## 

## 

## **CSU33013/CSU22012 Group 34 Requirements Document**

## **College Course Rating App**

## **Client : Michael Noonan**

## **Authors** Cian Galligan 18327898 John Sinclair 16325734 Ciara Lynch 19336403 Cian O’Gorman 19334475 Honglin Li 19315272

## 

## **Table of Contents**

## **Introduction**

# 1.1. Overview & Purpose

# 1.2. Scope

# 1.3. Objectives & Success Criteria

# 1.4. Definitions & Abbreviations

# 1.5. References

## **2. Proposed System**

# 2.1. Overview & Purpose

# 2.2. Functional Requirements

# 2.3. Non-Functional Requirements

# 2.4. System Prototype (models)

# 2.4.1 User Interface Mockup

# 2.4.2 UML Use Case Diagram

# 2.4.3 Object Model

# 2.4.4 Dynamic Model

# 

# 

# 

# 

# 

# 

# 

# Introduction

## 1.1 Overview & Purpose

The purpose of this project is to create the ‘glassdoor’ of college courses. We aim to create a web application that shows users rankings and relevant information on college courses they are interested in.

The app users would be able to rate college courses, see graduate opportunities, and examine how much the course would cost and also how much it would cost them per hour. Users would also have the ability to add their own reviews on college courses they have attended.

## 1.2 Scope

The following areas are in the scope of our web application:

* The creation of a college course rating system for Trinity postgraduate courses. For our project we have limited our scope to Trinity postgraduate courses as it allows the project to have a high quality, accurate dataset.

The following areas are out of scope:

* College courses outside of Trinity
* Undergraduate courses in Trinity

## 1.3 Objectives & Success Criteria

Objectives:

* Create a web application using the React framework that will elucidate the course picking process. The app will assist users in selecting the postgraduate course that best suits them by providing information such as the courses overall rating, description, a fees breakdown (cost per hour), student reviews with information on graduate paths, salary information and further career progression.

Success criteria:

* Implement a clean and effective frontend the user can interact with using the React framework.
* Implement a firebase backend that stores course, user and loan information.

1.4 Definitions & Abbreviations

Frontend - the part of the software system the user interacts with directly

Backend - the part of the system that is not accessed by the user directly, used to store and manipulate data

React - a javascript library for building user interfaces

Firebase - google's platform for creating mobile and web applications - provides user authentication and backend capabilities.

1.5 References

<https://firebase.google.com/>

<https://reactjs.org/>

<https://tailwindcss.com/docs>

Proposed System

## 2.1 Overview & Purpose

A concise and accessible web application that allows users to

* Review and read ratings of specific postgraduate courses
* View the salaries and job positions of previous graduates from the different postgraduate courses
* Outlines the cost per hour required from a student to complete a particular postgraduate course

## 2.2 Functional Requirements The system should be able to do the following:

## User Requirements:

## Sign up and create an account using their TCD email.

* Sign into their account.
* Search for different courses within the TCD domain.
* Rate different modules in their course on a scale from 1-5.
* Write reviews for different modules and courses.

System Requirements:

* Display name of the course and it’s modules.
* Display an average star rating for each course.
* Display the amount of hours involved for each course/module.
* Display a short description for each module if added.
* Display the amount of hours per course.
* Display a short course description if added.
* Display loan information from various banks.
* Display the cost per hour of a course.

Our client regularly referred to this project as “Glassdoor except for Colleges” so we are roughly basing our project on the Glassdoor website, as well as adding our own additional features to better tailor the web application as needed.

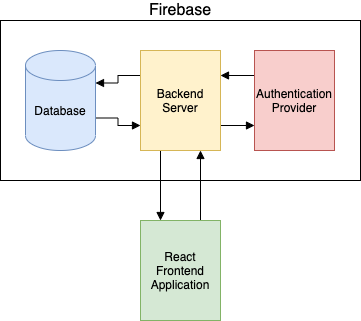
2.3 Non-functional Requirements

The system should be:

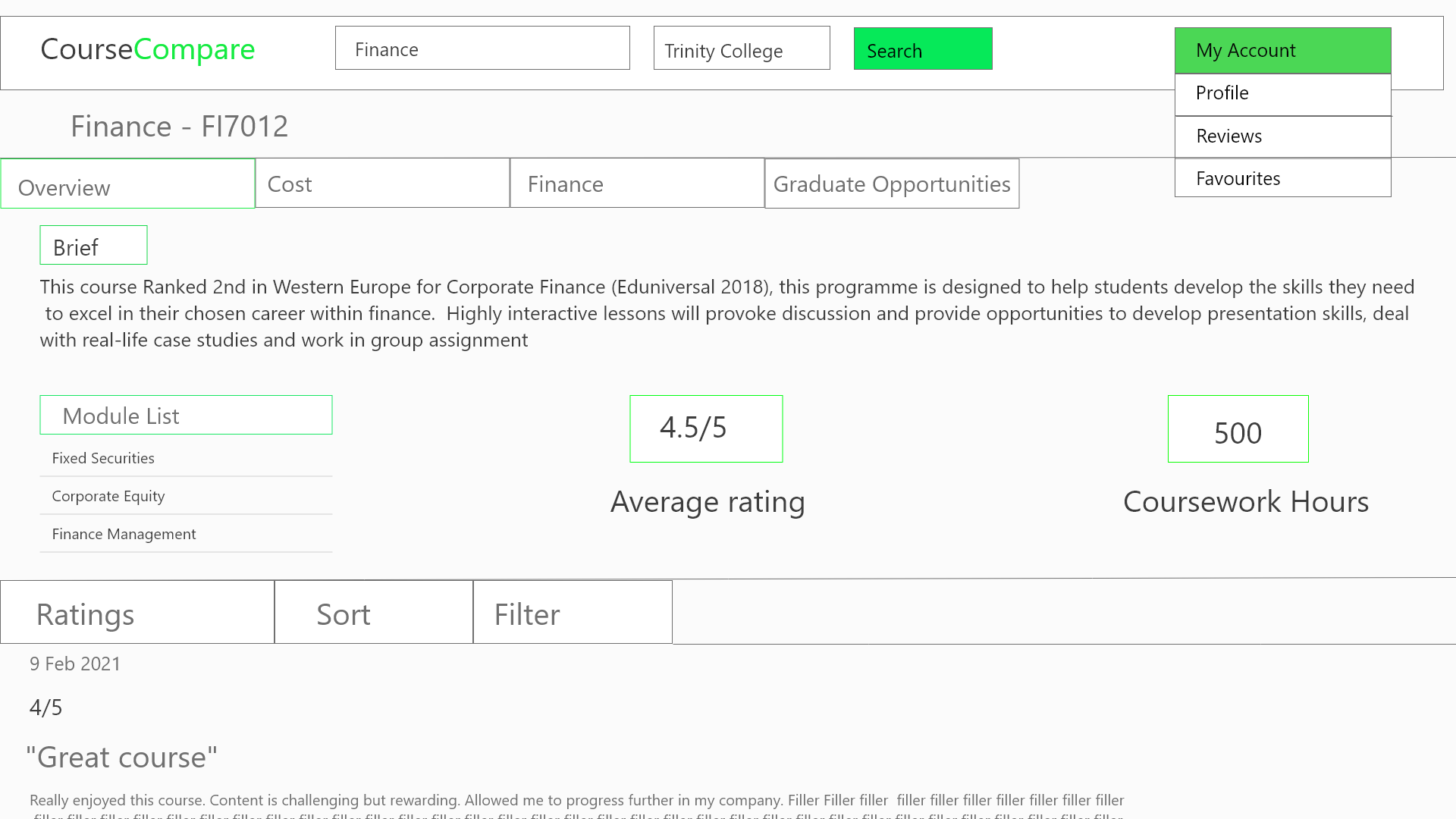
* Concise: The interface and pages should clearly display all the information the user requires
* Easy to use: The application should have a consistent layout that is intuitive for the user
* Scalable: To ensure that the system does not become redundant, it must be able to contain and display numerous postgraduate courses as they become available
* Recoverability: In the event of any crash the information and reviews inputted by the users must have the ability to be easily retrieved for redeployment (Will be using firebase, which is hosted by google so is low risk to the above stated possibility)
* Capacity: The system must be able to host and contain a large amount of information pertaining to each postgraduate course and the user input

2.4 System Prototype

This diagram displays the interaction that will take place between the Reactjs frontend application and the firebase backend. Firebase was chosen due to its simplicity and highly scalable nature. It stores data in the form of key value pairs bundled into so called documents. Firebase also provides authentication via google, perfect for our application as our scope limits us to students of Trinity College Dublin, all of whom have google accounts.



2.4.1 User Interface Mockups



**The features present in this user interface mockup are:**

* A search function for college courses
* Module list of college course
* Average rating of college course
* Coursework hours for college course
* Reviews of college course
* Sort and filter features for reviews
* Sidebar for viewing account details and favourite courses

To view more information about the college course such as loans, graduate opportunities and costs, users can toggle between the four tabs (overview, cost, finance, graduate opportunities).

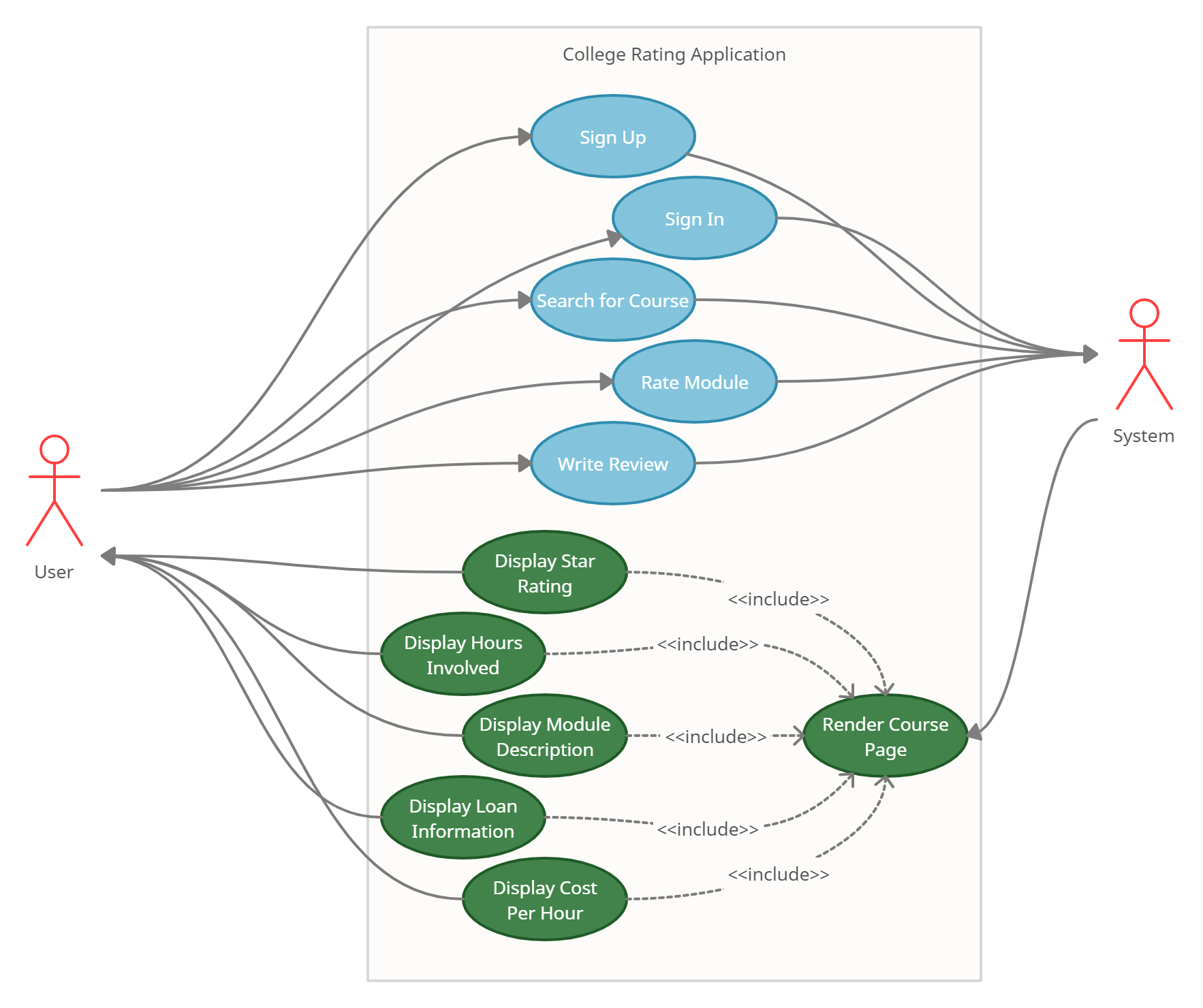
2.4.2 UML Use Case Diagram

There are two actors in the College Rating Application System. The two actors are the intended user of the application and the system itself which is running the college rating application.

The user is able to perform multiple actions on the college rating application, they are able to make an account and sign into it. The user also is able to search, rate and review courses and their respective modules.

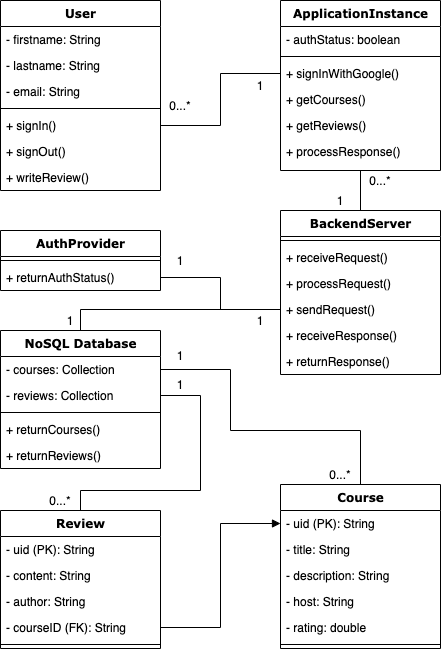
All actions performed by the user are handled by the system. The information received from the user is stored to the backend of the system which then displays the user interface with the newly updated information.

The system to display the newly received information must render the data to the front end. This results in the specified data being rendered to the screen for the user.



2.4.3 Object Model

The diagram below shows the software system in terms of objects. From this diagram we can see how the user will only interact with the Frontend Application Instance, this instance will then make calls to the Backend Server which will process the requests, providing authentication and database services. using the AuthProvider before then returning the requested information from the Firebase Database.



2.4.4 Dynamic Model

The dynamic model for college rating app is as follows. This particular sequence is where the system shows sign in page allowing the user to sign in. After sign-in authorisation by the system, the user can then search for courses, and add comments and ratings to the courses selected. After the user’s submission, the system subsequently updates its course information. The user may sign out in the end.

