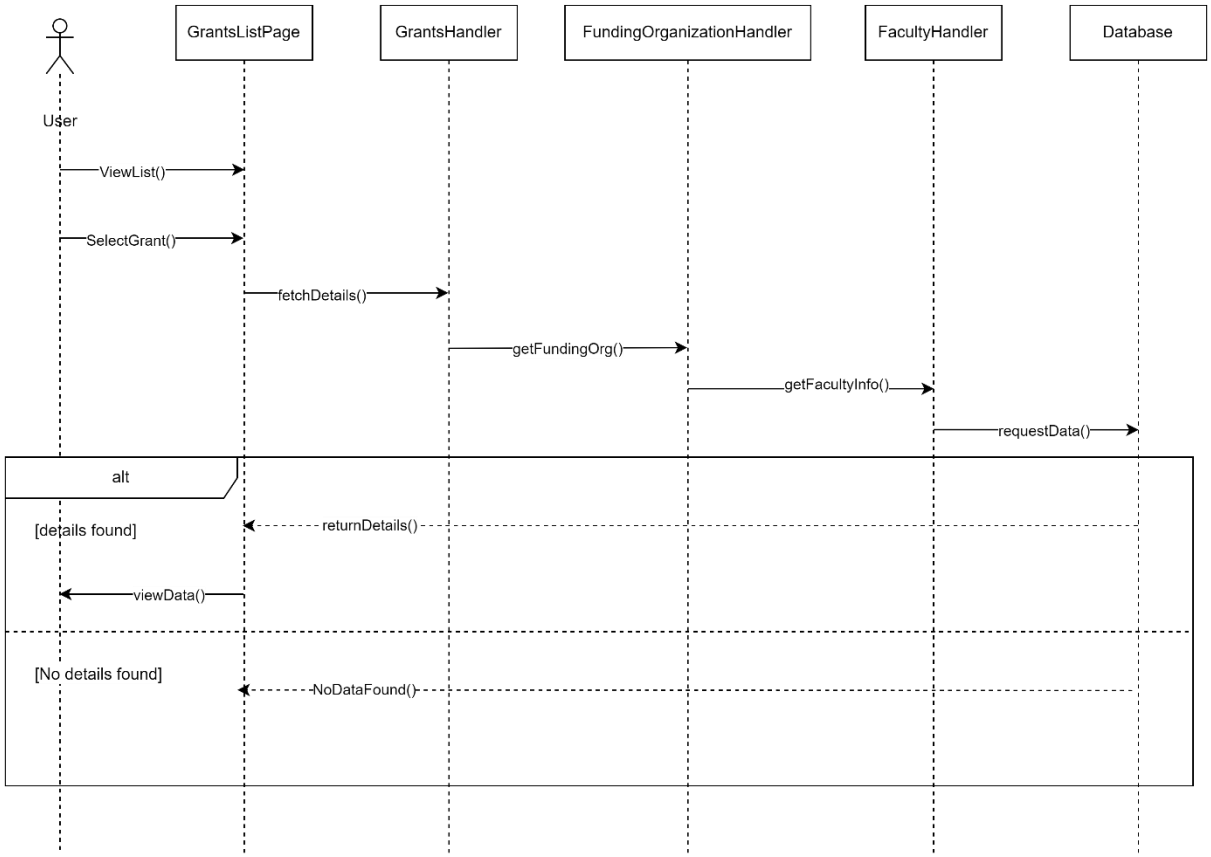


Figure 3.22: Sequence Diagram < View Grant Details >

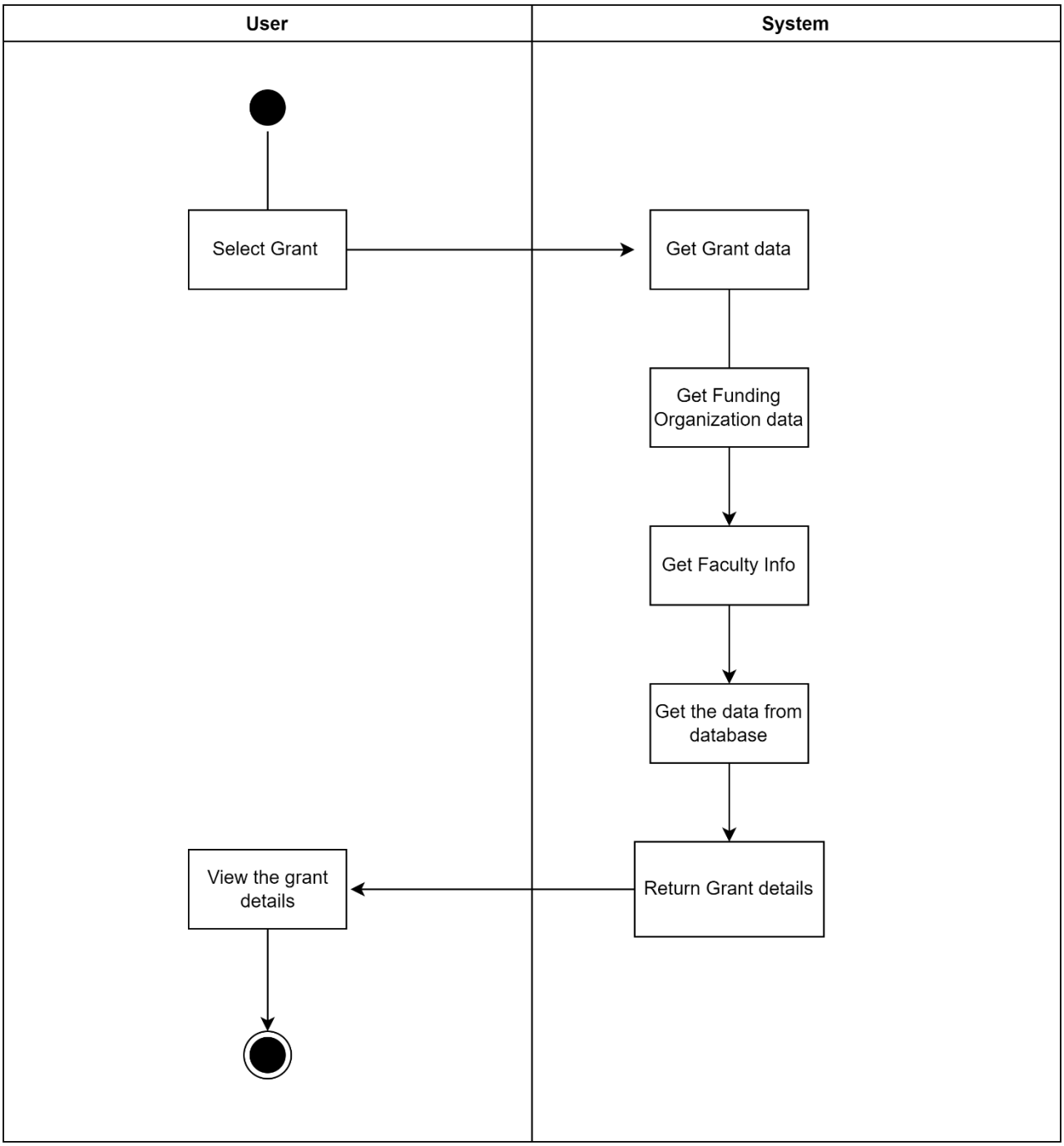


Figure 3.23: Activity Diagram < View Grant Details >

3.2.6.2 UC010: Use Case <Save Grants>

Table 3.5: Use Case Description for < Data Preprocessing and Dashboard Creation >

Use Case ID	UC
Use Case Name	**Save Grants**
Description	This use case allows the user who is logged in to save the grants he wants.
Actor(s)	User/ Researcher
Pre-conditions	1. The user must be logged in to the system. 2. The grant data are available in the database.
Normal Flow	1. Display the grant details page. 2. The user clicks on the save button. 3. The system stores the grant info. 4. The system accesses the user database. AF1 will be executed. 5. The system adds the grant info to the user database. 6. The system updates the system database.
Alternative Flow	1. If the system does not get the user data, view error message. NF1 will be executed.
Exception Flow	
Post Conditions	The user is able to save his grant details and he can view them in his profile.
Related Requirements	

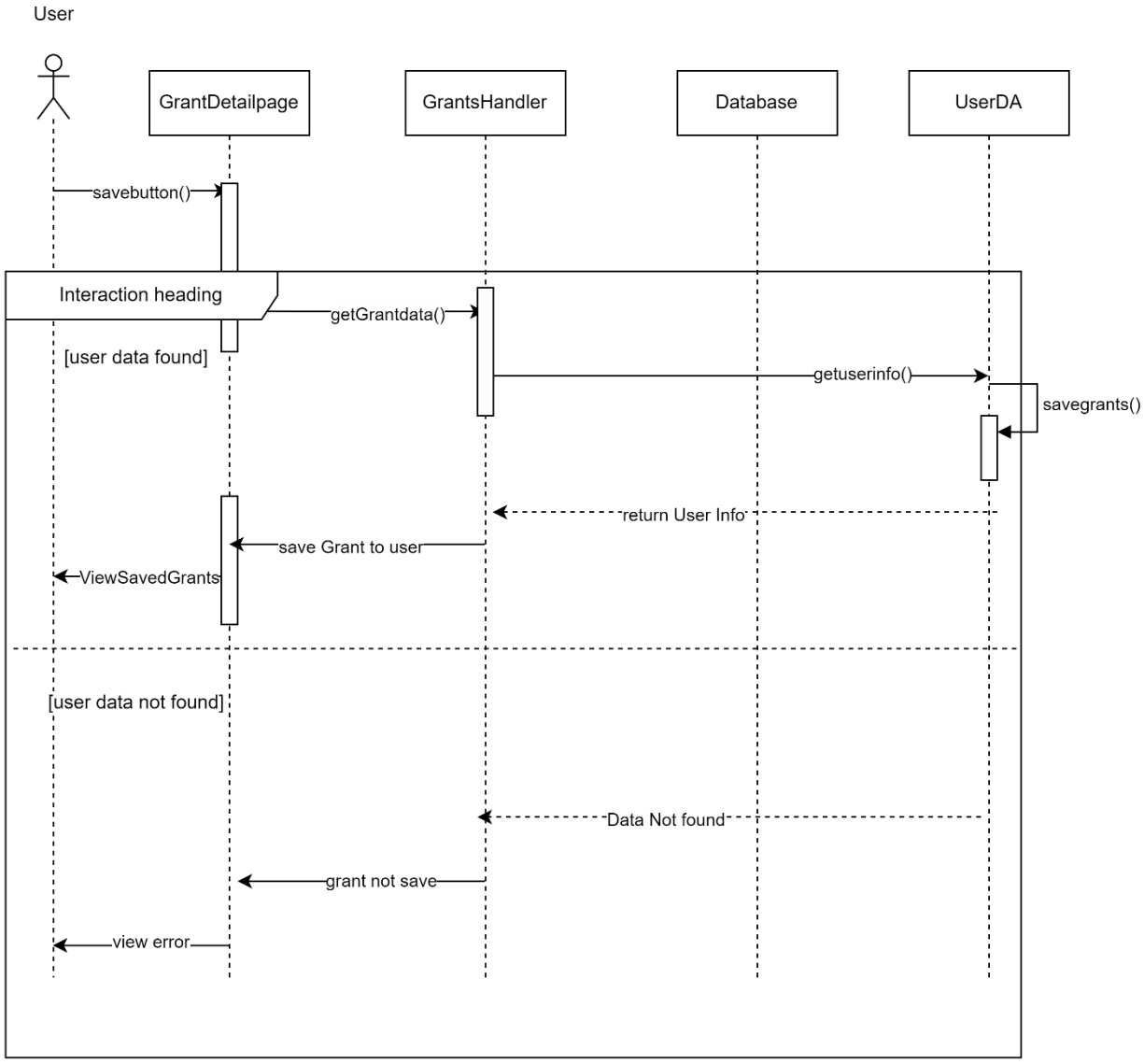


Figure 3.24: Sequence Diagram < Save Grants >

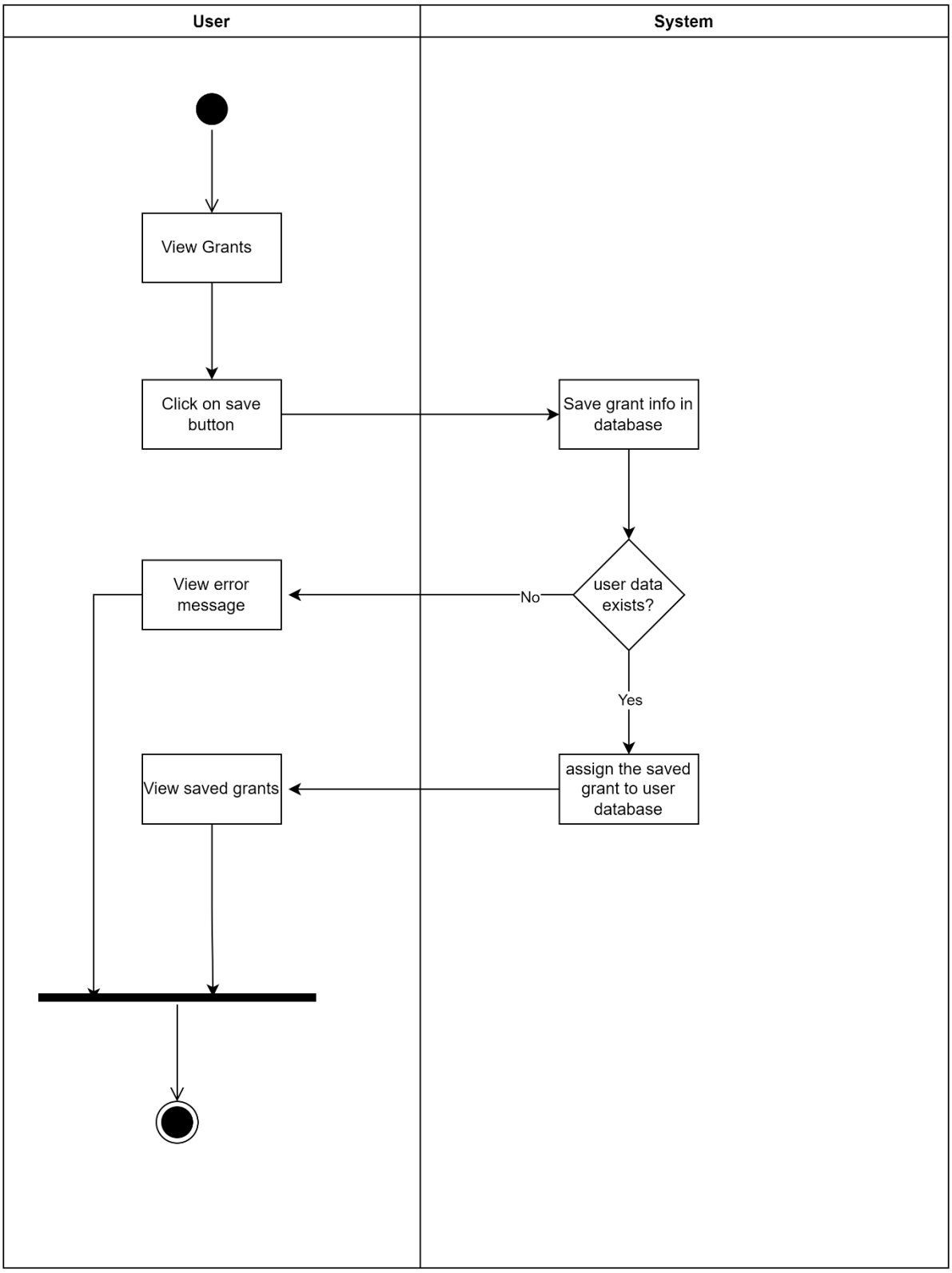


Figure 3.25: Activity Diagram < Save Grants >

3.3 Performance Requirements

I. Usability - To eliminate unintentional errors, the system includes prevention mechanisms such as pop-ups to verify if anything is incorrect before processing.

II. Security - To ensure data integrity, all data and information should be delivered securely to the server.

III. Response time - In order to properly accomplish particular activities, the system must have a fast-loading time and a high processing speed. Because of the importance of this criterion, it must be performed as soon as possible so that users can move on to the next phase in their workflow. In this case, the reaction time for the system to fully load into the system's main menu after the login session is successful must be less than five seconds.

IV. Failure contingencies - If the system fails, the user will be notified via error pages or screen messages. To avoid data loss, all information will be frozen to preserve it.

3.4 Design Constraints

It is suggested that the system be built like a normal web-based system, which can work on any web browser as long as it is connected to the internet.

3.5 Software System Attributes

I. Security: Only authorized users with different levels of access should be able to do specific tasks on the system. Aside from that, the information in the system is kept safe and secure by limiting approved access to the end-user in this system. Only the admins with specific rights can modify the dashboard.

II. Portability: Since the system is web-based, the user should be able to use any web browser that works and has a good link to the internet to access it. Google Chrome, Mozilla Firefox, Microsoft Edge, Apple Safari, and Opera are all examples of these kinds of web platforms.

III. Availability: Users should be able to use the system at any time and from anywhere, and it should be available 24 hours a day. In this case, users should be able to use the system during normal work hours.

IV. Maintainability: The people who make the system have to keep it in good shape so that they can find any problems and fix them by changing the module in question. To keep the system's quality and usefulness at a high level, it needs to be tested on a regular basis.

APPENDIX B: SYSTEM DESIGN DOCUMENT

SCSJ3323: Software Design and Architecture

Software Design Document

Research Grant Finder

Version 1.0

Printing Date

Department and Faculty

Prepared by: <Islam Mohammed Ruzhan>

REVISION PAGE

e. Overview

Describe the content of the current version.

f. Target Audience

State the targeted audience.

g. Project Team Members

List the team members and respective assigned module.

h. Version Control History

Version	Primary Author(s)	Description of Version	Date Completed
<1.0>	**Islam Mohammed Ruzhan**		

Note:
This template is an annotated outline for a software design document adapted from the IEEE Recommended Practice for Software Design Descriptions. The IEEE Recommended Practice for Software Design Descriptions have been reduced in order to simplify this assignment while still retaining the main components and providing a general idea of a project definition report. Please refer to IEEE Std 1016-1998 1 for the full IEEE Recommended Practice for Software Design Descriptions. Examples of models are from

Satzinger (2011). Compiled by Shahliza Abdul Halim, PhD and checked by Shahida Sulaiman, PhD on 2 May 2016.

TABLE OF CONTENTS

1. INTRODUCTION

The system architecture, database design, description of the data and user interface design of the Research Grant Finder System will be described in this Software Design Document (SDD).

1.1 Purpose

This SDD is documented for the following purposes:

i. To describe the system architecture design and database design of the Research Grant Finder System.

ii. To describe and illustrate the user interface design of the Research Grant Finder System.

1.2 Scope

The system that will be built is Research Grant Finder. It is a web-based dashboard that will be created by using data visualization on the extracted data that will be obtained by web scraping from the target websites.

With the help of this system, the researchers can view the grant statistics of the faculty and other information such as numbers of grants received by respective faculty, Highest grant providing organization and amount of grant received by respective researcher. A researcher can also look for grants from the extracted data from the websites and filter his/her search. A researcher can also save the grants he wish to apply for in future. Overall, this web application will be very helpful for the research works of the university as it provides opportunities to researchers to apply for grants more easily than it is at present.

1.3 Definitions, Acronyms and Abbreviation

Term	**Definition**
FR	Functional Requirement
SDD	Software Design Document
MVC	Model- View-Controller
NFR	Non-Functional Requirement

1.4 References

i. Sommerville, 2016. “Software Engineering”, 10th Edition, Addison Wesley

ii. IEE Recommended Practice for Software Requirements Specification

Specify complete list of references using a standardized reference format.

1.5 Overview

This document contains 5 sections which are described below:

i. Introduction: This section provides an overview of the SDD.

ii. System Overview: This section describes the overall functionality provided by the application to the users.

iii. System Architecture: This section describes the chosen system architecture in the application to be developed.

iv. Database Design: This section describes the database of the application to be implemented. It includes the data dictionary which describes all the data attributes used in the system.

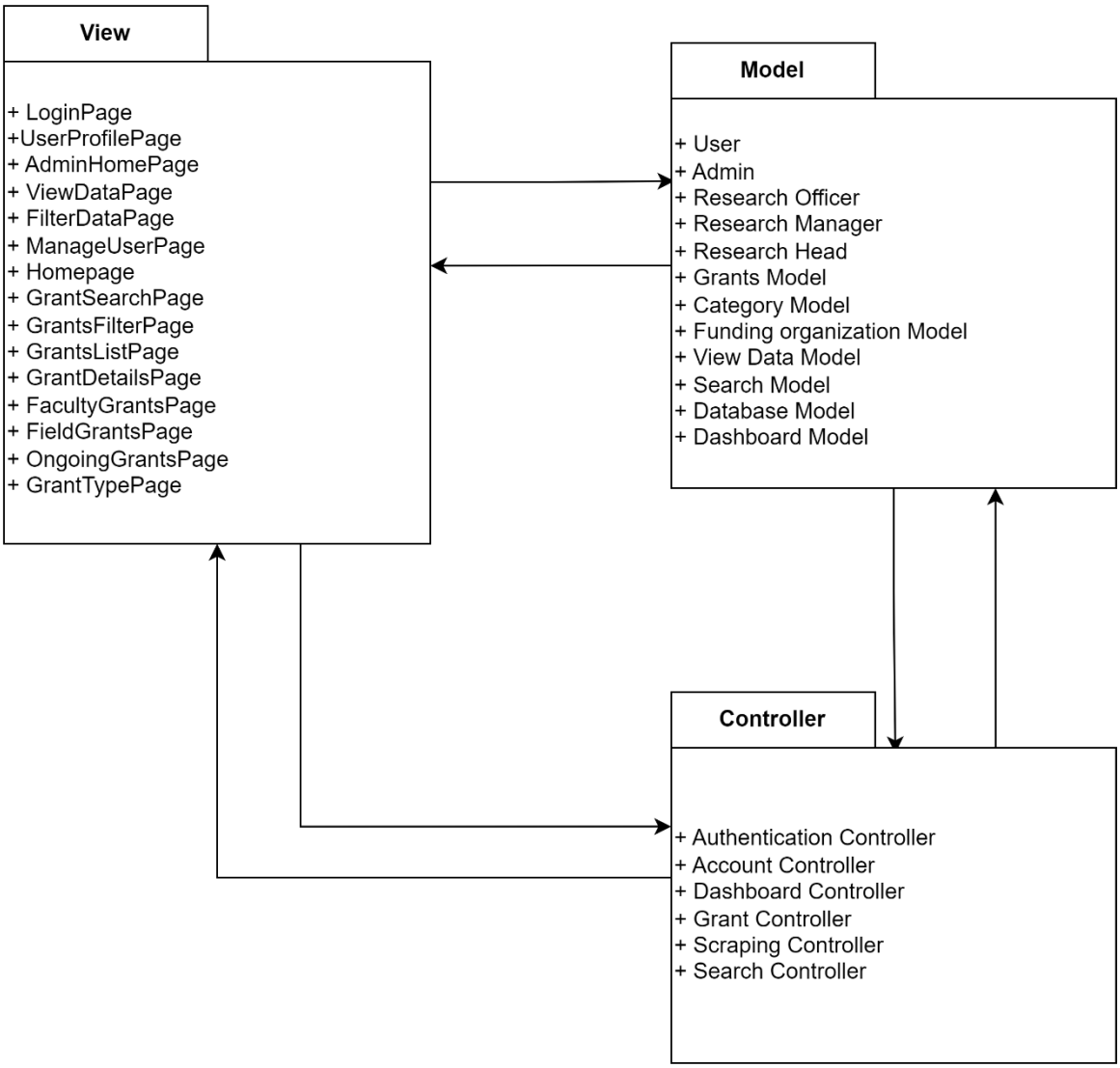
v. Interface Design: This section describes the view of how the application will look like for the user to interact with the application easily.

2. SYSTEM ARCHITECTURAL DESIGN

2.1 Architecture Style and Rationale

MVC or Model-View-Controller is a software architecture pattern in which an application is divided into three interrelated components: the Model, the View, and the Controller. It is commonly utilized in web application design and implementation of user interfaces. This is the most commonly used design pattern by developers. The MVC design pattern has numerous advantages in system design and implementation. This design approach makes it much easier to manage and modify code.

In an MVC design pattern, the model is utilized to handle database operations such as insertion, deletion, update, and retrieval. These interactions occur between the controller and the system database. The view is the system's visual representation that shows the user interfaces to the end-user and takes user input from those interfaces. It also shows the end-user the system outputs. Invoking the functions, processing the input, and sending the output to the display are all critical tasks of the controller. The diagram below is an architectural model that depicts the MVC architecture design pattern that will be used to develop Research Grant Finder System.



2.2 Component Model

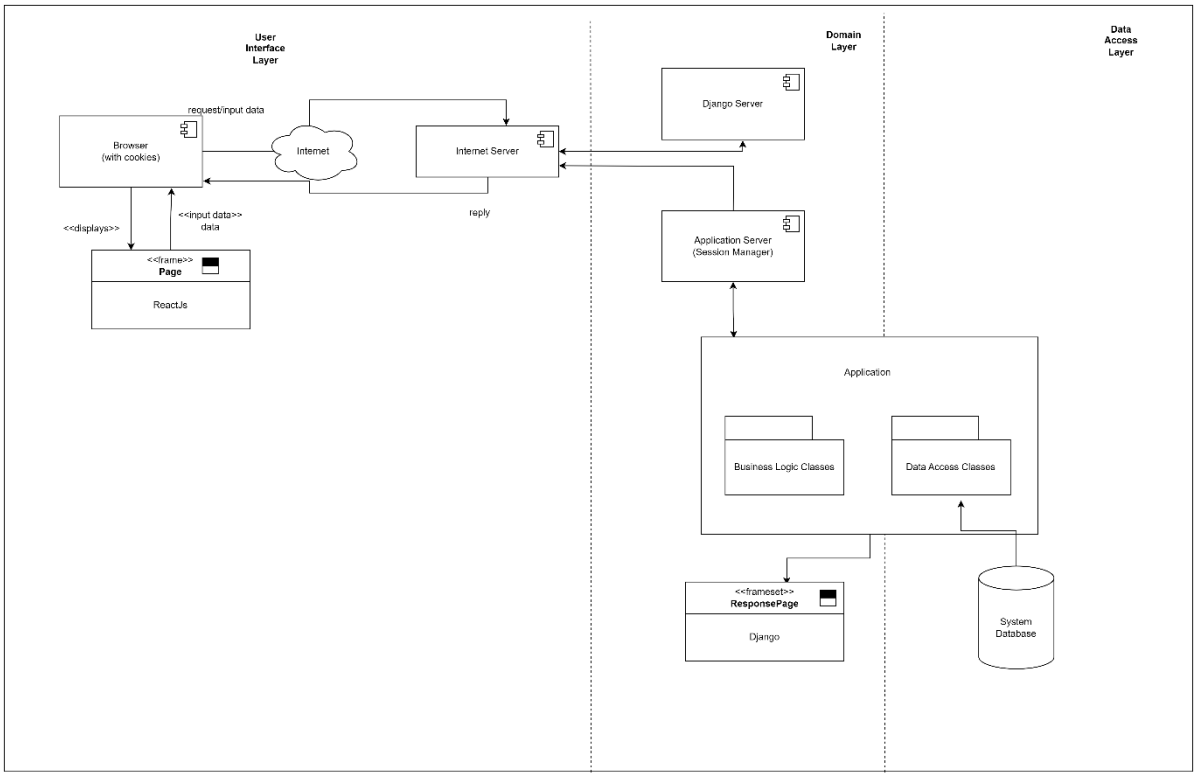


Figure 2.1: Component Model of < Research Grant Finder System>

The component model of the Research Grant Finder System is drawn above. A stable internet connection is necessary so that the system works properly. The language used for interface designing is ReactJS and for the database to communicate with the backend, which is Django, we used MongoDB. The grant data will be stored in the system database after they are extracted from the websites.

2.3 Use Case Diagram

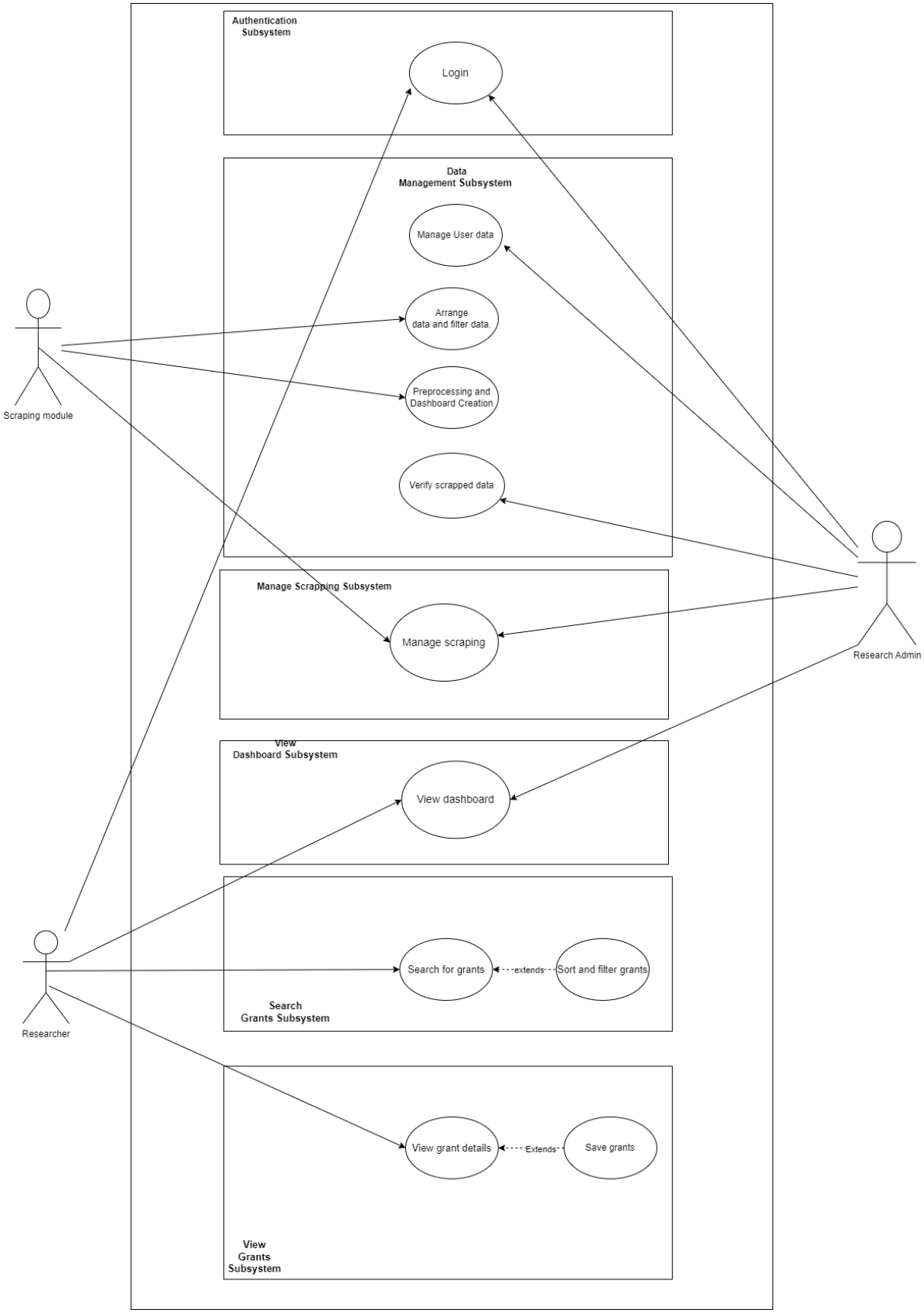


Figure 2.2: Use Case Diagram of <Research Grant Finder>

3. DETAILED DESCRIPTION OF COMPONENTS

3.1 Complete Package Diagram

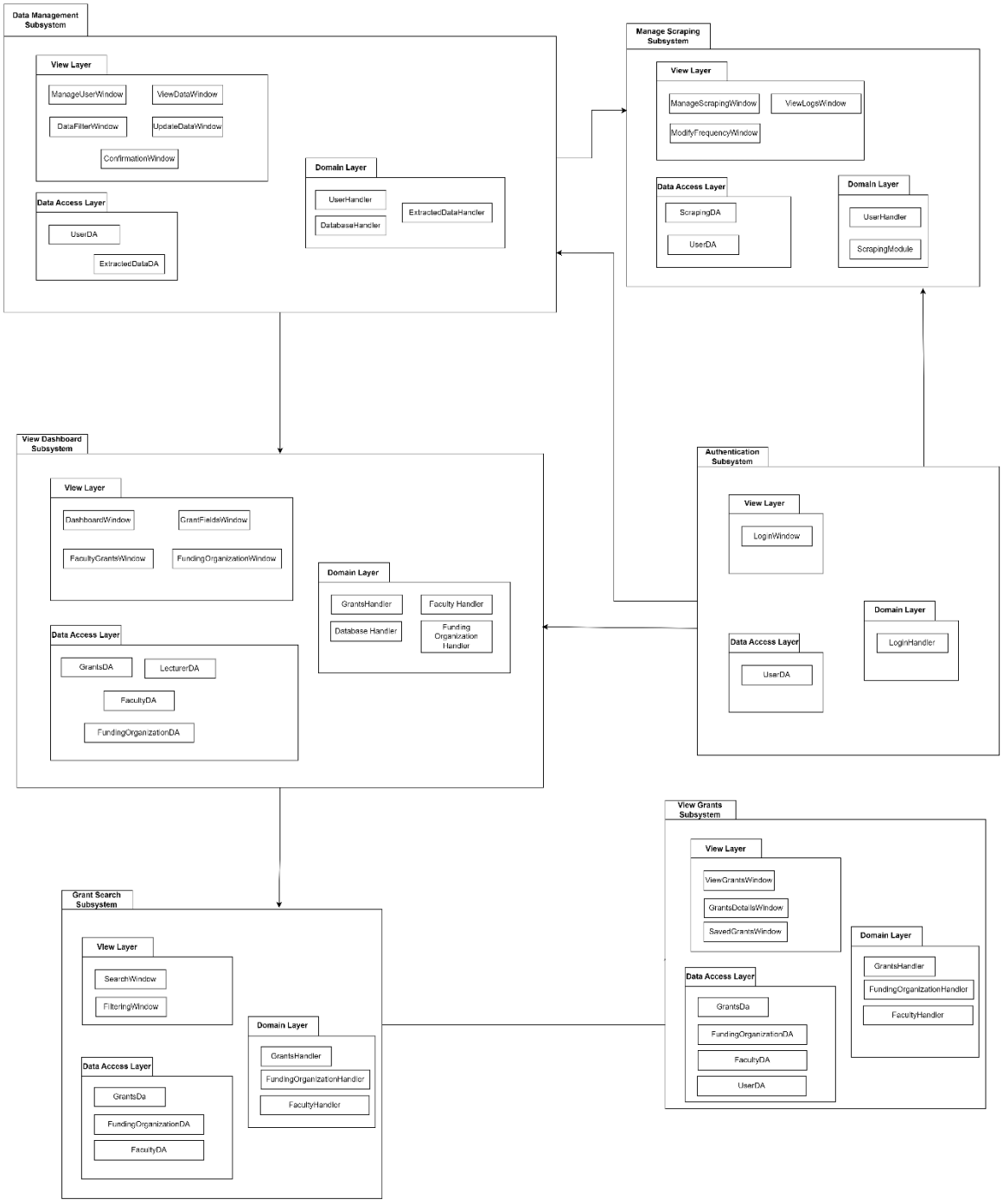
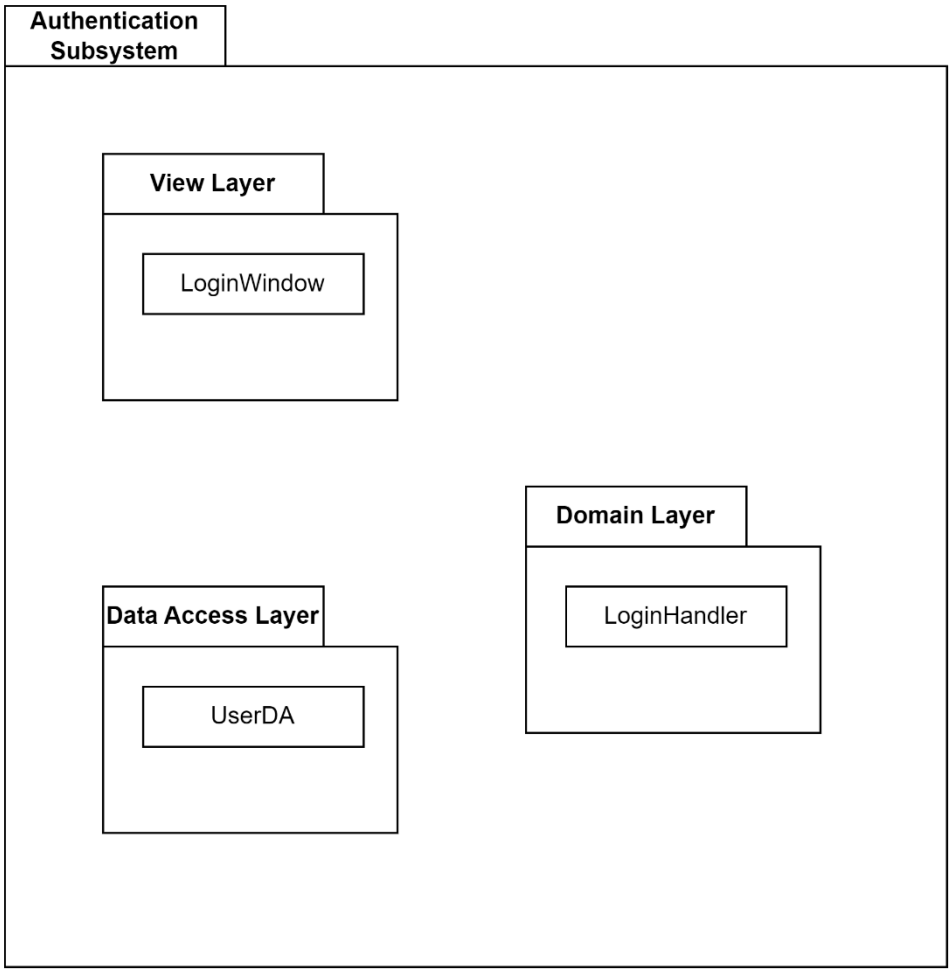


Figure 3.1: Subsystem of <Name of the System>

3.2 Detailed Description

3.2.1 Subsystem <Authentication>

3.2.1.1 P001: Package <Authentication>



3.2.1.2 Class Diagram

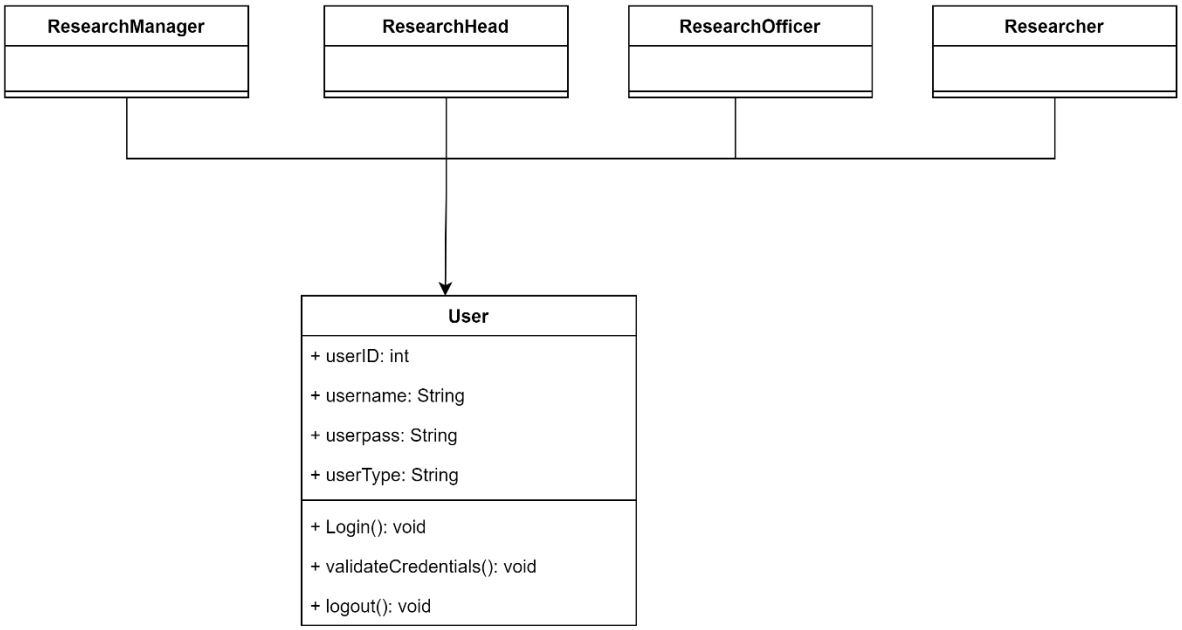


Figure 3.2: Class diagram for <Authentication>

3.2.1.3 Sequence Diagrams

a) SD001: Sequence diagram for Create New Phone Order

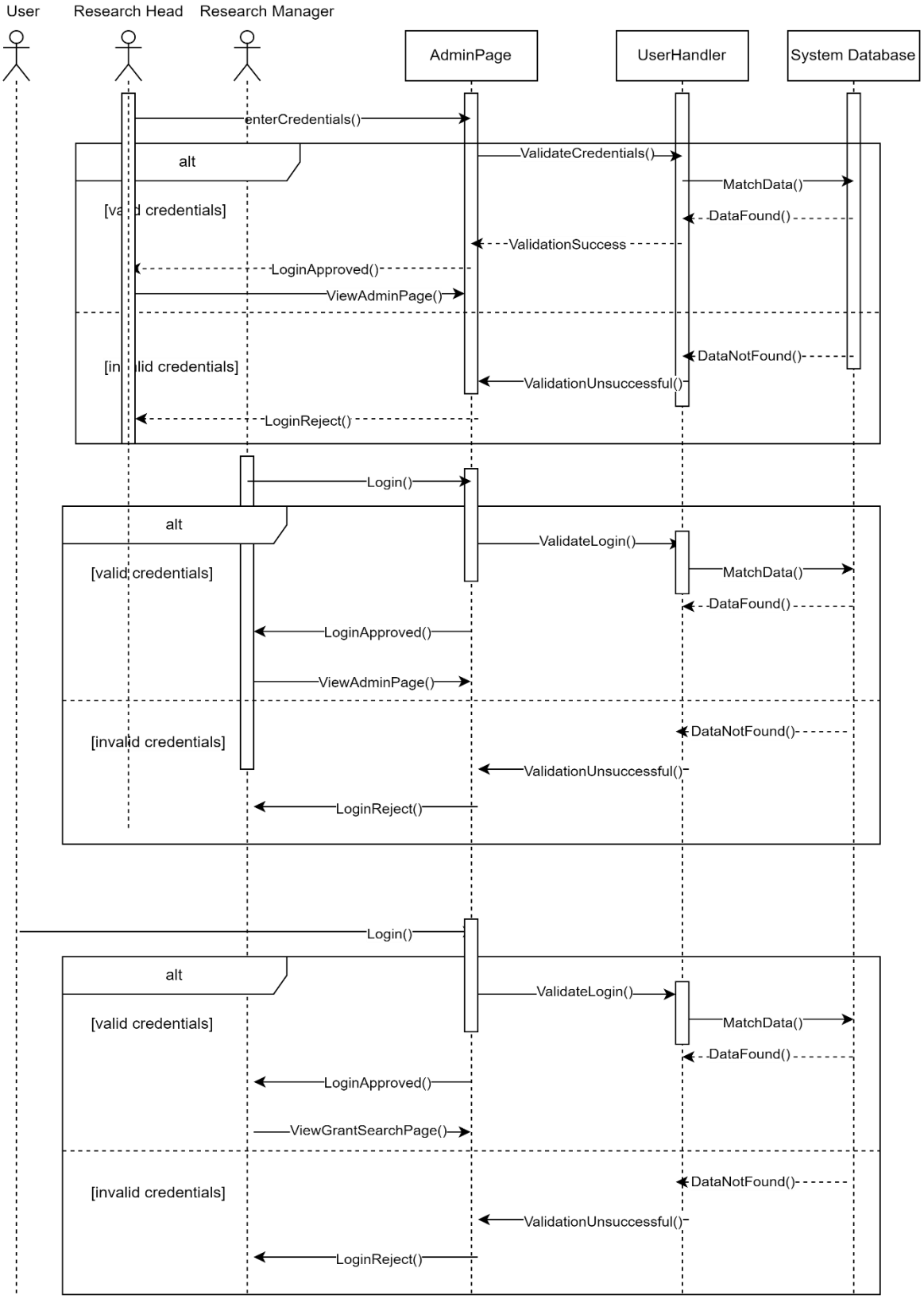
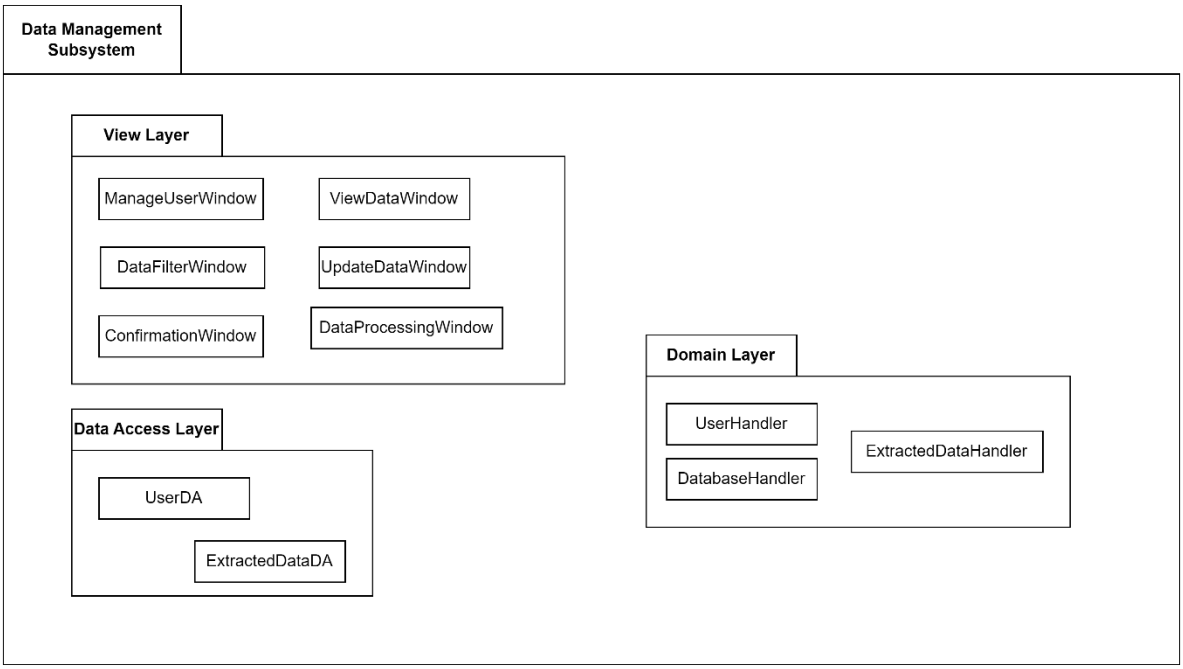


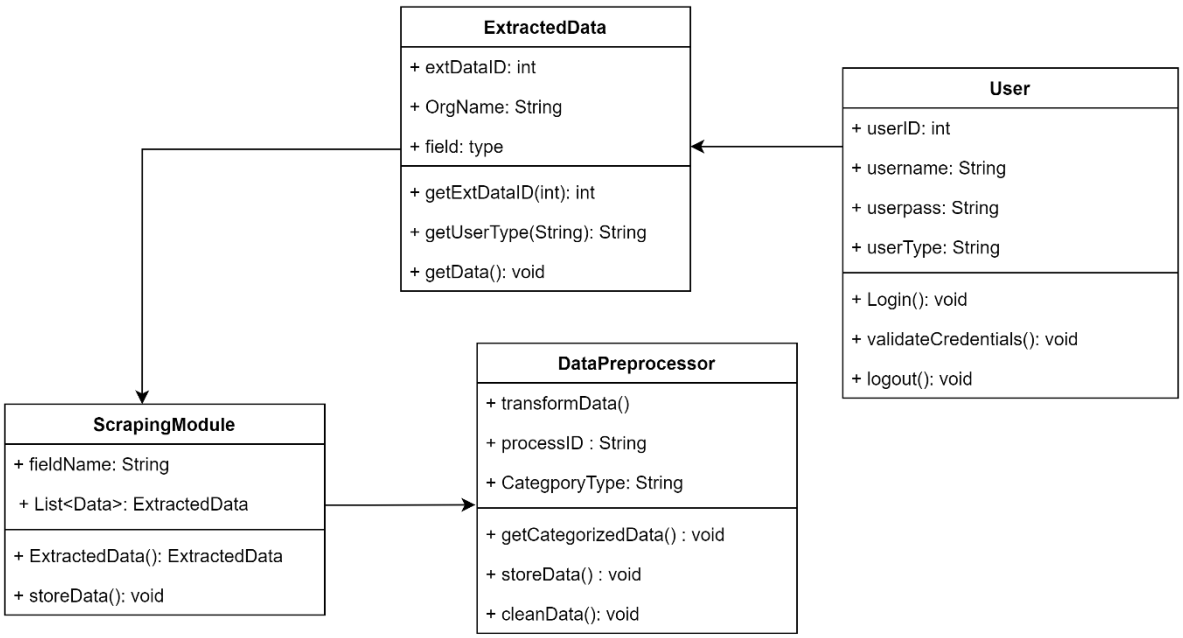
Figure 3.3: Sequence Diagram of <Login>

3.2.2 Subsystem <Data Management>

3.2.2.1 P002: Package < Data Management >

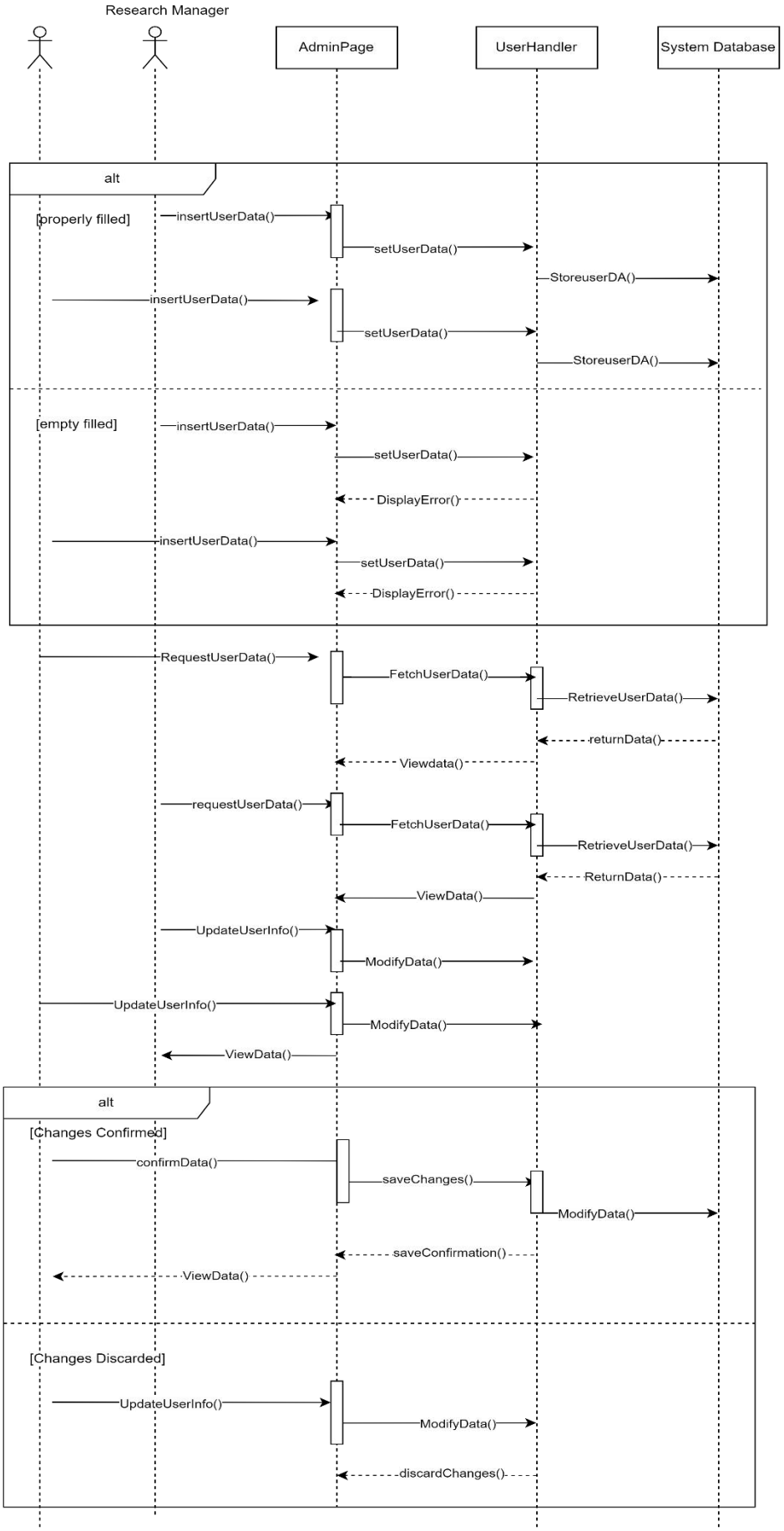


3.2.2.2 Class Diagram

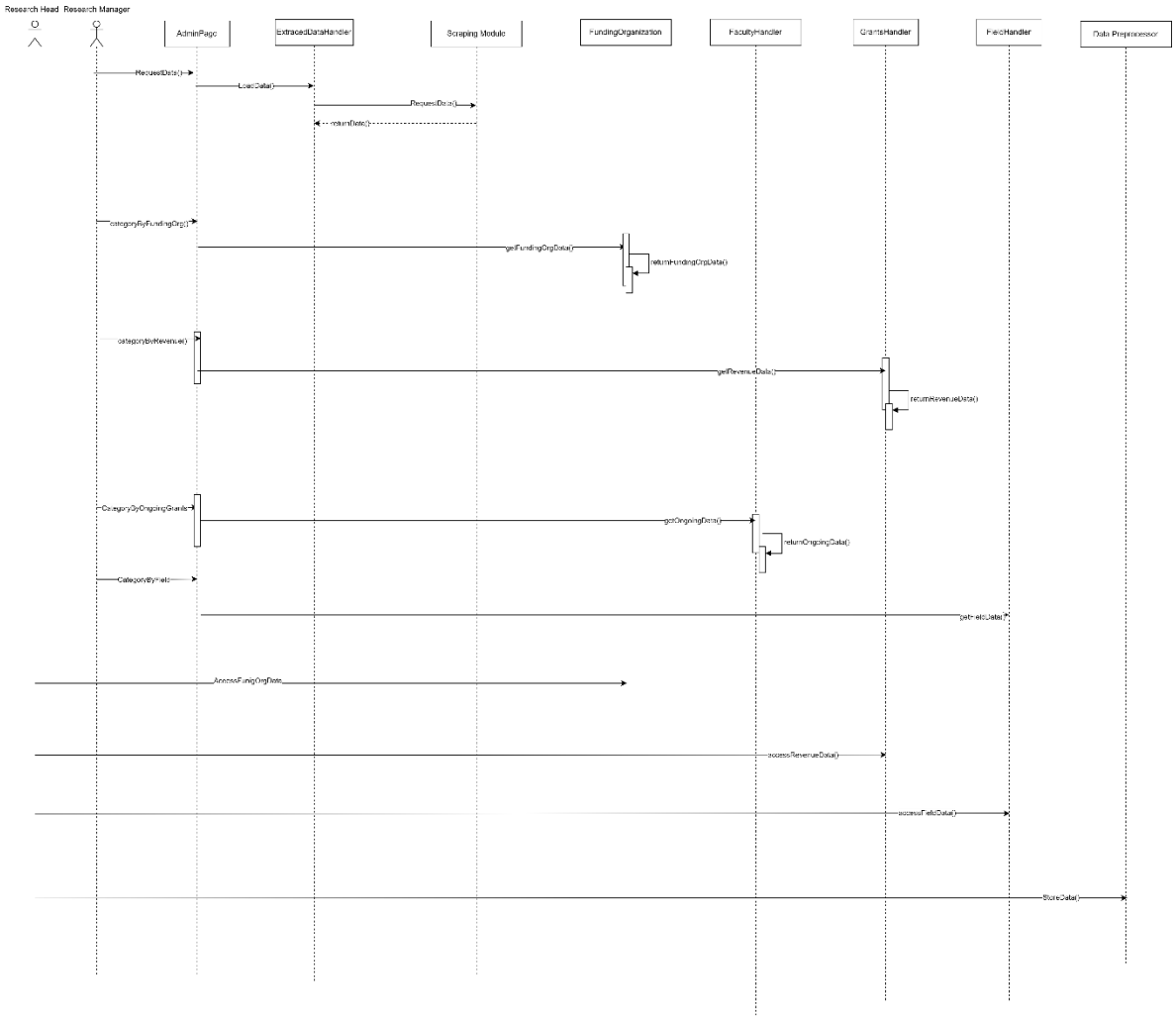


3.2.2.3 Sequence Diagrams

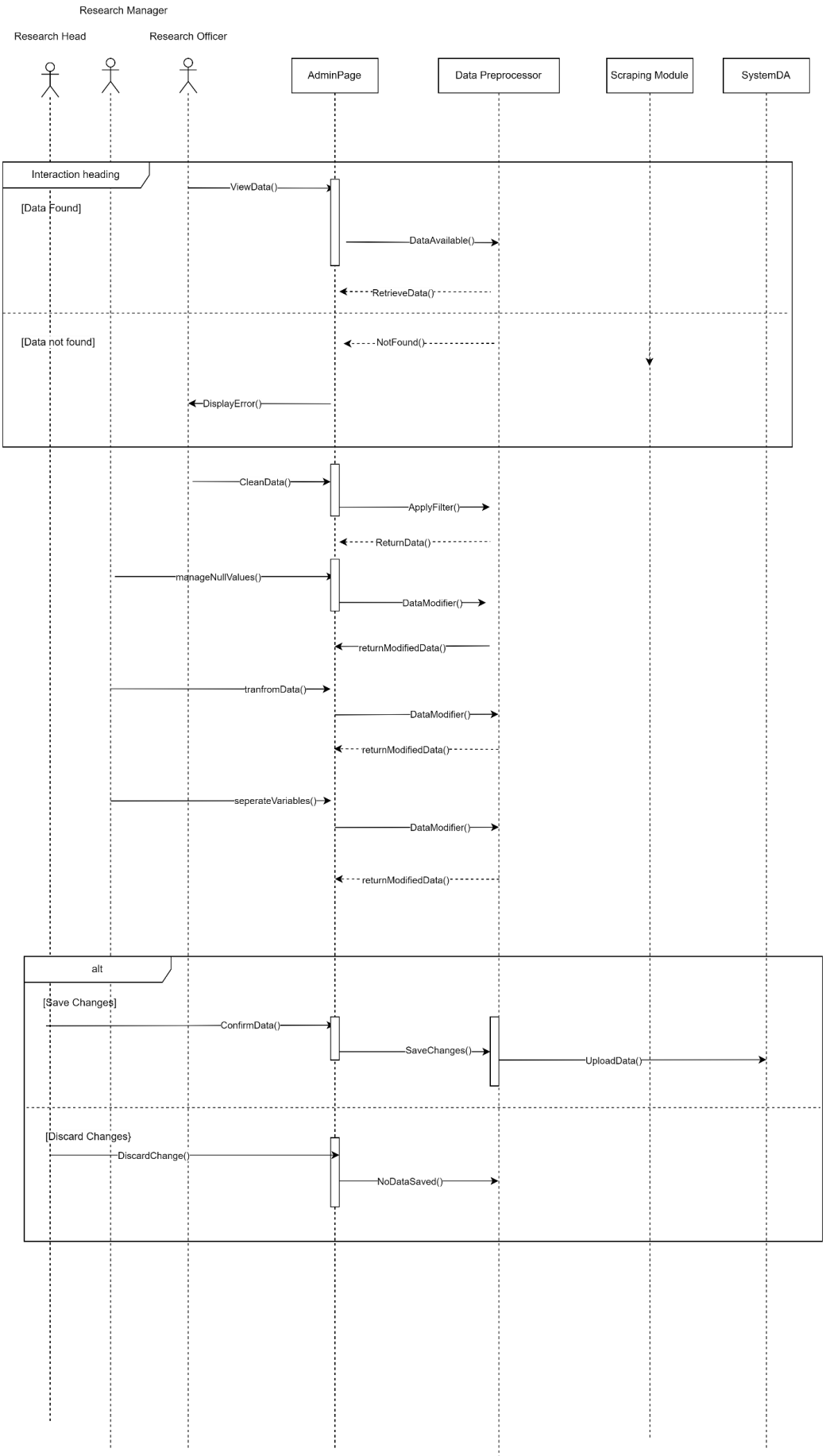
SD002: Sequence diagram for Manage user Data


Research Manager
AdminPage
UserHandler
System Database
alt
[properly filled]
insertUserData()
setUserData()
StoreuserDA()
insertUserData()
setUserData()
StoreuserDA()
[empty filled]
insertUserData()
setUserData()
DisplayError()
insertUserData()
setUserData()
DisplayError()
RequestUserData()
FetchUserData()
RetrieveUserData()
returnData()
Viewdata()
requestUserData()
FetchUserData()
RetrieveUserData()
ReturnData()
ViewData()
UpdateUserInfo()
ModifyData()
UpdateUserInfo()
ModifyData()
ViewData()
alt
[Changes Confirmed]
confirmData()
saveChanges()
ModifyData()
saveConfirmation()
ViewData()
[Changes Discarded]
UpdateUserInfo()
ModifyData()
discardChanges()

SD004: Sequence diagram for Arrange and Filter Data

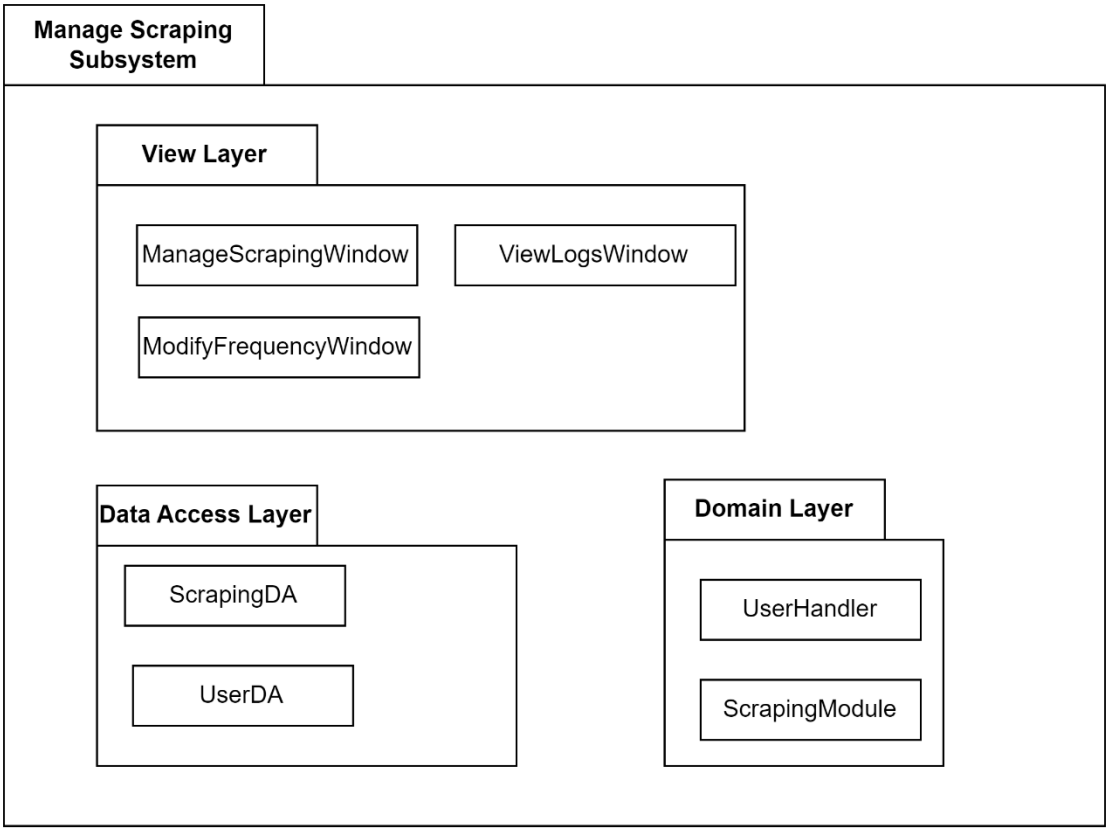


SD005: Sequence diagram for Data Preprocessing and Dashboard Creation

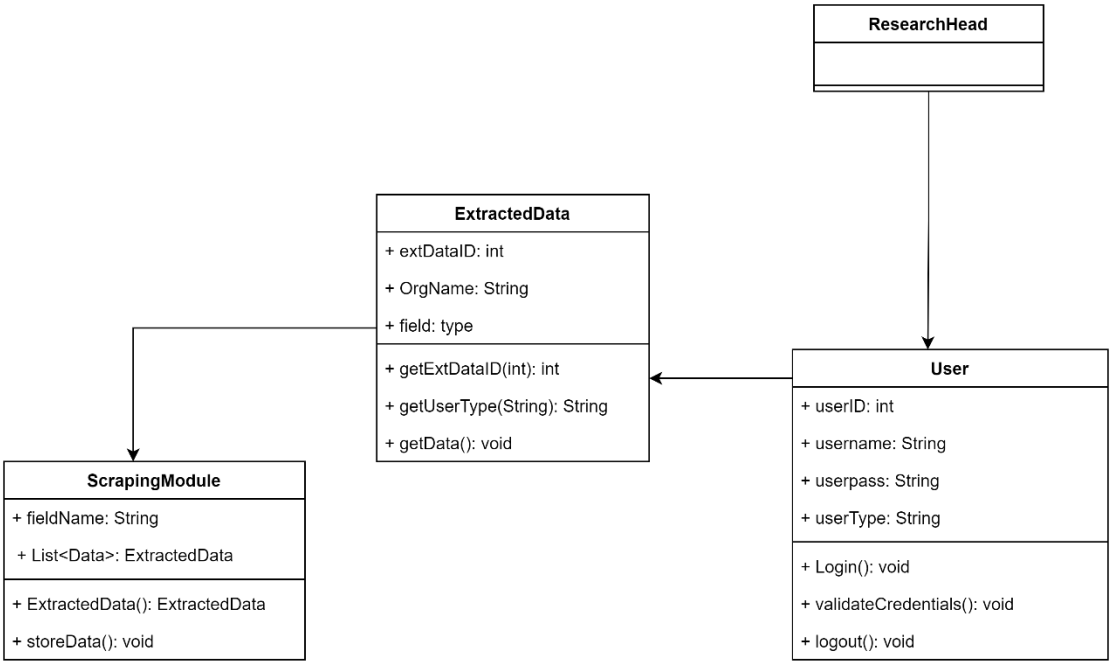

Research Manager
Research Head
Research Officer
AdminPage
Data Preprocessor
Scraping Module
SystemDA
Interaction heading
[Data Found]
ViewData()
DataAvailable()
RetrieveData()
[Data not found]
NotFound()
DisplayError()
CleanData()
ApplyFilter()
ReturnData()
manageNullValues()
DataModifier()
returnModifiedData()
tranfromData()
DataModifier()
returnModifiedData()
seperateVariables()
DataModifier()
returnModifiedData()
alt
[Save Changes]
ConfirmData()
SaveChanges()
UploadData()
[Discard Changes]
DiscardChange()
NoDataSaved()

3.2.3 Subsystem <Manage Scraping>

3.2.3.1 P003: Package < Manage Scraping >

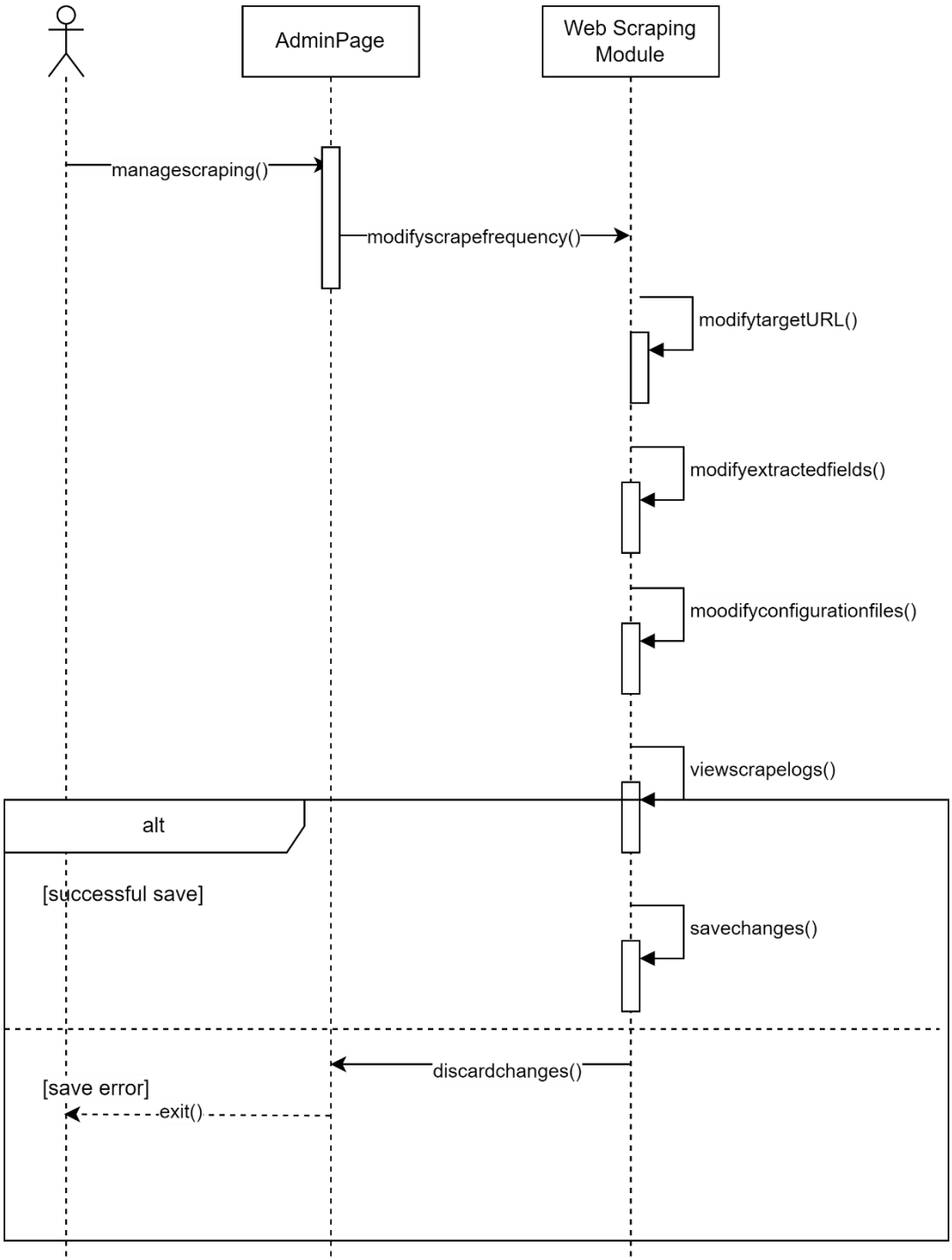


3.2.3.2 Class Diagram



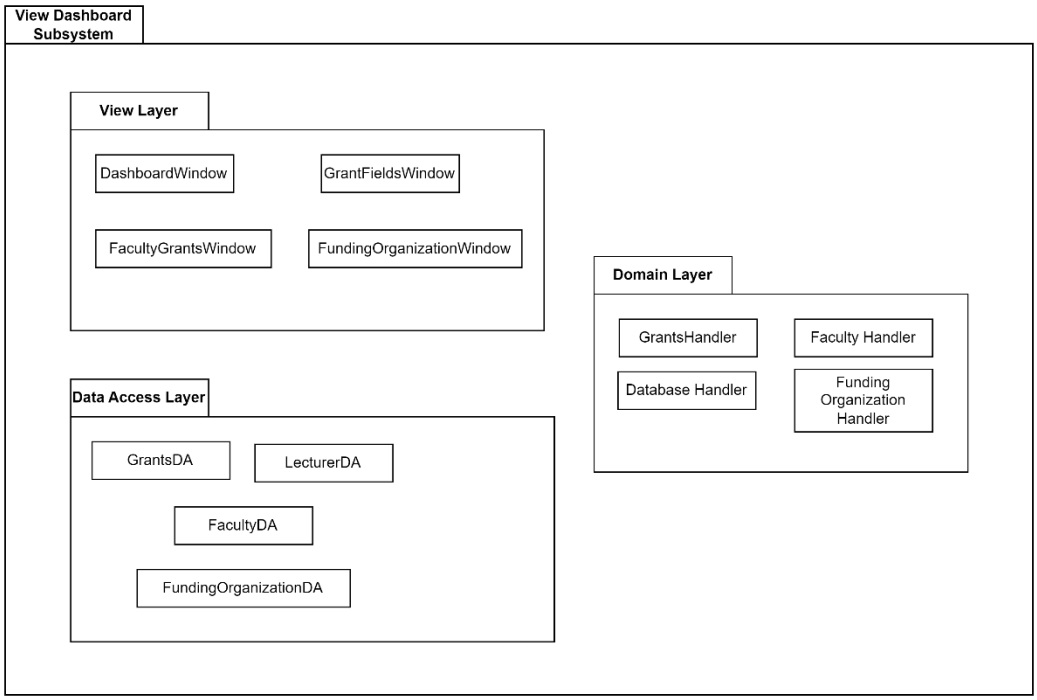
3.2.3.3 Sequence Diagrams

SD006: Sequence diagram for Data Preprocessing and Dashboard Creation

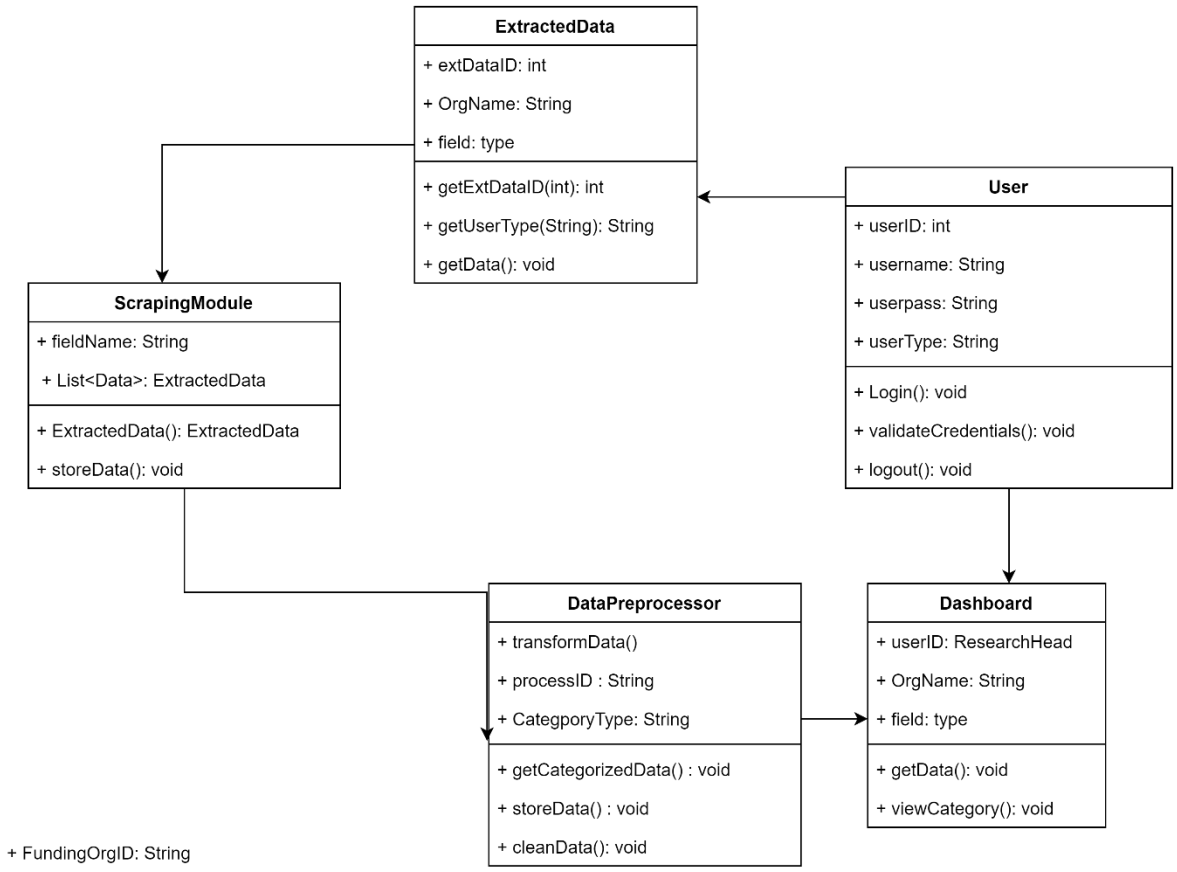


3.2.4 Subsystem <View Dashboard>

3.2.4.1 P004: Package < View Dashboard >

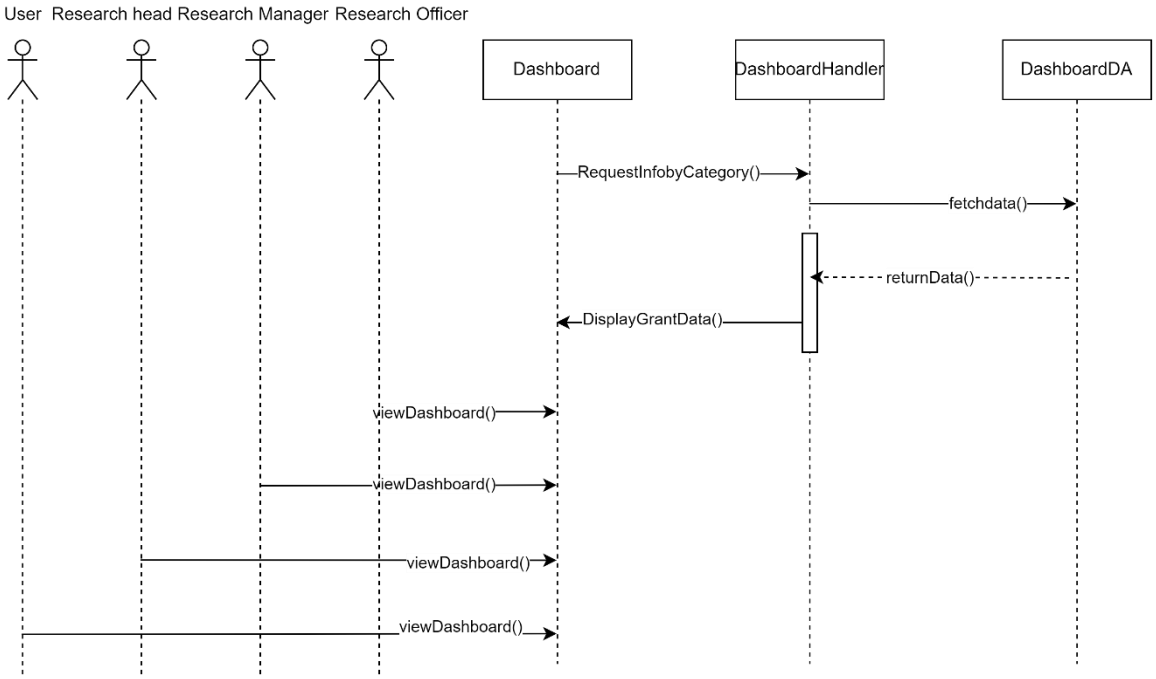


3.2.4.2 Class Diagram



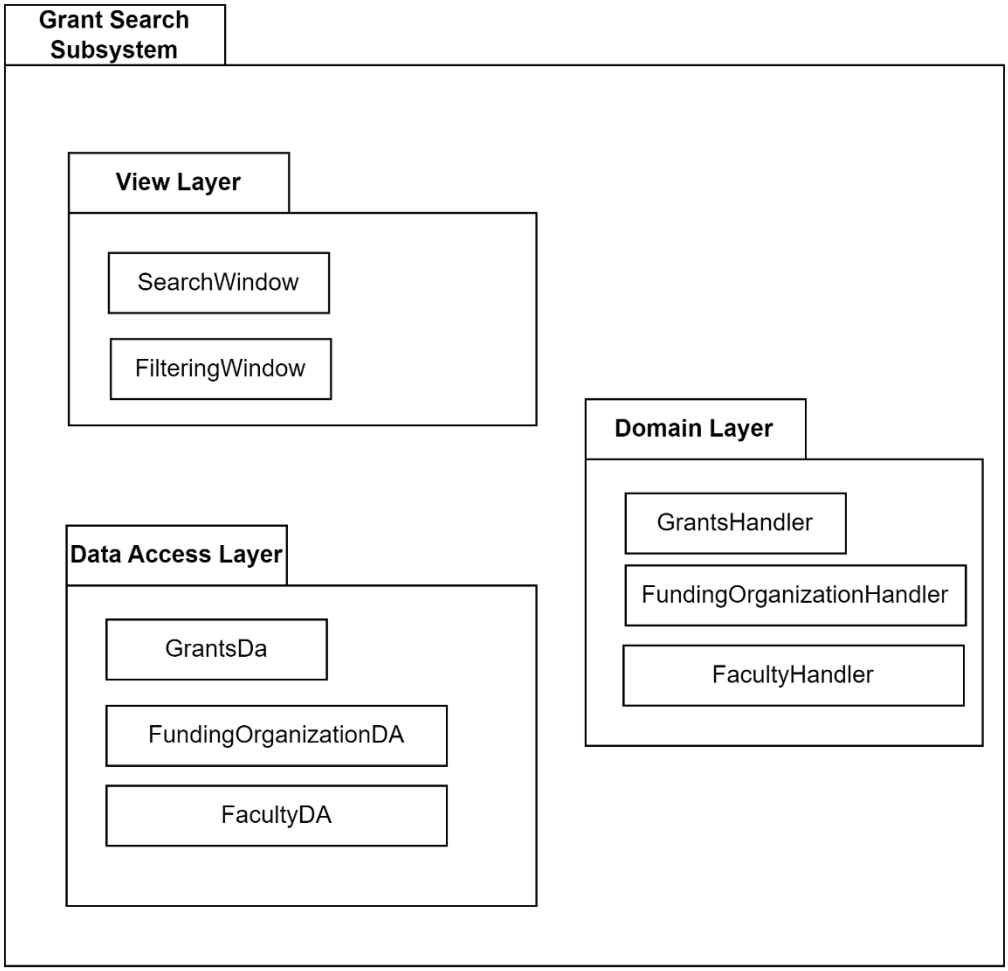
3.2.4.3 Sequence Diagrams

SD007: Sequence diagram for View Dashboard

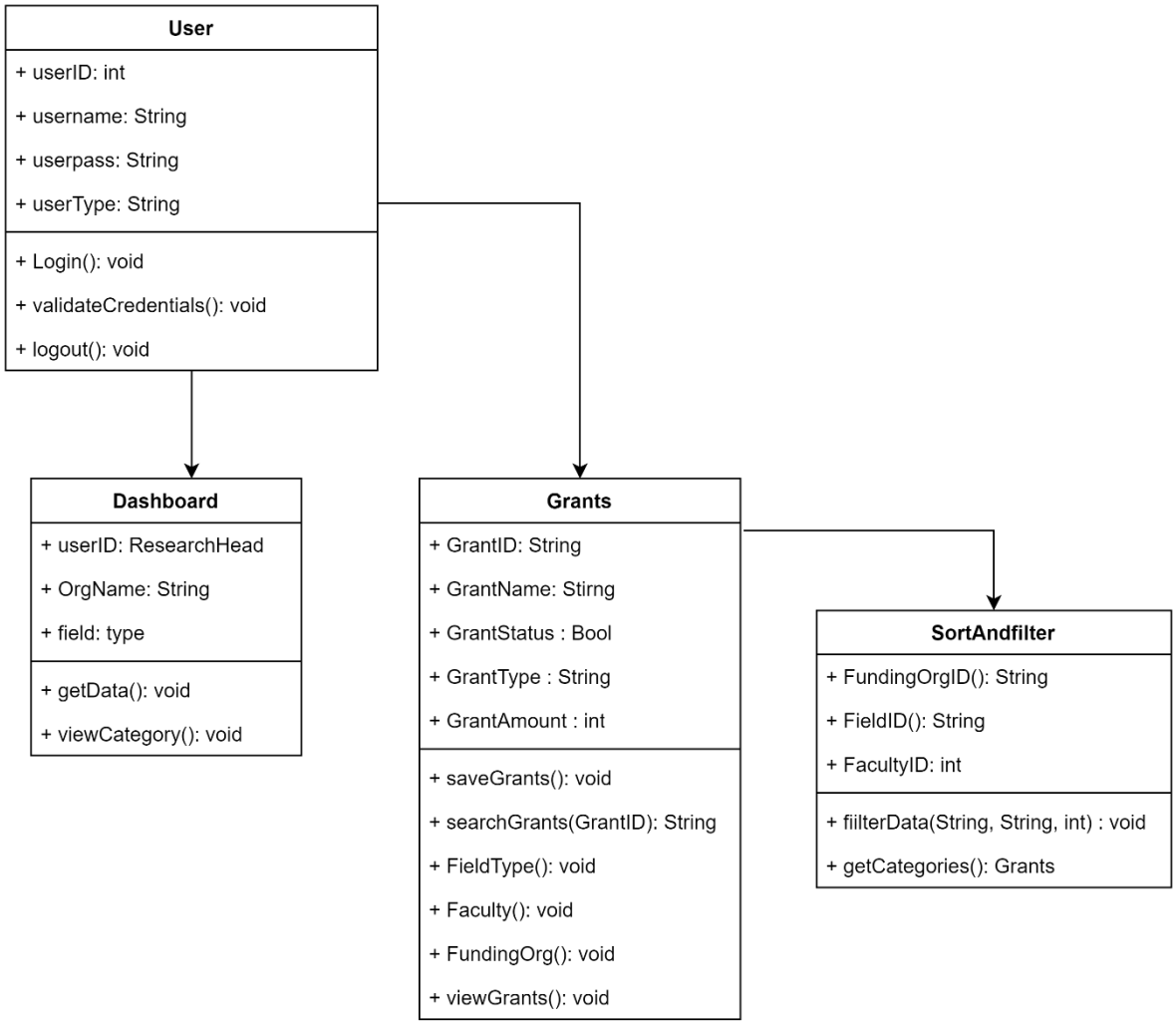


3.2.5 Subsystem <Search Grants>

3.2.5.1 P005: Package < Search Grants >

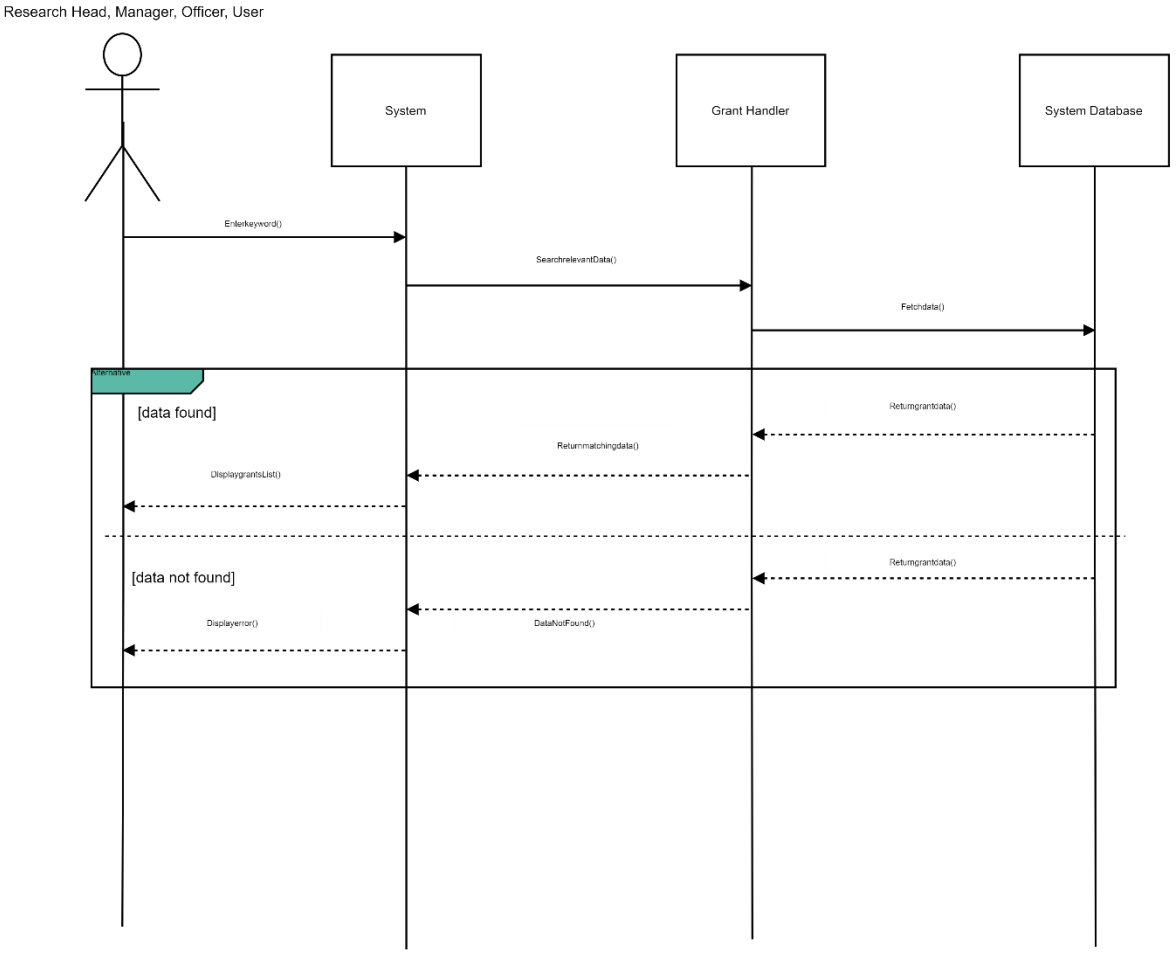


3.2.5.2 Class Diagram

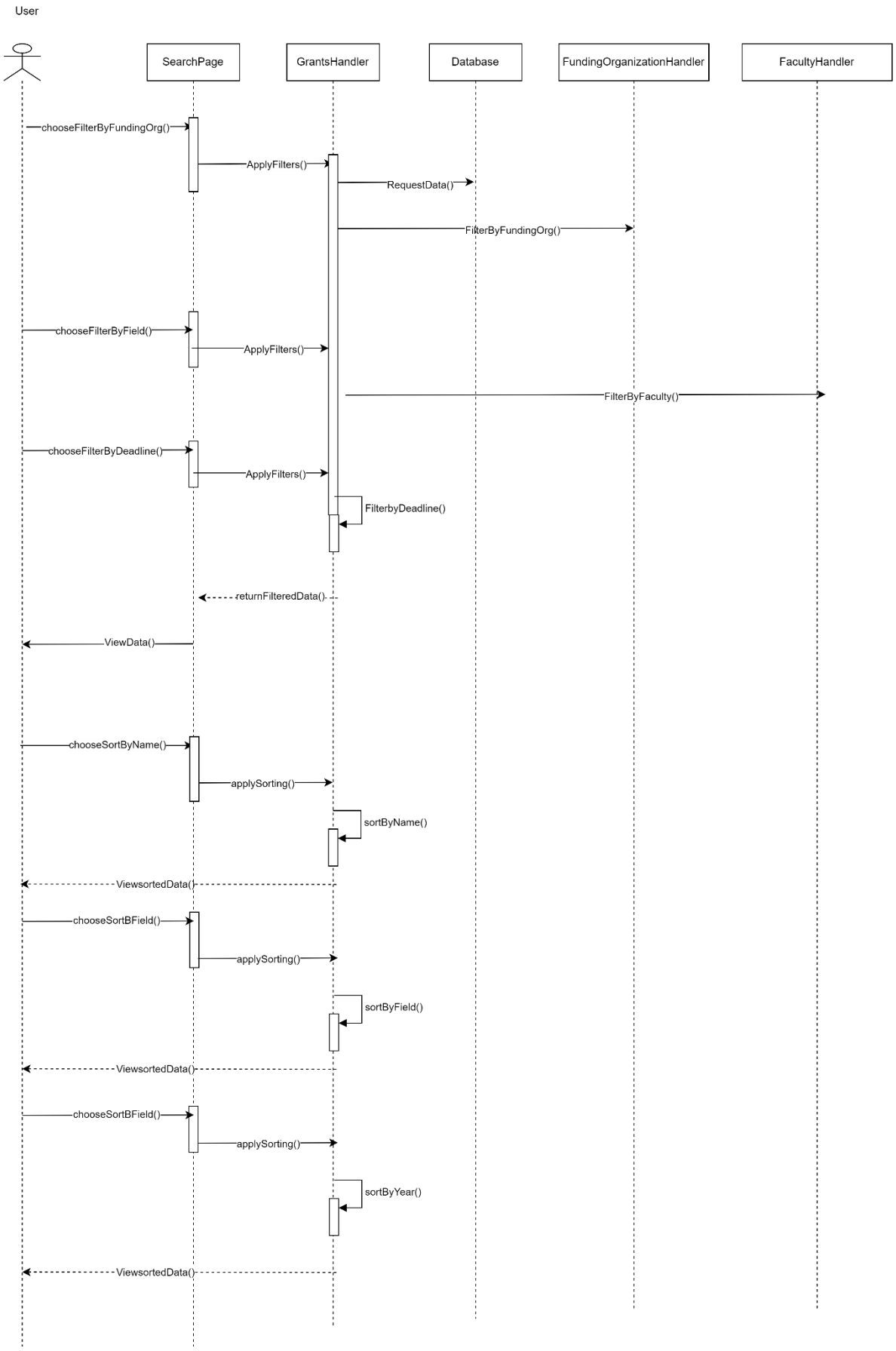


3.2.5.3 Sequence Diagrams

SD008: Sequence diagram for Search Grants

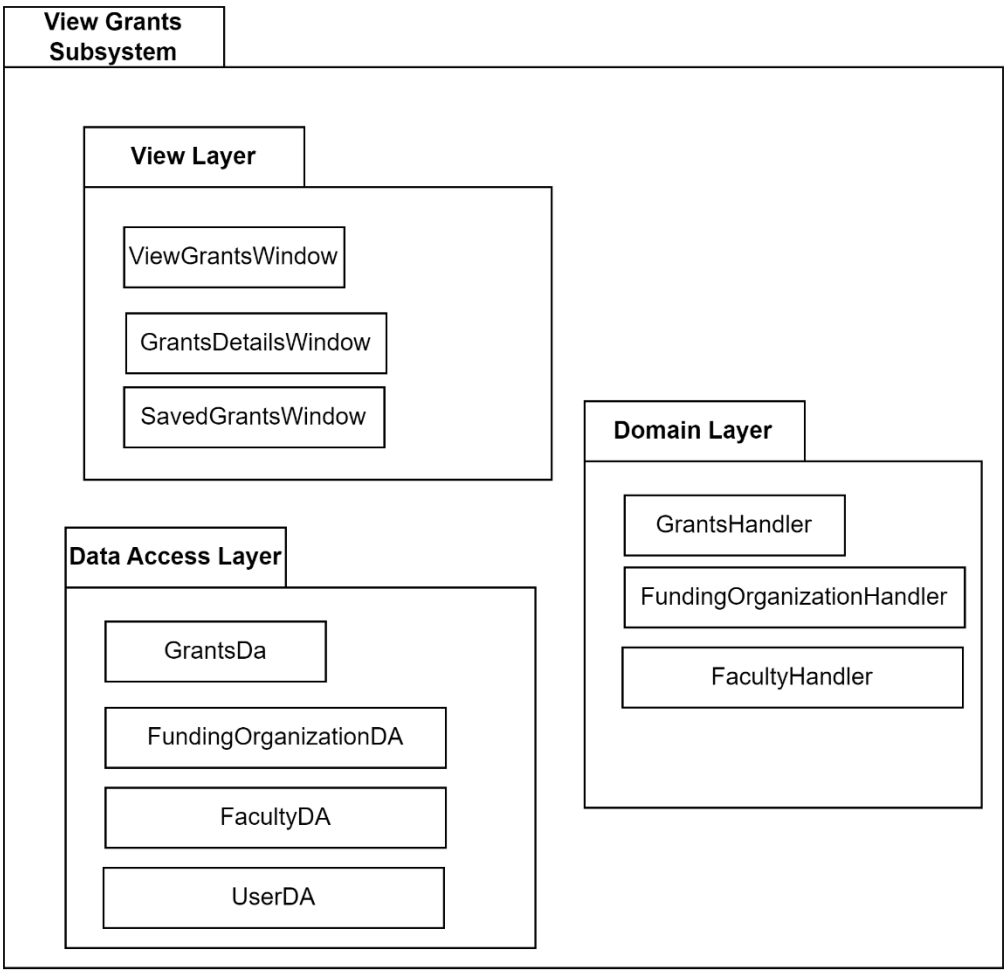


SD009: Sequence diagram for Sort & Filter Grants

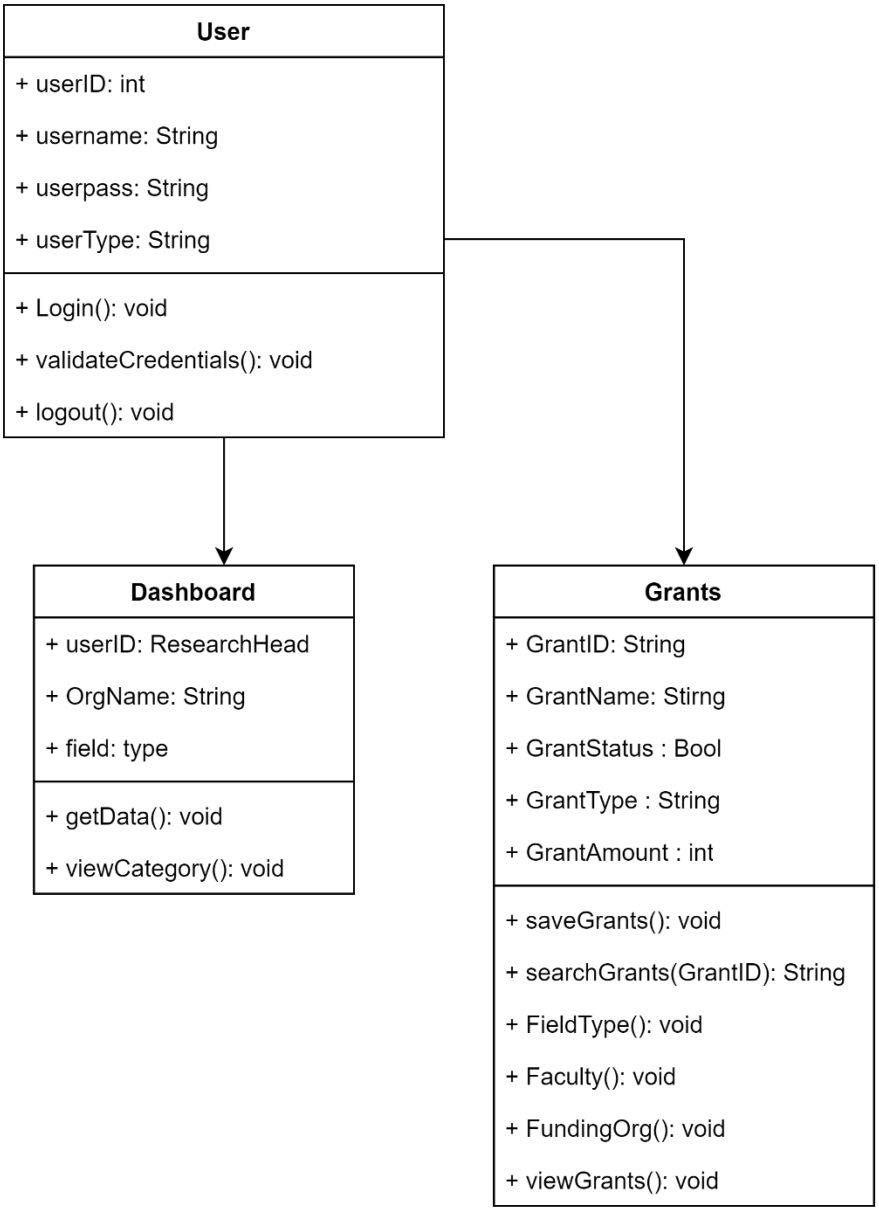

User
SearchPage
GrantsHandler
Database
FundingOrganizationHandler
FacultyHandler
chooseFilterByFundingOrg()
ApplyFilters()
RequestData()
FilterByFundingOrg()
chooseFilterByField()
ApplyFilters()
FilterByFaculty()
chooseFilterByDeadline()
ApplyFilters()
FilterbyDeadline()
returnFilteredData()
ViewData()
chooseSortByName()
applySorting()
sortByName()
ViewsortedData()
chooseSortBField()
applySorting()
sortByField()
ViewsortedData()
chooseSortBField()
applySorting()
sortByYear()
ViewsortedData()

3.2.6 Subsystem <View Grants>

3.2.6.1 P006: Package < View Grants >

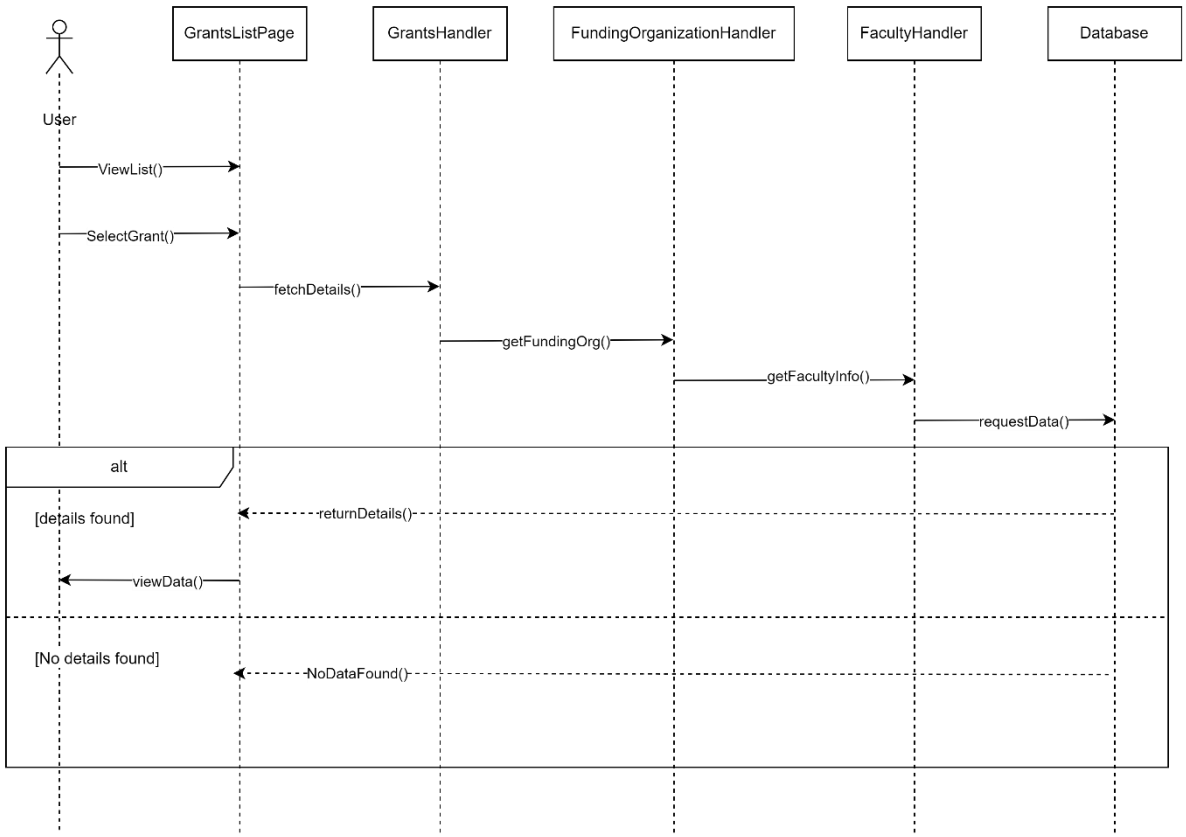


3.2.6.2 Class Diagram

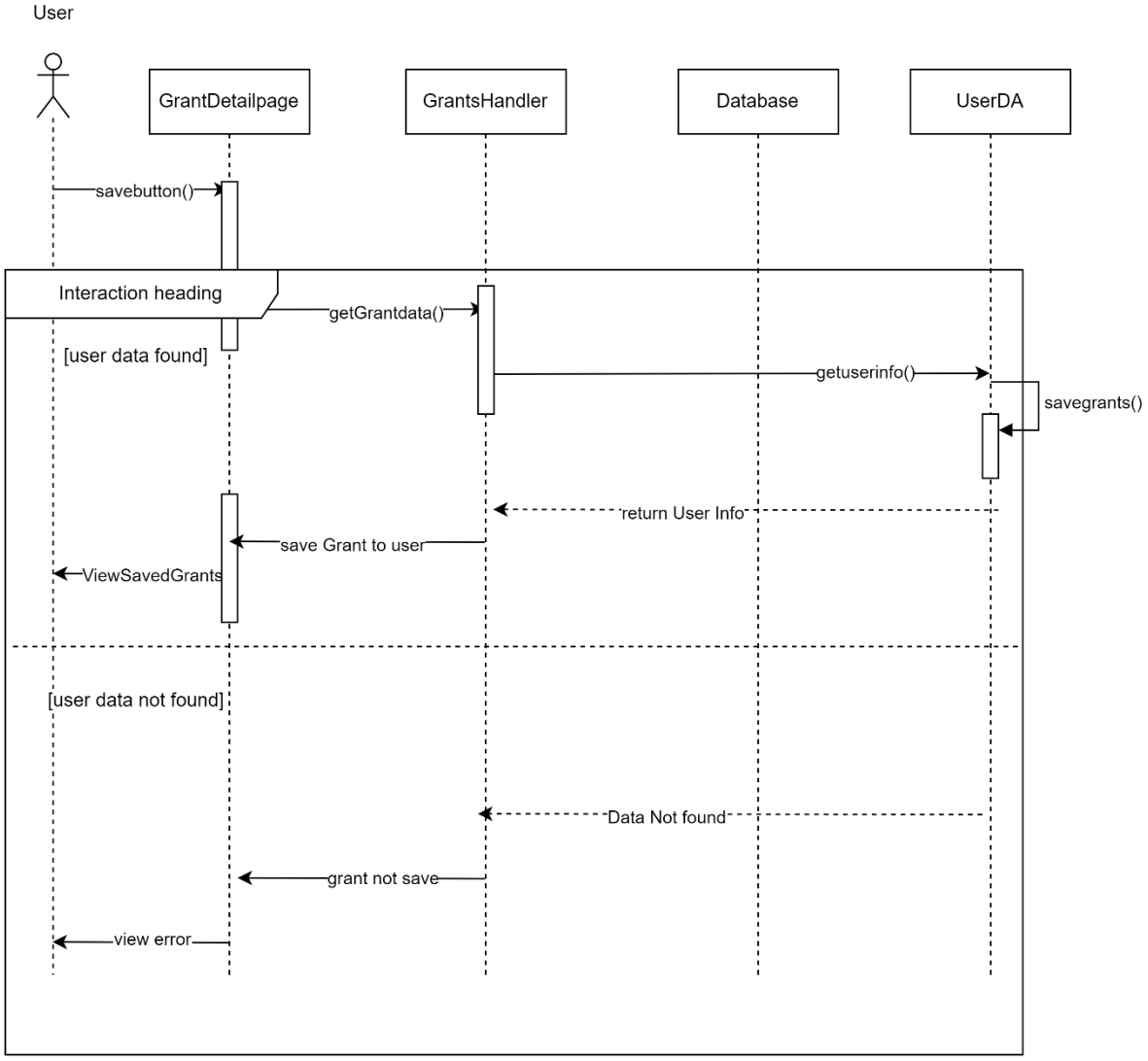


3.2.6.3 Sequence Diagrams

SD010: Sequence diagram for View Grant Details



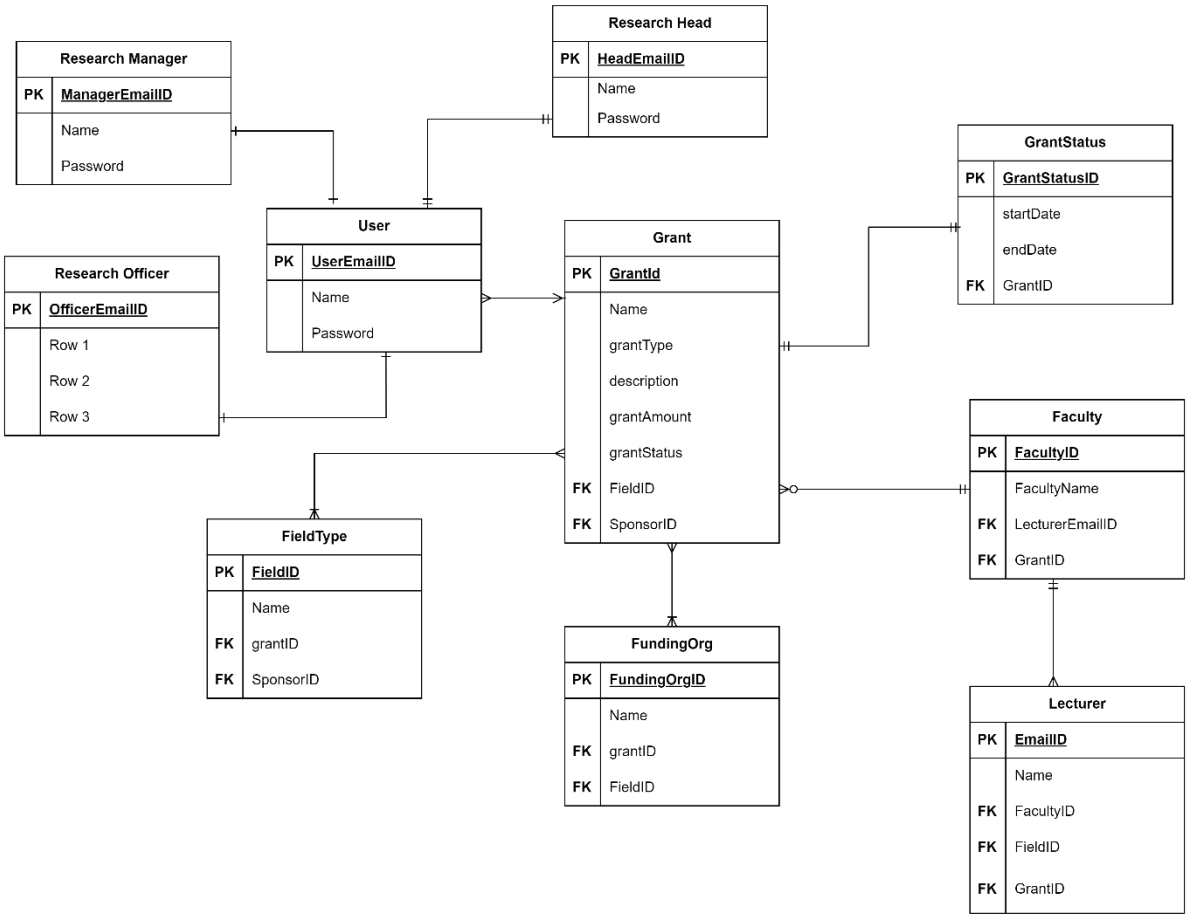
SD011: Sequence diagram for Save Grants


User
GrantDetailpage
GrantsHandler
Database
UserDA
savebutton()
Interaction heading
getGrantdata()
[user data found]
getuserinfo()
savegrants()
return User Info
save Grant to user
ViewSavedGrants
[user data not found]
Data Not found
grant not save
view error

4. DATA DESIGN

4.1 Data Description

The database design stores the information and system data. The information is safely kept in the system database. The figure below shows the database design of the system.



The data entities are described below:

Entity	Description
Research Admin	It stores the data of the research Manager such as email, username and password.
Grants	It stores the data of the grants that are extracted such as title, organization, release and expiry dates.
User	It stores the data of the Researcher
Field Type	It stores the data related to the field of research for grants.

Funding Organization	It stores the data of the funding organizations
Faculty	It stores the data of the faculty.
Grant Status	It stores the status of grants if it is ongoing or not.
Lecturer	It stores the data of the lecturer of the faculty.

4.2 Data Dictionary

Field Name	Datatype	Constraints	Description
Research Manager			
ManagerEmailID	String	Primary Key	Unique ID for research manager
Name	String	Not Null	Name of the research Manager
Password	String	Not Null	Password that the research manager uses to log into the system.
Research Head			
HeadEmailID	String	Primary Key	Unique ID for research head
Name	String	Not Null	Name of the research Head
Password	String	Not Null	Password that the research head uses to log into the system.
Research Officer			
OfficerEmailID	String	Primary Key	Unique ID for research officer

Name	String	Not Null	Name of the research officer
Password	String	Not Null	Password that the research officer uses to log into the system.
Field Type			
FieldID	Int	Primary Key	Unique ID for each field of research.
Name	String	Not Null	The name of the fields.
GrantID	Int	Foreign Key	The grantID to get the grants related to the field.
FundingOrgID	Int	Foreign Key	The FundingOrgID to get the sponsors related to the field.
Grant			
GrantID	Int	Primary Key	Unique ID for each grants.
Name	String	Not Null	The title of each grants.
grantType	String	Nullable	The type of grants.
Description	String	Nullable	The details of each grant.
grantAmount	Int	Not Null	The amount offered in each grants
grantStatus	Bool	Not Null	Determines if the grant is over or ongoing.

FieldID	Int	Foreign Key	Access the field of research related to grants
FundingOrganizationID	Int	Foreign Key	Access the funding organizations that provide grants.
Funding Organization			
FundingOrganizationID	Int	Primary Key	Unique ID for representing the funding organization.
Name	String	Not Null	Name of the funding organization
GrantID	Int	Foreign Key	Access the grant information that have the same funding organization.
FieldID	Int	Foreign Key	Access the number of fields an organization is funding.
Faculty			
FacultyID	Int	Primary Key	Unique ID for representing faculty.
FacultyName	String	Not Null	Provided the name of the faculty.
LecturerID	Int	Foreign Key	Access the lecturers in a faculty

GrantID	Int	Foreign Key	Access the grants by each faculty.
Grant Status			
GrantStatusID	Bool	Primary Key	Stores the true or false value if the grant exist or not.
startDate	Date	Not Null	The start date of the grant
endDate	Date	Not Null	The ending date of grants.
GrantID	Int	Foreign Key	Access the grant information to update the status.
Lecturer			
LecturerID		Primary Key	Unique ID to represent the lecturer of faculty.
Name	String	Nullable	Name of the lecturer.
FacultyID	Int	Foreign Key	Access the faculty details the lecturer belongs to.
FieldID	Int	Foreign Key	Access the field of research of the grants
GrantID	Int	Foreign Key	Access the grant information.

5. USER INTERFACE DESIGN

5.1 Overview of User Interface

The user interface is an essential part of the development of a project. With the help of user interfaces, a prior idea can be got how the system will look and function when it is developed, so in this way the flow of the system can be identified and if there is any flaw, it can be identified and improved. The Research Grant Finder system will have two different interfaces for the user and admin. Among the admin, there are 3 types who have specific limitations according to their designation.

5.2 Screen Images

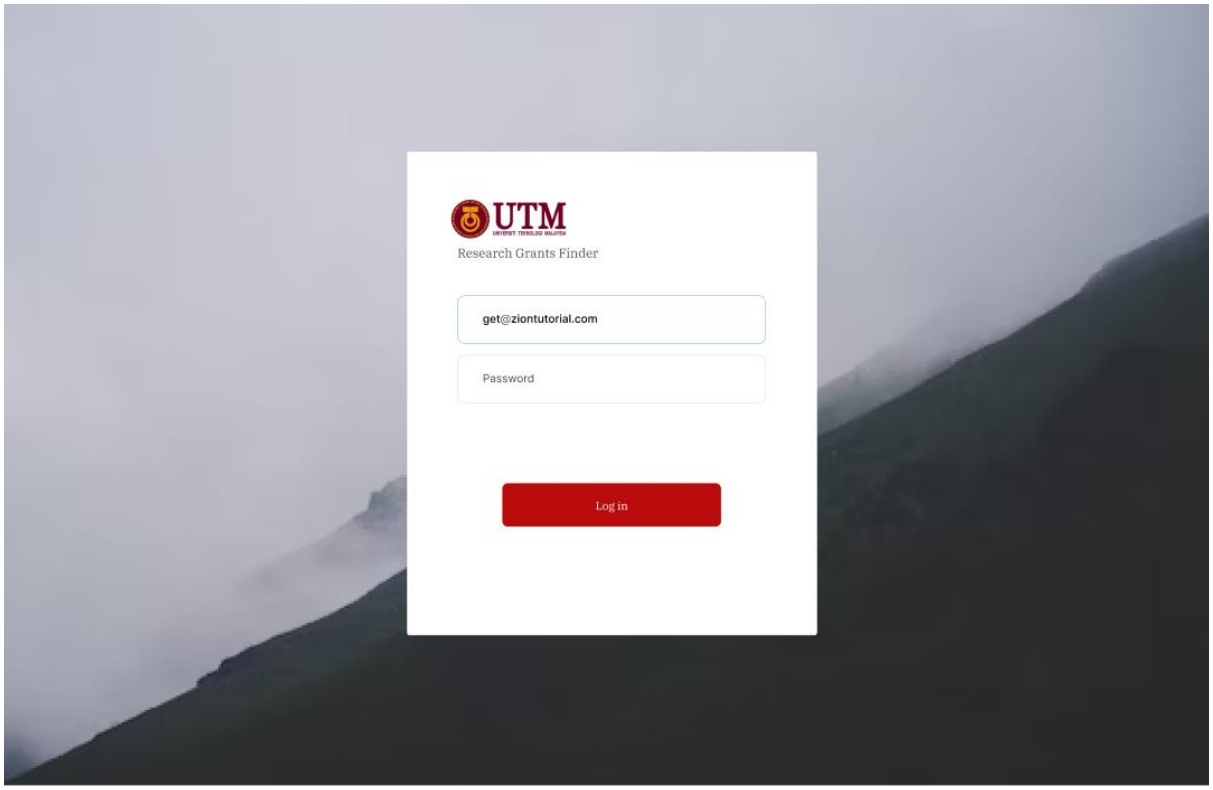


Fig: Login Screen

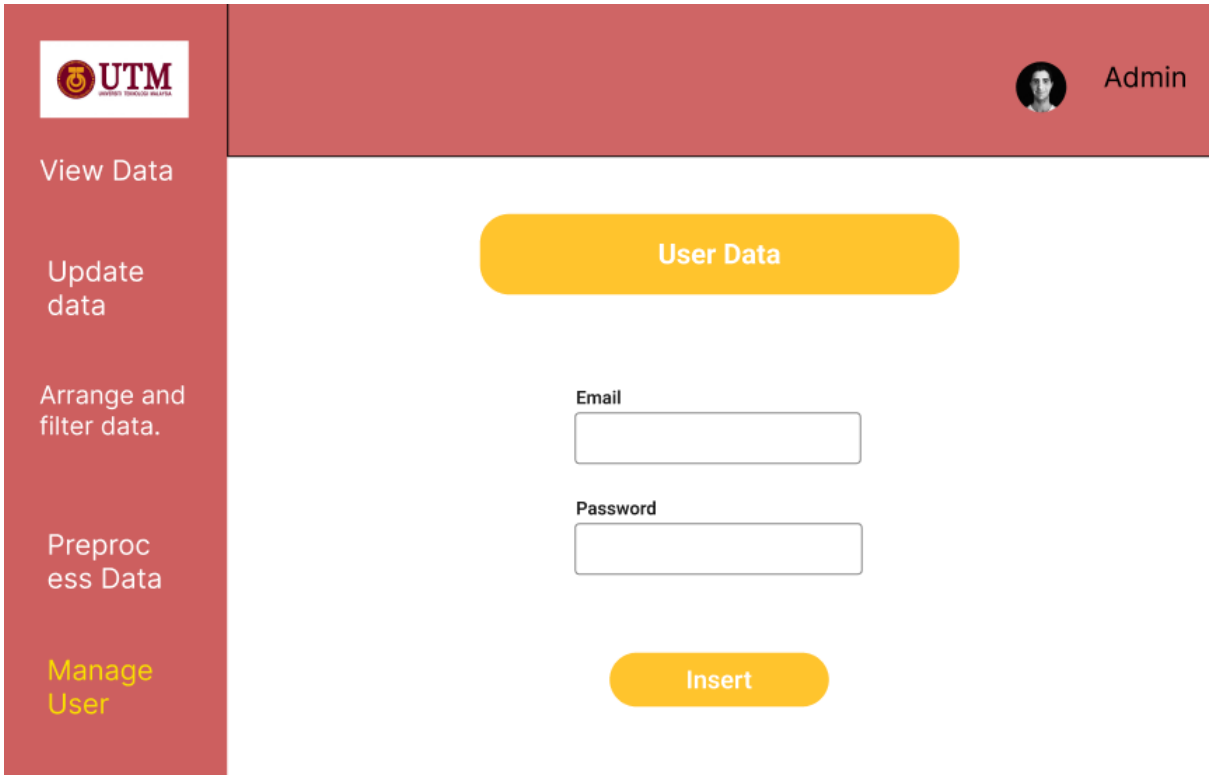
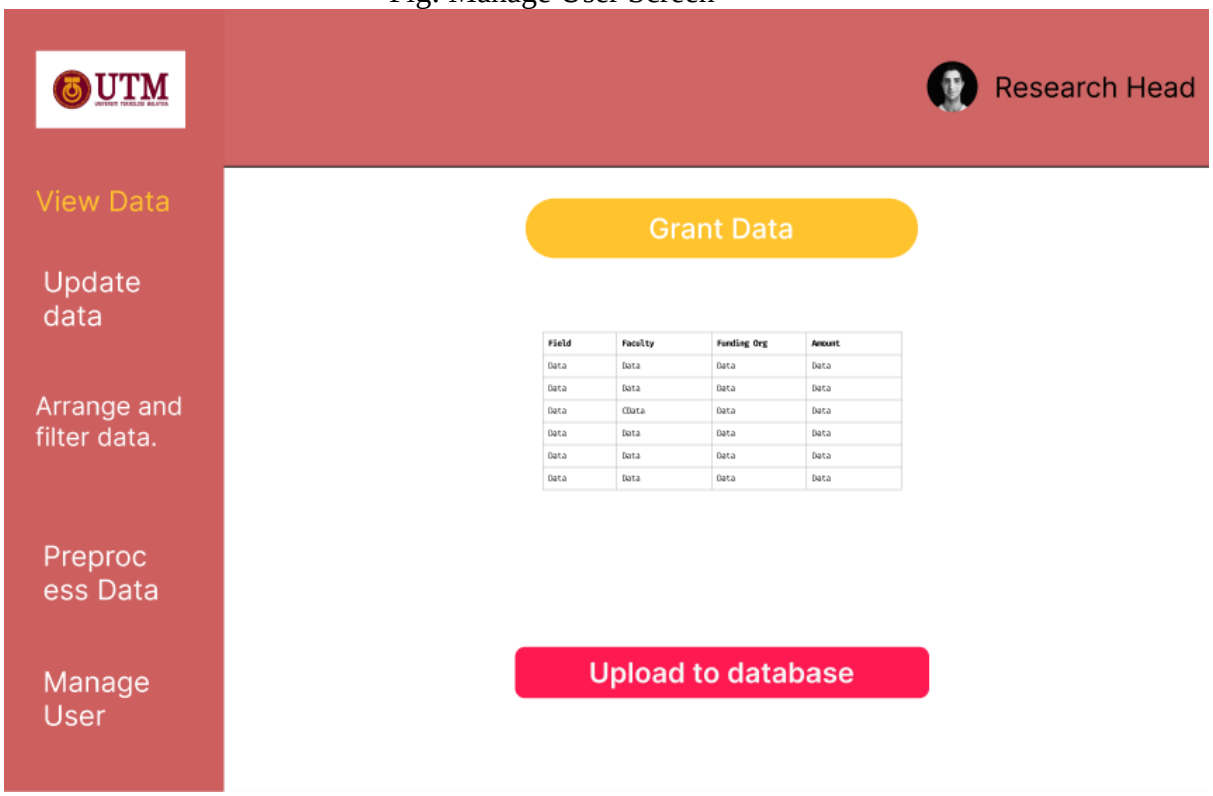
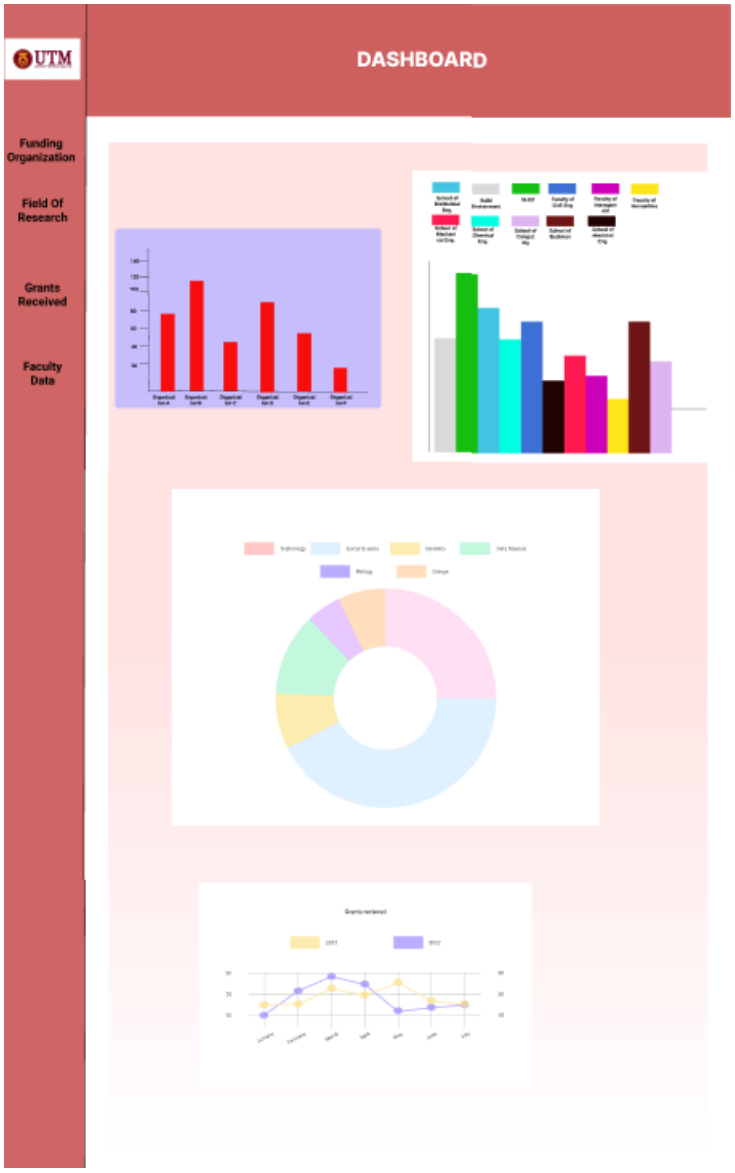


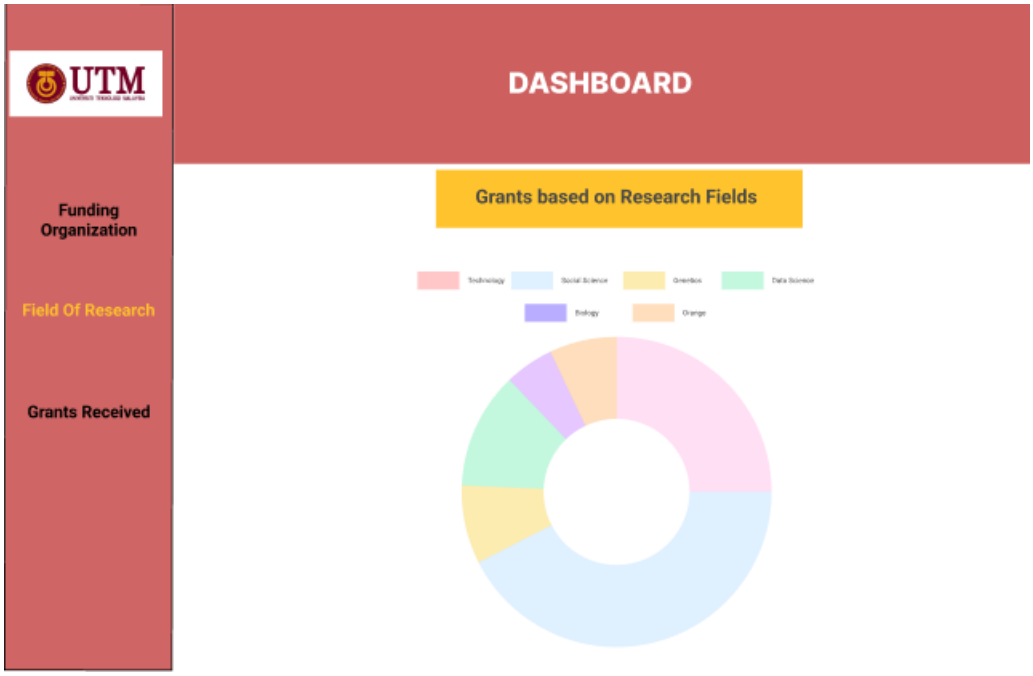
Fig: Manage User Screen

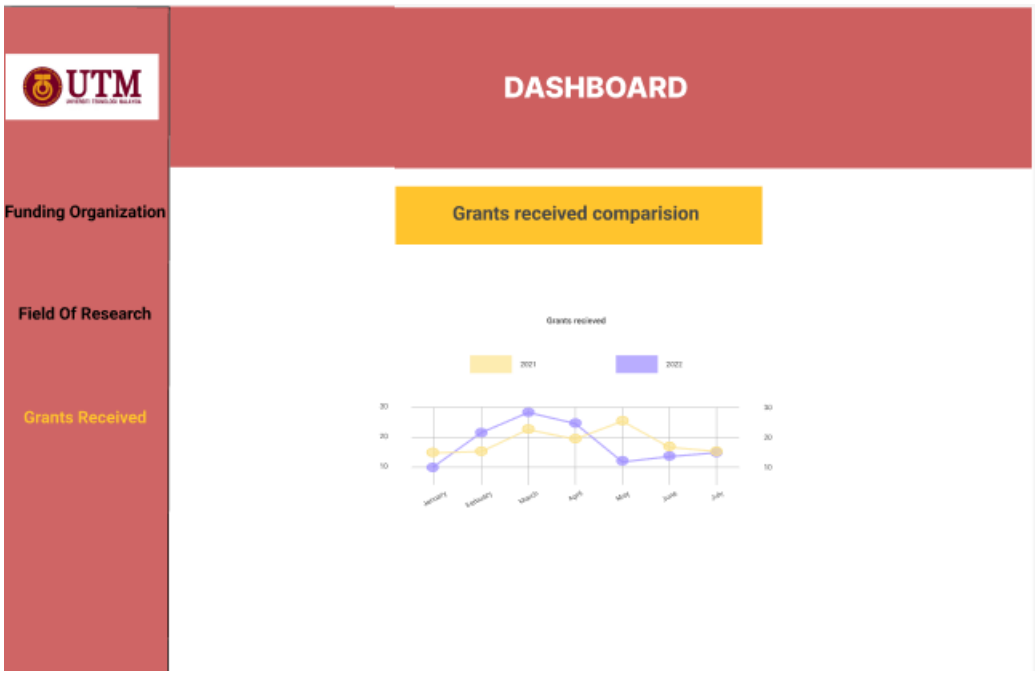


View data and upload to database


DASHBOARD
Funding Organization
Field Of Research
Grants Received
Faculty Data


DASHBOARD
Funding Orgabization
Field Of Research
Grants Received
Number of Grants provided by organizations
Organization Offering Highest Grant
Organization Offering least Grant
Organization F


DASHBOARD
Funding Organization
Field Of Research
Grants Received
Grants based on Research Fields


DASHBOARD
Funding Organization
Field Of Research
Grants Received
Grants received comparision

Dashboard

UTM

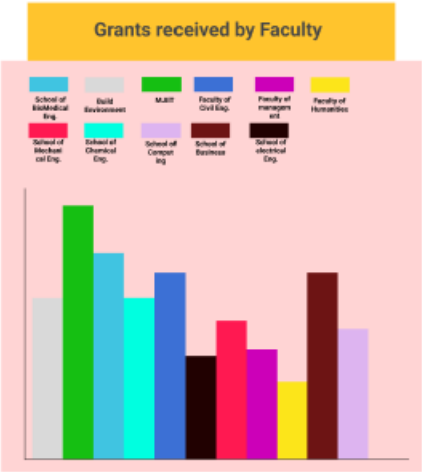
DASHBOARD

Funding Organization

Field Of Research

Grants Received

Faculty Data



6. REQUIREMENTS MATRIX

	P001	P002	P003	P004	P005	P006
UC001	X					
UC002		X				
UC002		X				
UC004		X				
UC005		X				
UC006			X			
UC007				X		
UC008					X	
UC009					X	
UC010						X
UC011						X

APPENDIX C: SOFTWARE TESTING DOCUMENT

Software Testing Documentation

RESEARCH GRANT FINDER

Version 1.0

25/06/2023

SCHOOL OF COMPUTING

Prepared by: Islam Mohammed Ruzhan

REVISION PAGE

i. Overview

Describe the content of the current version.

j. Target Audience

State the targeted audience.

k. Project Team Members

List the team members and respective assigned module.

l. Version Control History

Version	Primary Author(s)	Description of Version	Date Completed
<1.00>			

4. Introduction

Software testing is done once system development is complete. The main objective of this process is to convince both the user and the developer that the application is simple to use, capable of performing perfectly, and able to satisfy user needs. Additionally, through testing the software, faults can be found and fixed, allowing things should be improved. Software testing is essential for verifying the product's quality.

4.1 Purpose

This software testing documentation provides the necessary information about testing activities that include test action and expected results for each test cases of Research Grant Finder System.

4.2 Scope

The software product is Research Grant Finder System which visualizes the grant data to the users and also enables them to search for grants from the repository. Testing covers the application's functional requirements to ensure that there are no errors or flaws.

4.3 Definitions, Acronyms and Abbreviation

Term	Description
STD	Software Testing Documentation
FC	Faculty of Computing

4.4 References

i. *Software testing - documentation Online Courses and eBooks Library*. Available at: https://www.tutorialspoint.com/software_testing/software_testing_documentation.htm (Accessed: 25 June 2024).

4.5 System Overview

This SDD document contains the test cases to be executed during the testing phase of the Research Grant Finder System.

5. TEST CASES, DATA AND EXPECTED RESULTS

5.1 Test TC001 for Module <Authentication>: <Login (UC001)>

.

This test contains the following test cases:

UC001_01: Successful login

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the login page.	-	System displays the login page.	The login page is displayed	Pass
2	The user enters the correct user details	Email: abdur@gmail.com Password: a@34ruz	The system successfully logs in according to the user credentials.	Login successful	Pass
3	The user clicks on the login button.	-	The system verifies the user successfully.	Only verified users can access the system	Pass

UC001_02: Invalid Login details

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the login page.	-	System displays the login page.	The login page is displayed	Pass
2	The user enters the details	Email: abdur2 Password: a@34ruz	The system displays the user input.	Successful display of user input	Pass
3	The user clicks on the login button.	-	The system displays error messages.	Error message is displayed as alert	Pass

UC001_03: Incorrect Login details

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the login page.	-	System displays the login page.	The login page is displayed	Pass
2	The user enters the details that is not in the database	Email: ruz@gmail.com Password: a@34ruz	The system displays the user input.	Successful display of user input	Pass

3	The user clicks on the login button.	-	The system displays error message because the data did not match with the database.	Error message is displayed for bad credentials	Pass

5.2 Test TC002 for Module<Manage Data>: <Manage User Data (UC002)>

This test contains the following test cases:

UC002_04: Input valid user details while adding users.

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The admin enters the create user page.	-	System displays the create user page.	The create user page is displayed	Pass
2	The admin enters the details to create a user account.	Email: uaben@gmail.com Password: a@34ruz	The system displays the input.	Successful display of user input	Pass
3	The admin clicks on the add user button.	-	The system waits for confirmation.	Successful user creation	Pass

UC002_05: Input invalid user details while adding users.

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The admin enters the create user page.	-	System displays the create user page.	The create user page is displayed	Pass
2	The admin enters the invalid details to create a user account.	Email: uaben123 Password: a@34ruz	The system displays the input.	Successful display of user input	Pass
3	The admin clicks on the add user button.	-	The system displays error.	Error message is displayed for wrong format	Pass

UC002_06: Successfully view the user data

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The admin enters the User list page.	-	System displays the User list page.	The create user list page is displayed	Pass
2	The admin clicks on view users button	-	The system views the user data.	Admin can view the list of users	Pass

UC002_07: Unsuccessful to view the user data

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The admin enters the User list page.	-	System displays the User list page.	The create user list page is displayed	Pass
2	The admin clicks on view users button	-	The system is unable to view the data	Error message displayed	Pass

5.3 Test TC003 for Module< Manage Data >: <Arrange and filter data (UC004)>

UC004_01: Successfully categorize the data.

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The scrapping module initiates the scrapping	-	The web scrapping of the websites starts.	The loading widget starts indicating the start of scrapping.	Pass
2	The scrapping module access the query selectors and the JavaScript contents of the website and get hold of the targeted selectors.	-	The required tags and selectors are selected and accessed by the module.	The grants count increases indicating the scraping is ongoing.	Pass
3	The scrapping module gets hold of the applied filters.	-	The filters are applied to the website.	Successful application of filters	Pass
4	The scrapping module returns the filtered data.	-	Filtered data are stored in the module.	Data are stored in an array	Pass

UC004_02: Successfully access the data.

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The Scrapping module gets access to the target website.	-	The website is opened in an automated browser.	The website opens in an automated browser	Pass
2	The module gets access to the query selectors that contain the data.	-	The scrapping is initialized.	Initializes scrapping	Pass
3	The module gets access to the data.	-	The filtering process is initialized.	The scrapping initializes the filtering	Pass

5.4 Test TC004 for Module< Manage Scraping >: < Manage Scraping (UC006)>

UC006_01: Successfully modify the scraping

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The research admin enters the scrapping page.	-	System displays the scrapping page.	The scrapping page is displayed	Pass
2	The research admin clicks on NIH Scrape button.	-	The NIH scrapper is executed.	The scrapping of NIH is started	Pass
3	The research admin clicks on Grants Gov Scrape button.	-	The Grants Gov scrapper is executed.	The scrapping of Grants Gov is started	Pass

UC006_02: Unsuccessful to modify the scraping

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The research admin enters the scrapping page.	-	System displays the scrapping page.	The scrapping page is displayed	Pass

2	The research admin clicks on NIH Scrape button.	-	The system displays error message at NIH scrapping	Error message is displayed as an alert when the NIH button is clicked	Pass
3	The research admin clicks on Grants Gov Scrape button.	-	The system displays error message at Grants Gov scrapping	Error message is displayed as an alert when the Grants Gov button is clicked	Pass

5.5 Test TC005 for Module< View Dashboard >: < View Dashboard (UC007)>

UC007_01: Successfully view the dashboard

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the system	-	The system fetches the dashboard data and view the dashboard.	The dashboard can be viewed	Pass

UC007_02: Unsuccessful to view the dashboard

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the system	-	The system shows error message.	Error message displayed to reload the dashboard	Pass

5.6 Test TC006 for Module< Search Grants >: < Search for grants(UC008)>

UC008_01: Successfully find grants.

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the grants list page.	-	System displays the search bar.	Search bar displayed successfully	Pass

2	The user enters the keyword	-	The system displays the keyword.	Users input is displayed correctly	Pass
3	The user clicks on the search button.	-	System retrieves grants that match the keywords	System filters according to search results	Pass

UC008_02: Unsuccessful to find grants.

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the grants list page.	-	System displays the search page.	Search bar displayed successfully	Pass
2	The user enters the keyword	computer	The system displays the keyword.	Users input is displayed correctly	Pass
3	The user clicks on the search button.	-	System does not find any search results.	No data found message is shown	Pass

5.7 Test TC006 for Module< Search Grants >: < Sort and filter grants(UC009)>

UC009_01: Sort and filter grants.

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the grants list page.	-	System displays the filter page.	The filtering tools are displayed in the grants list page	Pass
2	The user chooses the filters	-	The filters are applied	The data is sorted according to the filters	Pass

5.8 Test TC007 for Module< View Grants>: < View Grant details(UC0010)>

UC010_01: View Grant details

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user clicks on the grants obtained from search	-	The system displays the	The user can view the grants	Pass

			grants details page.		

UC010_02: Unable to view Grant details

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user clicks on the grants obtained from search	-	The system shows error.	Error message is shown that the unable to fetch grants	Pass

5.9 Test TC007 for Module< View Grants>: < Save Grants(UC011)>

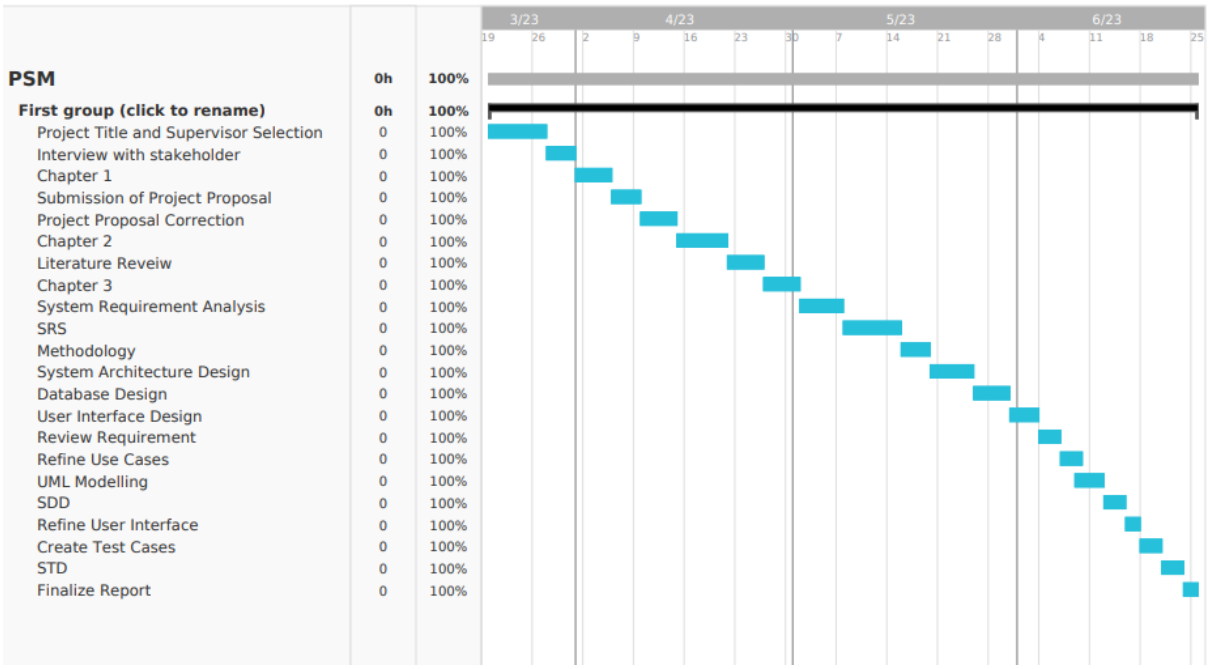
UC011_01: Save Grants

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the details page.	-	The system displays the grants details page.		
2	The user clicks on the save grants button.	-	The system saves the grant to the user database		

UC011_02: Unable to Save Grants

No.	Action	Input Data	Expected Result	Actual Result	Pass/Fail
1	The user enters the details page.	-	The system displays the grants details page.		
2	The user clicks on the save grants button.	-	The system displays error message and does not save the grant		

Appendix D: Gantt Chart


3/23
4/23
5/23
6/23
19
26
2
9
16
23
30
7
14
21
28
4
11
18
25
PSM 0h 100%
First group (click to rename) 0h 100%
Project Title and Supervisor Selection 0 100%
Interview with stakeholder 0 100%
Chapter 1 0 100%
Submission of Project Proposal 0 100%
Project Proposal Correction 0 100%
Chapter 2 0 100%
Literature Reveiw 0 100%
Chapter 3 0 100%
System Requirement Analysis 0 100%
SRS 0 100%
Methodology 0 100%
System Architecture Design 0 100%
Database Design 0 100%
User Interface Design 0 100%
Review Requirement 0 100%
Refine Use Cases 0 100%
UML Modelling 0 100%
SDD 0 100%
Refine User Interface 0 100%
Create Test Cases 0 100%
STD 0 100%
Finalize Report 0 100%