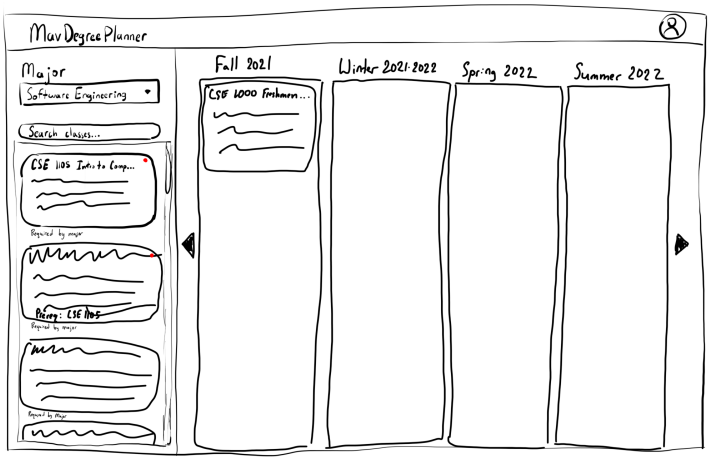
Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specification
CSE 4316: Senior Design I
Fall 2021



Team IDP
Interactive Degree Planner

Daniel Newville
Le Uyen Nguyen
Ijaz Mohamed Umar
Safi Ullah

Revision History

Revision	Date	Author(s)	Description
0.1	12.04.2021	DN	document creation
0.2	12.10.2021	Group	document completion
0.3	05.07.2021	Group	document revision

CONTENTS

List of Figures

List of Tables

1 INTRODUCTION

Degree planning is never an easy task. It requires students to have a thorough understanding about the course information, description, pre-/co-requisites, and etc. Any mistake or carelessness during this process may result in wasted time, money, effort, and negative impact on goal accomplishment. Therefore, Interactive Degree Planner is developed with a mission to facilitate the degree planning process for students in the Department of Computer Science and Engineering of the University of Texas ar Arlington.

1.1 PURPOSE AND USE

Interactive Degree Planner is a web-based application that is expected to simplify the process of planning an academic roadmap. The user should be able to log-in, choose the desired major in the CSE department, and start building his/her own degree planner. The website will abstract away all the complexity during this process, such as pre- and co-requisites, lookup, and etc. Besides, it takes advantage of drag-and-drop function in order to provide a user-friendly interface.

1.2 INTENDED AUDIENCE

The website aims to provide services for current and/or future CSE students at the University of Texas at Arlington.

2 System Overview

This section should describe the overall structure of your software system. Think of it as the strategy for how you will build the system. An architectural "layer" is the top-level logical view, or an abstraction, of your design. Layers should be composed of related elements of similar capabilities, and should be highly independent of other layers, but should have very clearly defined interfaces and interactions with other layers. Each layer should be identified individually and should be unique as to its function and purpose within the system. This section should also contain the high-level block diagram of the layers, as shown in the example below, as well as detailed descriptions of the functions of each layer.

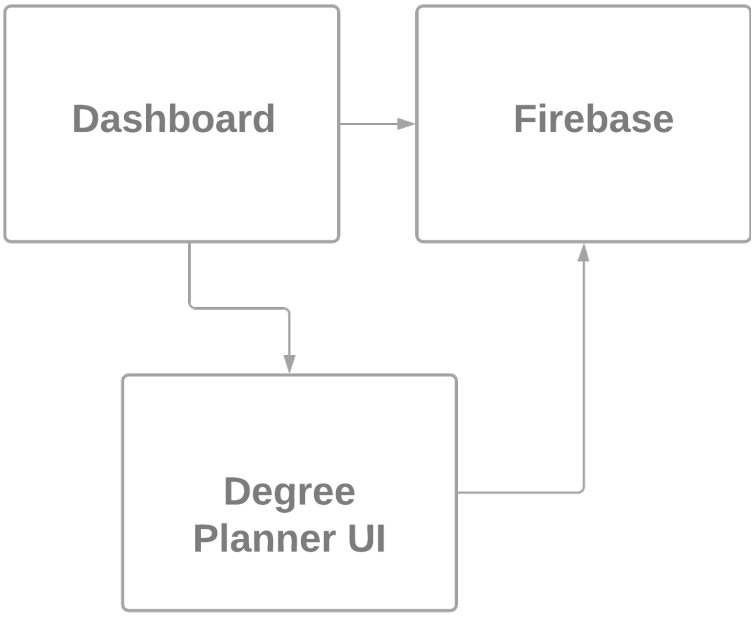


Figure 1: A simple architectural layer diagram

2.1 Degree Planner UI Description

Each layer should be described separately in detail. Descriptions should include the features, functions, critical interfaces and interactions of the layer. The description should clearly define the services that the layer provides. Also include any conventions that your team will use in describing the structure: naming conventions for layers, subsystems, modules, and data flows; interface specifications; how layers and subsystems are defined; etc.

The degree planner UI will have features allowing users to easily select and interact with their degree plans. It will allow users to select classes, display information, and give the user access to navigate between the Dashboard and the degree planning section. The UI will interact with Firebase when the student selects classes. It will get the data from Firebase and bring it to the UI layer. Once the UI has the data, it will then display the data to the users. The UI will also allow the users to navigate between screens and give them access to switch to the Dashboard screen. We will call this the Degree Planner UI layer.

2.2 Firebase Description

Each layer should be described separately in detail. Descriptions should include the features, functions, critical interfaces and interactions of the layer. The description should clearly define the services that the layer provides. Also include any conventions that your team will use in describing the structure: naming conventions for layers, subsystems, modules, and data flows; interface specifications; how layers and subsystems are defined; etc.

The firebase layer has features such as keeping data, authentication, and sending data. When a user selects classes the UI layer will request the data from firebase. Firebase will

2.3 Dashboard Description

Each layer should be described separately in detail. Descriptions should include the features, functions, critical interfaces and interactions of the layer. The description should clearly define the services that the layer provides. Also include any conventions that your team will use in describing the structure: naming conventions for layers, subsystems, modules, and data flows; interface specifications; how layers and subsystems are defined; etc.

The dashboard layer of the degree planner will have a link to the degree planner, it will allow the user to sign in and out of their account, and it will let the user see the flowchart of their major. The user will be able to make an account in the dashboard. They will give their information and be able to make an account. Once their account is made, their data will be sent over and stored in firebase. They can then sign in and out with the use of the dashboard and everytime they sign in and out. The authentication will be handled through firebase. In the dashboard there will also be the option for the user to click on a link to see the flowchart of their major. This layer will be called the dashboard layer.

3 SUBSYSTEM DEFINITIONS & DATA FLOW

This section breaks down your layer abstraction to another level of detail. Here you grapically represent the logical subsytems that compose each layer and show the interactions/interfaces between those subsystems. A subsystem can be thought of as a programming unit that implements one of the major functions of the layer. It, therefore, has data elements that serve as source/sinks for other subsystems. The logical data elements that flow between subsystems need to be explicitly defined at this point, beginning with a data flow-like diagram based on the block diagram.

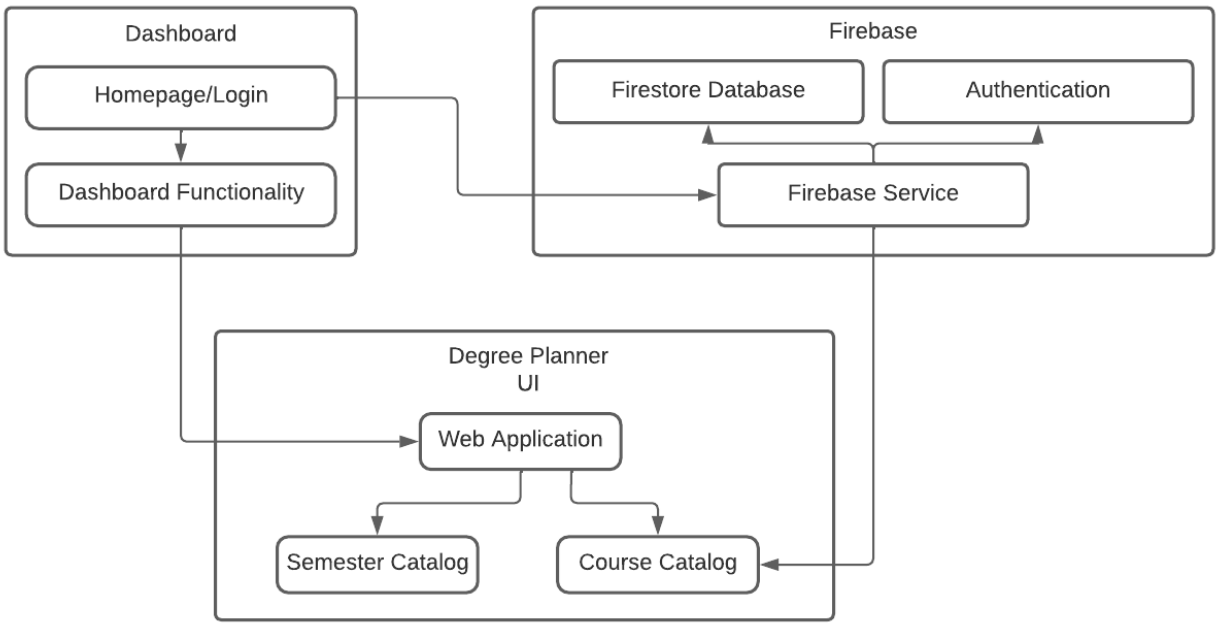


Figure 2: A simple data flow diagram

4 Degree Planner UI Layer

The Degree Planner UI layer consists of Web application, Course catalog, and Semester catalog. The main functionality of this layer is to provide the user interface, where the user is able to prepare his/her degree plan and view the related information. Among the above-mentioned subsystems, the Web application handles the most of the task, while the rest support

4.1 Web Application

Web application subsystem provides the user interface for the process of planning degree. The two fundamental goal of this layer is to display information and to interact with user.

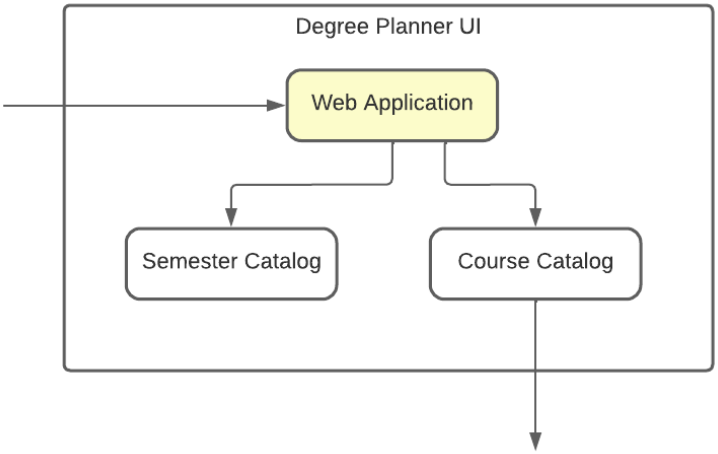


Figure 3: Web Application subsystem

4.1.1 Assumptions

- The user must have Internet connection to use the service.
- The subsystem is only available for users authenticated in Dashboard layer.
- The subsystem does not provide any functionality to modify data and no data changes should be made in this subsystem.

4.1.2 Responsibilities

This subsystem takes responsibility to display the required information, including: the user's major, list of courses for the corresponding major, list of semesters, total hours per semester, and etc. All data fetched from external sources are assumed to be up-to-date and cannot be modified in this layer. The user will perform planning tasks, such as drag and drop to add and/or remove courses. That is, the list of course objects from Course catalog subsystem will be made draggable, while the area/column for each available semester will become droppable. Besides, there will be navigation options to link to Export subsystem.

4.1.3 Subsystem Interfaces

Table 2: Web Application subsystem interfaces

ID	Description	Inputs	Outputs
#1	Choose major	Option	Screen
#2	Display courses	Object of every element in the list of courses provided by Course catalog subsystem	Screen
#3	Search for courses	Text Numbers	Object of the searched course
#4	Display semesters	List of semesters	Screen

4.2 Course Catalog

Course catalog subsystem acts as the interface between the Degree Planner UI layer and Firebase database layer. It will fetch data, i.e., the list of courses, from the database, and sends it to the Web application subsystem.

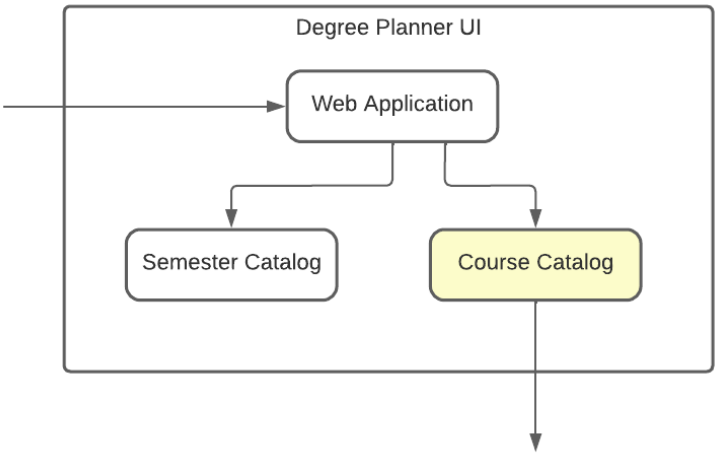


Figure 4: Course Catalog subsystem

4.2.1 Assumptions

- The user must have Internet connection.
- The subsystem gets the raw data from the Firebase layer.
- The subsystem fetches the up-to-date required data from the Firebase layer.
- The subsystem cannot modify data fetched from the Firebase database. It does not provide any functionality to modify data and no data changes should be made in this subsystem.

4.2.2 Responsibilities

This subsystem will take input, i.e., major, from the web application subsystem, and fetch the corresponding list of courses. Since the database only provides raw data, this subsystem will process each query to create a course object. The list of course objects will then pass to the Web application subsystem.

4.2.3 Subsystem Interfaces

Table 3: Course Catalog subsystem interfaces

ID	Description	Inputs	Outputs
#1	Fetch data	Text	Database response with the list of courses from selected major
#2	Process data	Queries	List of objects
#3	Calculate total hours	Queries	Number

4.3 Semester Catalog

Semester catalog subsystem acts as the interface between the Degree Planner UI layer and Dashboard layer. It will read data, i.e., the starting semester, from the user information, and sends it to the Web application subsystem.

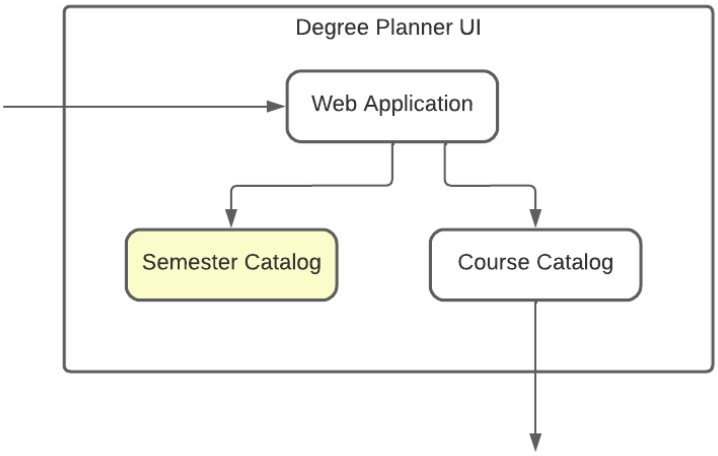


Figure 5: Semester Catalog subsystem

4.3.1 Assumptions

- The user must have Internet connection.
- The subsystem cannot modify data fetched from the Dashboard. It does not provide any functionality to modify data and no data changes should be made in this subsystem.

4.3.2 Responsibilities

This subsystem automatically reads the user's starting semester and returns the list of all valid and available semesters to the Web application subsystem.

4.3.3 SUBSYSTEM INTERFACES

Table 4: Semester Catalog subsystem interfaces

ID	Description	Inputs	Outputs
#1	Fetch data	N/A	List of semesters in the next 10 years from the staring semester

5 Firebase Layer

The Firebase layer consists of three subsystems, Firebase Firestore Database, Firebase Authentication, and Firebase Service.

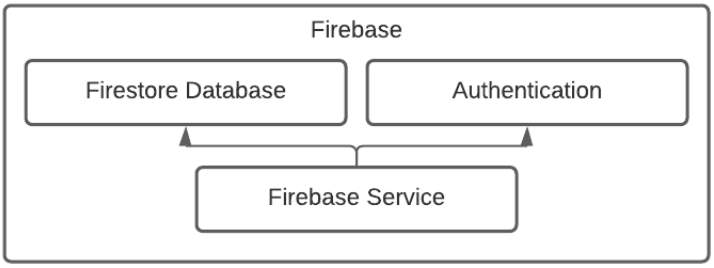


Figure 6: Firebase layer diagram

5.1 Firebase Firestore Database

Firebase Firestore Database is hosted by Google and it holds the data for all users and all of the courses. The data is provided to the website upon request by users.

5.1.1 Assumptions

- Exceptions will be handled by the calling subsystems.
- The user must have Internet connection
- The subsytem should not be used directly by any other subsystem except for Firebase Service

5.1.2 Responsibilities

- Storing the course catalog
- Storing the user's data, including their user ID, major, starting semester, and chosen courses

5.1.3 Subsystem Interfaces

Table 5: Firebase Firestore Database Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Get Courses		Courses
#2	Get Courses Taken		Courses Taken

5.2 Firebase Authentication

Firebase Authentication is hosted by Google and it handles authenticating the users before they can use the website. A user will be authenticated when the correct credentials are provided, or a new account is created.

5.2.1 Assumptions

- Exceptions will be handled by the calling subsystems.
- The user must have Internet connection
- The subsytem should not be used directly by any other subsystem except for Firebase Service

5.2.2 Responsibilities

- Authenticating users so only logged in users can access the degree planner.
- Logging out users
- Changing password

5.2.3 Subsystem Interfaces

Table 6: Firebase Authentication Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Log in	Email Password	User Object
#2	Sign up	Username Email Password	User Object
#3	Logout		Success
#4	Change Password	Email	Email Sent

5.3 Firebase Service

The next subsystem is Firebase Serivice. This subsystem is programmed in the website and controls how the website interacts with the Firebase Database Subsystem and Firebase Authentication.

5.3.1 Assumptions

- Exceptions will be handled, and rethrown is neccessary.
- The user must have Internet connection
- The subsytem should be the only subsystem to access Firebase Firestore Database and Firebase Authentication

5.3.2 Responsibilities

- Control all interactions with Firebase Firestore Database and Firebase Authentication to reduce code duplication
- Authenicating users
- Log out user
- Changing password
- Update user data
- Retrieve all courses

5.3.3 Subsystem Interfaces

Table 7: Firebase Service Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Log in	Email Password	User Object
#2	Sign up	Username Email Password	User Object
#3	Logout		Success
#4	Forgot Password	Email	Email Sent
#5	Get Courses		Courses
#6	Set Major	Major	Success
#7	Set Starting Semester	Starting Semester	Success
#8	Add Chosen Course	Semester Course ID	Success
#9	Remove Chosen Course	Semester Course ID	Success
#10	Get Chosen Courses		Courses

6 Dashboard Layer

This is the dashboard subsystem where it will consist of a link to the Degree Planner, have the ability to view the flowchart of the chosen major, and will be able to sign in and sign out of the user account.

6.1 Homepage/Login

Initially this page will look as the homepage, where the user will have the option to create an account or sign in. Once signed in, the page will turn into the dashboard, then they'll have the option of signing out.

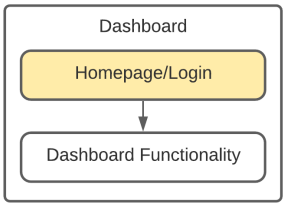


Figure 7: Homepage/Login

6.1.1 Assumptions

An assumption is the user isn't signed in so that means they can view the homepage, and the user is signed in to view the dashboard.

6.1.2 Responsibilities

Initially when the user is not signed-in then there will be only the options of sign-in/create account where this will be shown as a homepage. Only when the user is signed in will the homepage be turned into the dashboard where the user can access other functions of it.

6.1.3 Subsystem Interfaces

Table 8: Homepage/Login Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Create an Account	Fill out given form Set Username and Password	Account is Created
#2	Sign in	Username and Password	User will be Signed In
#3	Sign Out	Click Sign Out button	User will be Signed Out

6.2 Dashboard Functionality

The dashboard will have a link to the Degree Planner functionality, then it will provide an option for the user to view their flowchart for the given major such as Software Engineering, and finally they will be able to sign out of their account.

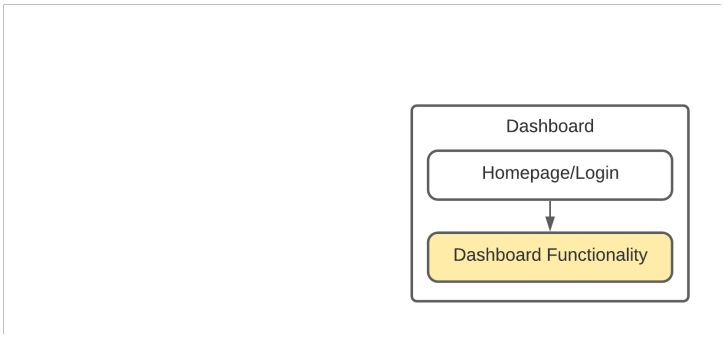


Figure 8: Dashboard Functionality

6.2.1 Assumptions

It is assumed that the user is already signed in so that they can access these features.

6.2.2 Responsibilities

This will provide a link to access the degree planner GUI, then a option to view the flowchart of the chosen major, and also have the option of signing out.

6.2.3 Subsystem Interfaces

Table 9: Dashboard Functionality Subsystem interfaces

ID	Description	Inputs	Outputs
#1	Degree Planner Link	Click on Link	Goes to Degree Planner
#2	View Flowchart	Click on the Button	Flowchart will be available

REFERENCES