

# Internal Design Document

07, February 2020

ASSOCIATED ENGINEERING



**Version:**

0.1

**Team:**

AE3: Team Turtle

**Team Members:**

Bob Ghosh  
Siddhartha Gupta  
Eddie Huang  
Jungwook Jang  
Shuaiqi Zhang  
Wenhong Zhang

**Documented By:**

Bob Ghosh  
Siddhartha Gupta  
Eddie Huang  
Jungwook Jang  
Shuaiqi Zhang  
Wenhong Zhang

**Approved By:**

Siddhartha Gupta  
Eddie Huang  
Jungwook Jang  
Shuaiqi Zhang  
Wenhong Zhang

## Changelog:

02-07-2020, v0.1

Documenting the first implementation of the Design Document.

## Table of Contents

1. Introduction .....	4
1.1 Purpose .....	4
1.2 Scope .....	4
2. Design Consideration .....	4
2.1 Assumptions.....	4
2.2 Constraints .....	4
2.3 System Availability .....	4
2.4 Design Methodology .....	5
3. Project Development Environment .....	5
3.1 Programming Environment.....	5
3.2 Production and Test Environments.....	5
4. Software Architecture. ....	7
5. Data Design.....	8
5.1 Data Dictionary .....	8
5.2 ER Diagram.....	9
6. Software User Interface Design .....	9
6.1 User Interface Description .....	9
6.2 UI Diagram .....	9
7. API Design.....	14
8. Algorithm and Data Structure.....	14
9. Analysis of Trade-Offs and Risks of the Project .....	15
9.1 Back-End Development .....	15
9.2 Testing .....	15
9.3 Front-End Development .....	15

## 1. Introduction

The end-goal of the project is to have a secure web-based resource management system for all employers and clients of the Associated Engineering firm. The system should be able to manage resources to the authenticated users, update user profile or existing projects, forecast based on current resource utilization and be accessed only by the authorized individuals.

### 1.1 Purpose

The purpose of this document is to showcase a detailed description of the designs of the CPSC319 AE's Resource Management System. Essentially, this document is intended for our own group to use as guidelines to implement the project. Moreover, this document will be assessed by the selected representatives to comment and update the current design of the project.

### 1.2 Scope

This document gives a detailed description of the overall design of the project. It describes an environment for programming, production, and testing. Additionally, it presents a detailed description of the software architecture, the structure of database, the user interface design and the API design.

## 2. Design Consideration

### 2.1 Assumptions

The user of the AE resource management system is registered in the Azure Active Directory system. Moreover, the user is informed of the principal operations of a computer and general web-based applications.

### 2.2 Constraints

The system is only accessible by a registered user or an admin. The system should be runnable and optimized for IE 11 and Google Chrome.

### 2.3 System Availability

Technically, the project is built to work on all operating systems as it is web-based. The application is always available through any PCs, that is connected to the organization's internal network. However, if the server hosting the data shuts down unexpectedly, it might

not update changes on the system.

## 2.4 Design Methodology

The system will be open to further development and modification in the future. The adopted design pattern is combination of MVC (Model-View-Control) and Façade (Wrapped) Design Pattern. Therefore, the system has several controllers, however these controllers are wrapped by the main Facade UI. Lastly, all components in the system are built with abstraction.

## 3. Project Development Environment

### 3.1 Programming Environment

The list may be updated as the project proceeds.

Application	Version	Usage
Visual studio code	Ver. 1.42	IDE
GitLab	Latest	For cloning repo, version control
JavaScript	Ver. 9	Programming language for the front-end development
C#	Ver. 8.0	Programming language for the back-end development
.NET Core 3.0	Ver 3.0	.NET Framework
SQL server Express	14.0.1000.169	For database
Node.js/NPM	Latest LTS version	For client
Slack	Latest Slack application	For communication between the team
Lucid Chart	Latest draw.io version	For diagram (UML)
C# Unit Testing Frameworks	Latest version	For unit tests
Windows 10	Ver. 1909	For system tests and development
macOS	Ver. 15.3 (Catalina)	For development

### 3.2 Production and Test Environments

Details about Testing will be elaborated in the upcoming Test Plan Document. Following is a brief description of the tests we are going to apply in this project:

#### 1. Unit Test

Unit test will be performed on major functions on each class to ensure all behaviours of classes are expected and work properly. Modularity is key here.

## 2. UI Test

UI testing will be done manually to ensure all operations run as expected.

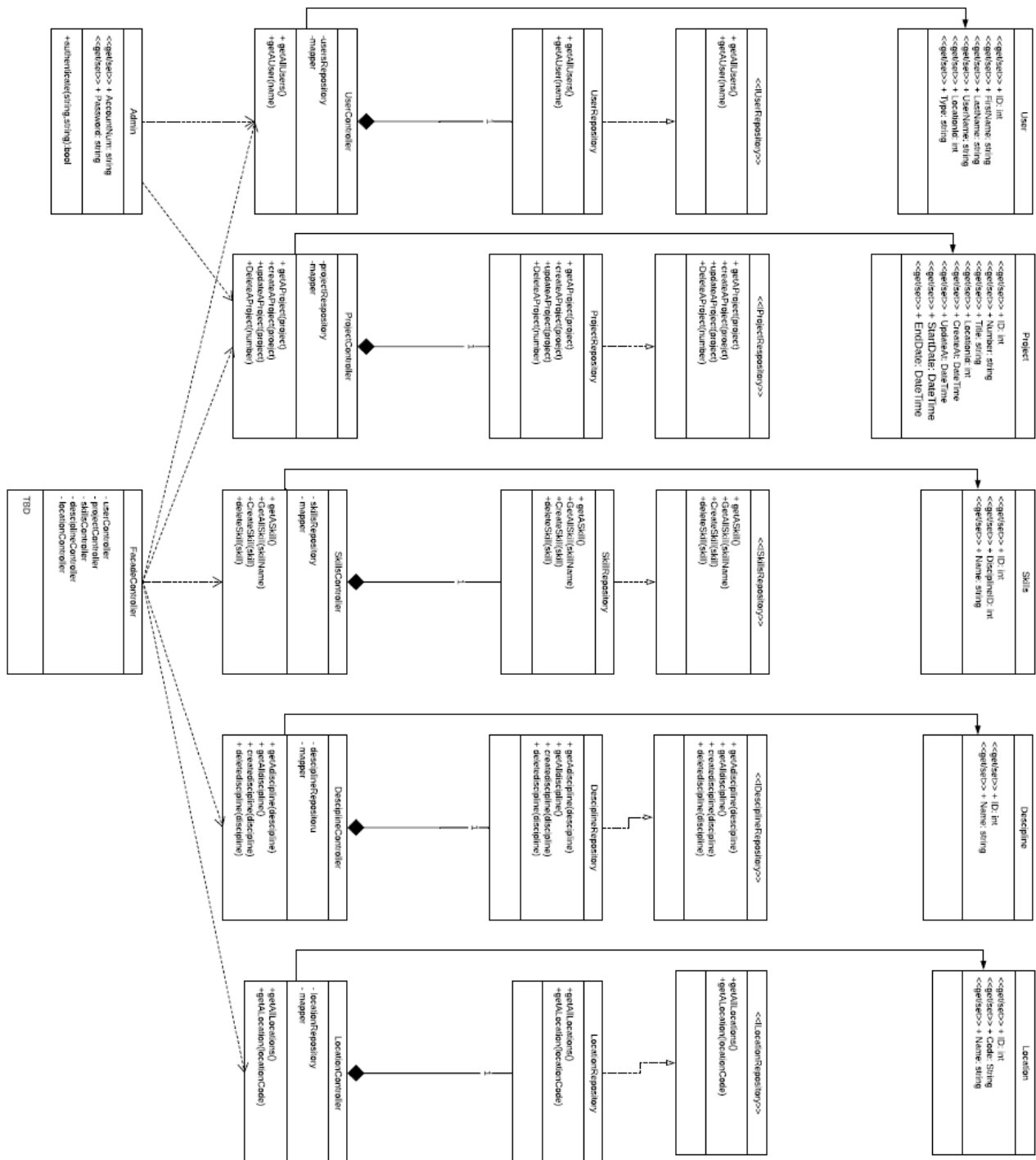
Moreover, whether there is any unexpected behaviour of user interface on various platforms will also be tested.

## 3. End to End Test

End to End testing is necessary to ensure the end product meets the criteria of the required MVP (Minimum Viable Product) as we set up from the early phase of the project.

## 4. Software Architecture.

This UML diagram shows overall class relationship and their functionalities:



## 5. Data Design

### 5.1 Data Dictionary

```
CREATE TABLE [dbo].[Users](
    [Id] [int] NOT NULL IDENTITY(1,1),
    [FirstName] [nvarchar](50) NOT NULL,
    [LastName] [nvarchar](50) NOT NULL,
    [Username] [nvarchar](50) NOT NULL,
    [LocationId] [int] NOT NULL,
    CONSTRAINT [PK_Users] PRIMARY KEY CLUSTERED ([Id]),
    CONSTRAINT [UK_Users_Username] UNIQUE ([Username]),
    CONSTRAINT [FK_Users_Locations] FOREIGN KEY ([LocationId]) REFERENCES [Locations]([Id])
)

CREATE TABLE [dbo].[Projects](
    [Id] [int] NOT NULL IDENTITY(1,1),
    [Number] [nvarchar](50) NOT NULL,
    [Title] [nvarchar](255) NOT NULL,
    [LocationId] [int] NOT NULL,
    [CreatedAt] DATETIME2 NOT NULL DEFAULT SYSUTCDATETIME(),
    [UpdatedAt] DATETIME2 NOT NULL DEFAULT SYSUTCDATETIME(),
    [StarDate] DATETIME2 NOT NULL DEFAULT SYSUTCDATETIME(),
    [EndDate] DATETIME2 NOT NULL DEFAULT SYSUTCDATETIME(),
    CONSTRAINT [PK_Projects] PRIMARY KEY CLUSTERED ([Id]),
    CONSTRAINT [FK_Projects_Locations] FOREIGN KEY ([LocationId]) REFERENCES [Locations]([Id]),
    CONSTRAINT [UK_Projects_Number] UNIQUE ([Number])
)

CREATE TABLE [dbo].[UserInProjects](
    [UserId] INT NOT NULL,
    [ProjectId] INT NOT NULL,
    CONSTRAINT [PK_UserInProjects] PRIMARY KEY ([UserId], [ProjectId]),
    CONSTRAINT [FK_UserInProjects_Users] FOREIGN KEY ([UserId]) REFERENCES [Users]([Id]),
    CONSTRAINT [FK_UserInProjects_Projects] FOREIGN KEY ([ProjectId]) REFERENCES [Projects]([Id])
)

CREATE TABLE [dbo].[Locations](
    [Id] [int] NOT NULL IDENTITY(1,1),
    [Code] [nvarchar](10) NOT NULL,
    [Name] [nvarchar](100) NOT NULL
    CONSTRAINT [PK_Locations] PRIMARY KEY CLUSTERED ([Id]),
    CONSTRAINT [UK_Locations_Code] UNIQUE ([Code]),
    CONSTRAINT [UK_Locations_Name] UNIQUE ([Name])
)

CREATE TABLE [dbo].[Disciplines]
(
    [Id] INT NOT NULL PRIMARY KEY,
    [Name] NVARCHAR(100) NOT NULL
)

CREATE TABLE [dbo].[UserWorksDiscipline](
    [UserId] INT NOT NULL,
    [DisciplineId] INT NOT NULL,
    CONSTRAINT [PK_UserWorksDiscipline] PRIMARY KEY ([UserId], [DisciplineId]),
    CONSTRAINT [FK_UserWorksDiscipline_Users] FOREIGN KEY ([UserId]) REFERENCES [Users]([Id]) ON DELETE CASCADE,
    CONSTRAINT [FK_UserWorksDiscipline_Discipline] FOREIGN KEY ([DisciplineId]) REFERENCES [Disciplines]([Id]) ON DELETE CASCADE
)

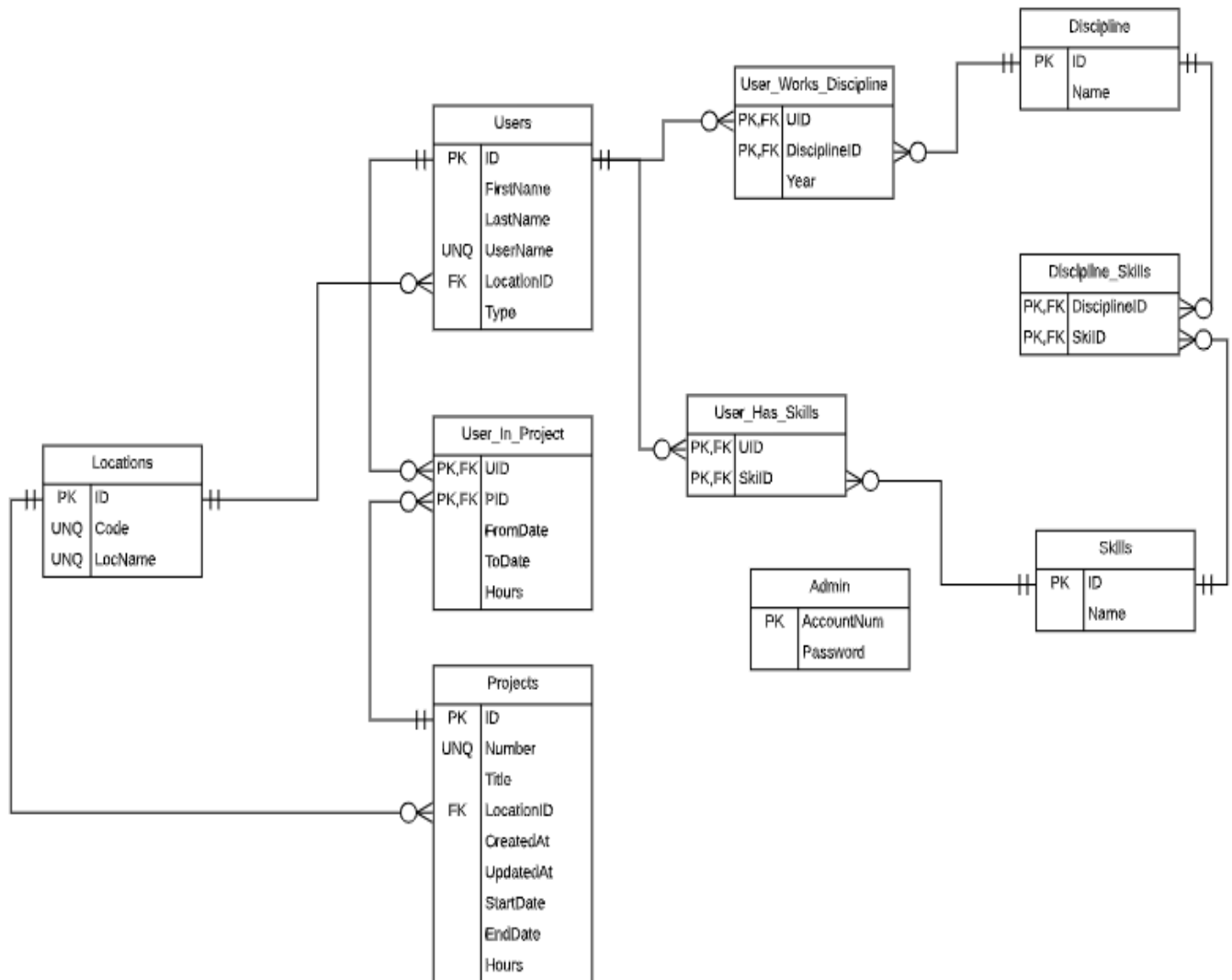
CREATE TABLE [dbo].[Skills](
    [Id] INT NOT NULL,
    [DisciplineId] INT NOT NULL,
    [Name] NVARCHAR(100) NOT NULL,
    CONSTRAINT [FK_Skills_Disciplines] FOREIGN KEY ([DisciplineId]) REFERENCES [Disciplines]([Id]) ON DELETE CASCADE,
    CONSTRAINT [PK_Skills] PRIMARY KEY ([DisciplineId], [Id])
)

CREATE TABLE [dbo].[UserHasSkills](
    [UserId] INT NOT NULL,
    [SkillId] INT NOT NULL,
    CONSTRAINT [PK_UserHasSkills] PRIMARY KEY ([UserId], [SkillId]),
    CONSTRAINT [FK_UserHasSkills_Users] FOREIGN KEY ([UserId]) REFERENCES [Users]([Id]) ON DELETE CASCADE,
    CONSTRAINT [FK_UserHasSkills_Skills] FOREIGN KEY ([SkillId]) REFERENCES [Skills]([Id]) ON DELETE CASCADE
)
```



## 5.2 ER Diagram

This diagram shows the overall database and relationship between entities.



## 6. Software User Interface Design

### 6.1 User Interface Description

The system user interface is designed according to several principles.

1. **Simplicity**: UI is designed to be simple and easy to use.
2. **Open to Modifications**: UI can be further upgraded and/or modified.
3. **Consistency**: each main view of the UI is organized in such a way that is related tasks.

### 6.2 UI Diagram

Planned view of the main User Interface:

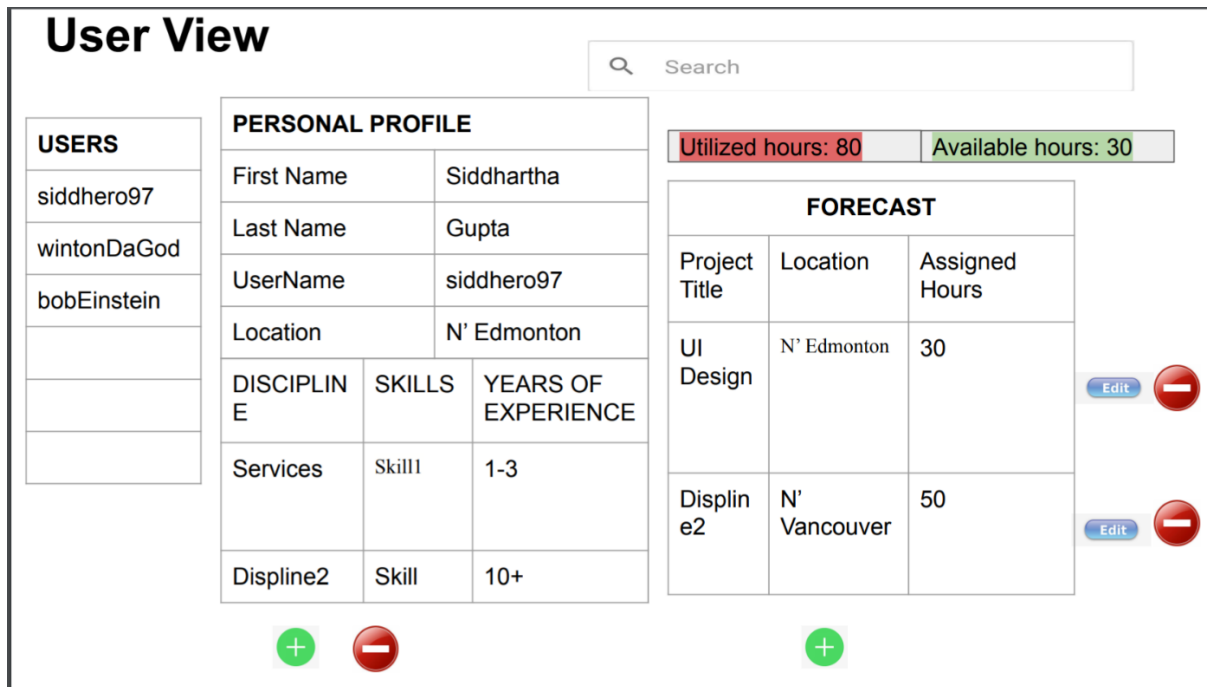


Figure 6.1(user view)

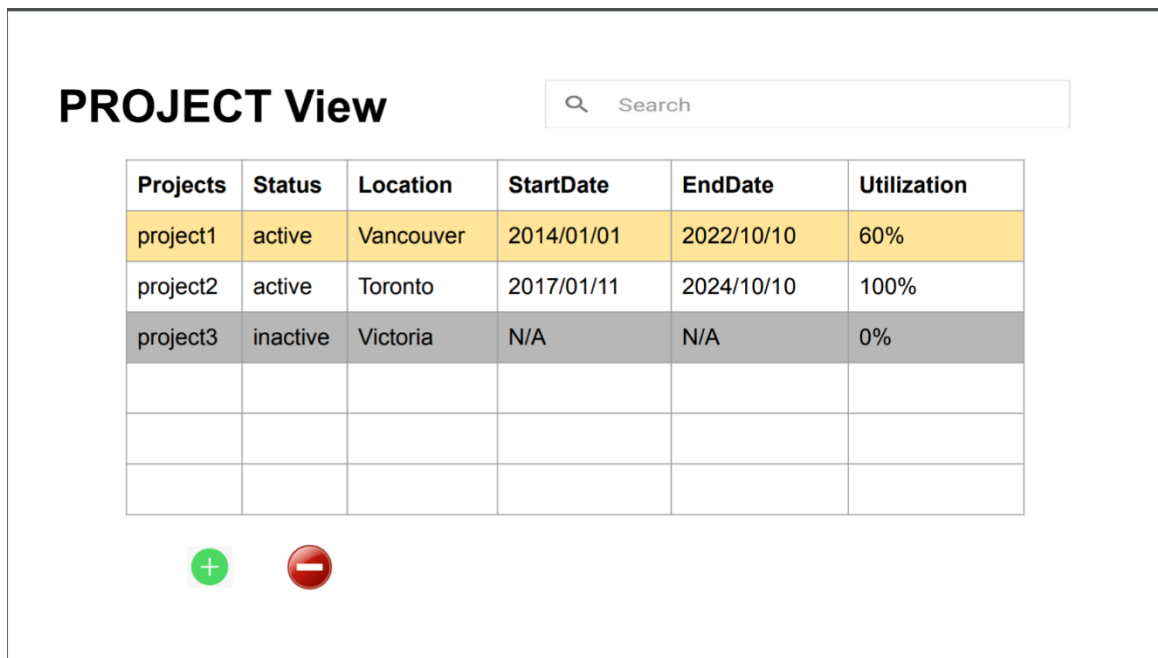


Figure 6.2 Project view. Inactive project will be highlighted in grey, and active project with least utilization will be place on top of the chart and will be highlighted in yellow.

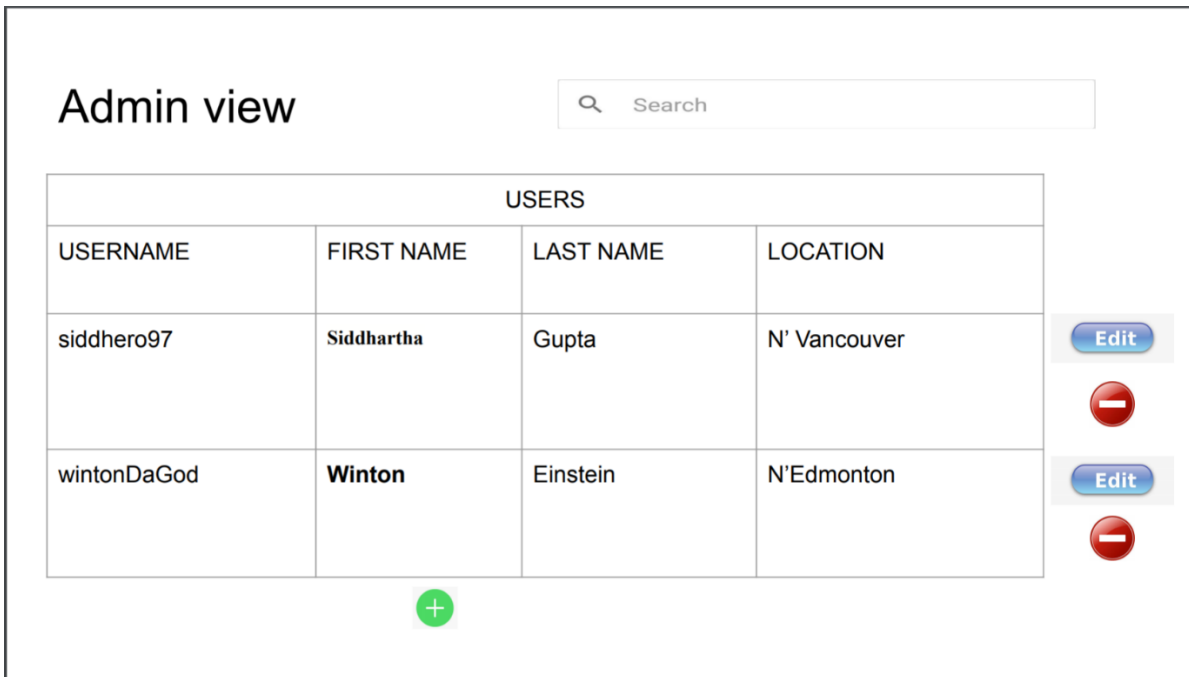


Figure 6.3 Admin View.

The following diagram shows that how to remove/add entities on User/Project

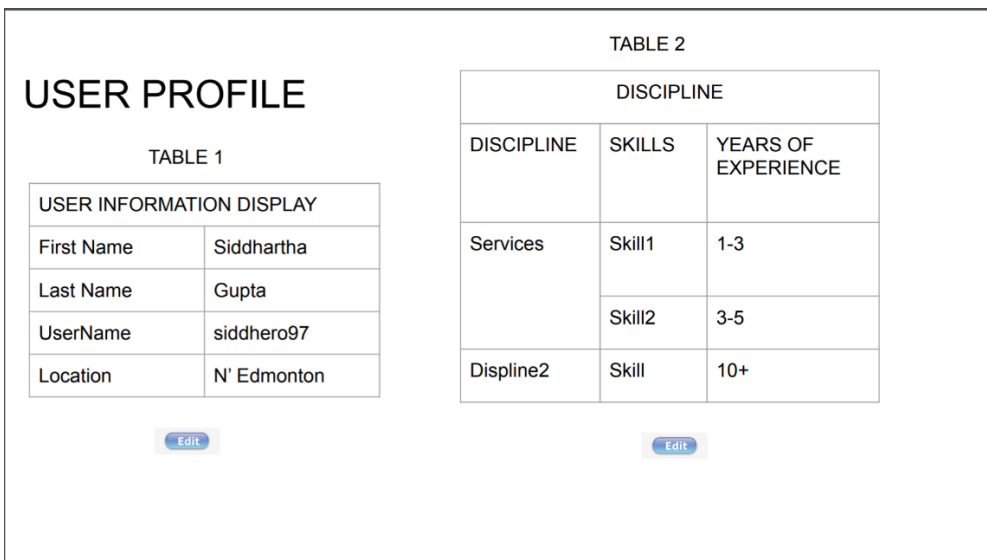


Figure 6.4 User profile view.

## CHANGES TO TABLE WHEN IS PRESSED

DISCIPLINE				
DISCIPLINE	SKILLS	YEARS OF EXPERIENCE		
Services	Skill1	1-3		
Displine2	Skill	10+		
				

Figure 6.5 This is detailed view of removing entities on discipline table.

## CHANGES TO TABLE WHEN IS PRESSED











DISCIPLINE				
DISCIPLINE	SKILLS	YEARS OF EXPERIENCE		
Services	Skill1	1-3		
	Skill2	3-5		
Displine2	Skill	10+		
Drop Down Showing All Disciplines	Drop Down Showing Skills Associated With Experience	Drop Down Showing All Possible Years Of Experience		
				

Figure 6.6 This is detailed view of adding entities of discipline table.

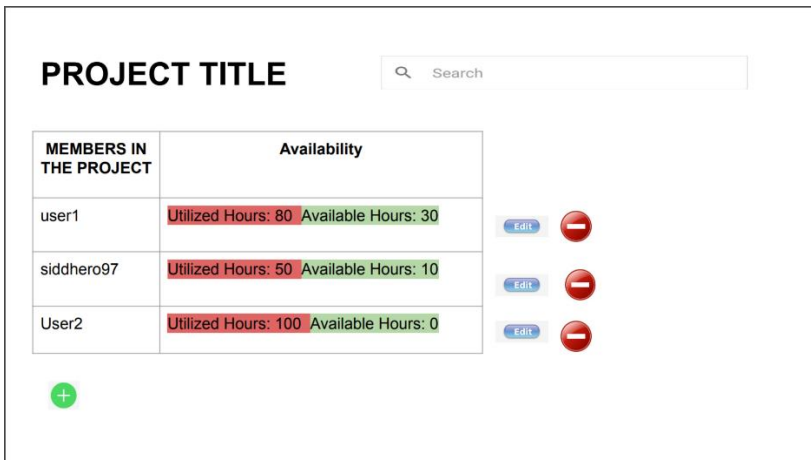
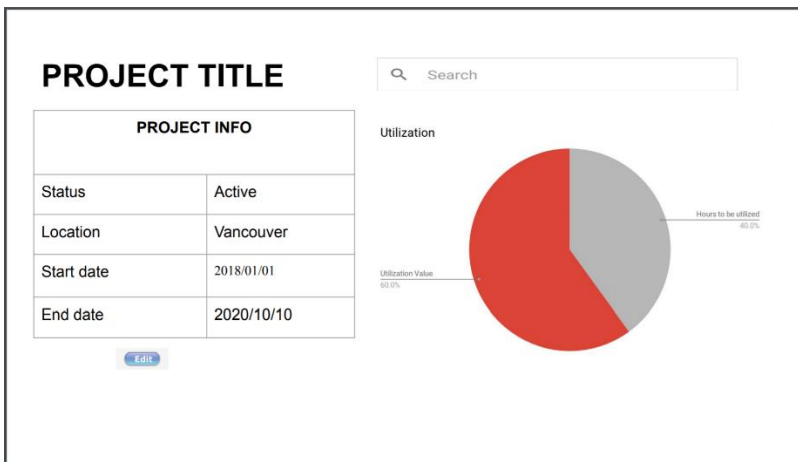


Figure 6.7 This is view when user click a project from project view.

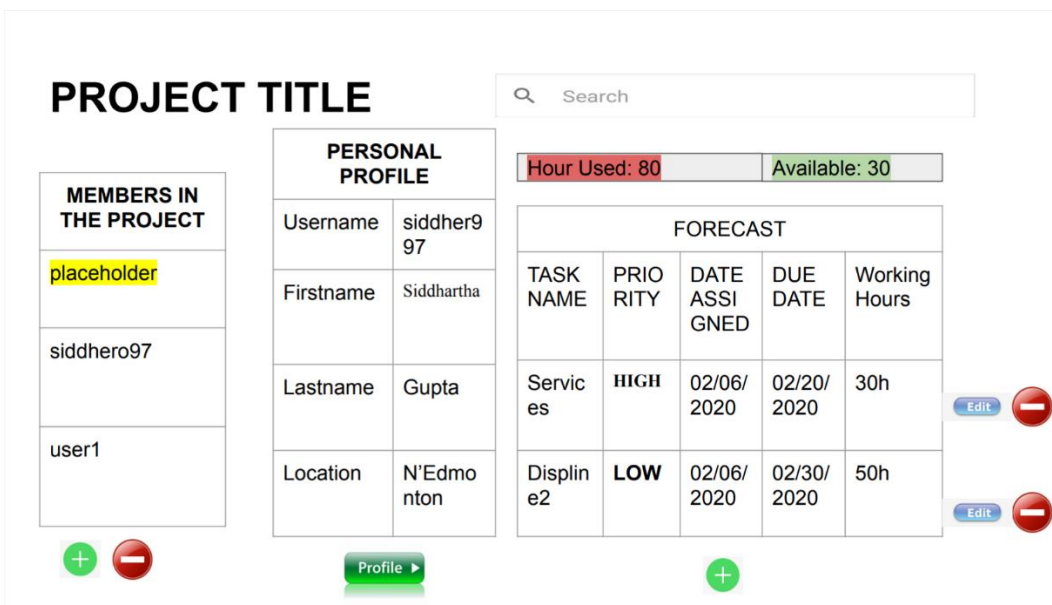


Figure 6.8 This is pop up view from project title when user click members in the project.

## When we choose placeholder

MEMBERS IN THE PROJECT
user1
siddhero97
placeholder

PLACEHOLDER INFO	
Disciple	Software Engineering
Skill	-Javascript -Java -C++
Year of experience	5+
Location	N'Edmonton
Availability	30h

Result	
Username	Availability
wintonTheGod	30h
bobEinstein	40h
rickBeiber	50h

Edit

Search

Figure 6.9 This is pop up view from project title when user click place holder icon.

## 7. API Design

Any Open Source API maybe added to the list if it is deemed necessary in the future.

API	Version	Usage
AzureAD	Latest version	For authentication
React	V16.12.0	For the front-end development
Bootstrap	V4.4.1	For the front-end development
Google developer	Latest version	For the front-end testing
TBD		

## 8. Algorithm and Data Structure

Since this system does not require complicated operation, complex algorithms and/or data structure would not be necessary. However, we may use one in the future if required.

## 9. Analysis of Trade-Offs and Risks of the Project

### 9.1 Back-End Development

Since we have decided using all proposed technology from AE, there are not many notable trade-offs pertaining to the back-end development. However, for data structure to represent the data, we decided to use a very simple architecture. We may improve upon the data structure in this project, however, as the system does not require knotty operations, using efficient data structure may not improve efficiency significantly.

### 9.2 Testing

For full system tests, we would need to use a setup which runs IE 11 natively. That is to say we would need to run Windows 8 or newer in our development systems. For this, we could run Windows on an Open-Source or free VM (for example, Oracle's VirtualBox), run natively on a PC-system, or we can Dockerize the application. Dockerization is our stretch goal and would require extra time and resources, which we might not be able to achieve. Most of the development would take place in a macOS environment, as majority of our team members have Mac as their development environment: this could result in some unexpected behaviours which we could have neglected or not have encountered during our rigorous testing.

### 9.3 Front-End Development

As said above, most of our team members would be using macOS as their development environment, so this could result in some unexpected behaviours which we could have neglected or not have encountered during testing or development.