UB Course Registration System Project

Group: Group 11

Team Member: Shiva Shankar Ganesan (1056083)

Team Member: Nitish Subhash Soman (1064382)

TABLE OF CONTENTS

PHASE 1:

ABSTRACT:

In this project, we are designing a web application for University of Bridgeport Course Registration System. This gives the complete flexibility to the three types of users like students, adviser and admin to manage courses.

We are using ASP.Net C# programming language to implement this project using Microsoft visual studio 2017 IDE. To implement RDBMS, we are using SQL Server that can run on Google Cloud Platform (GCP) cloud engine.

PROBLEM STATEMENT:

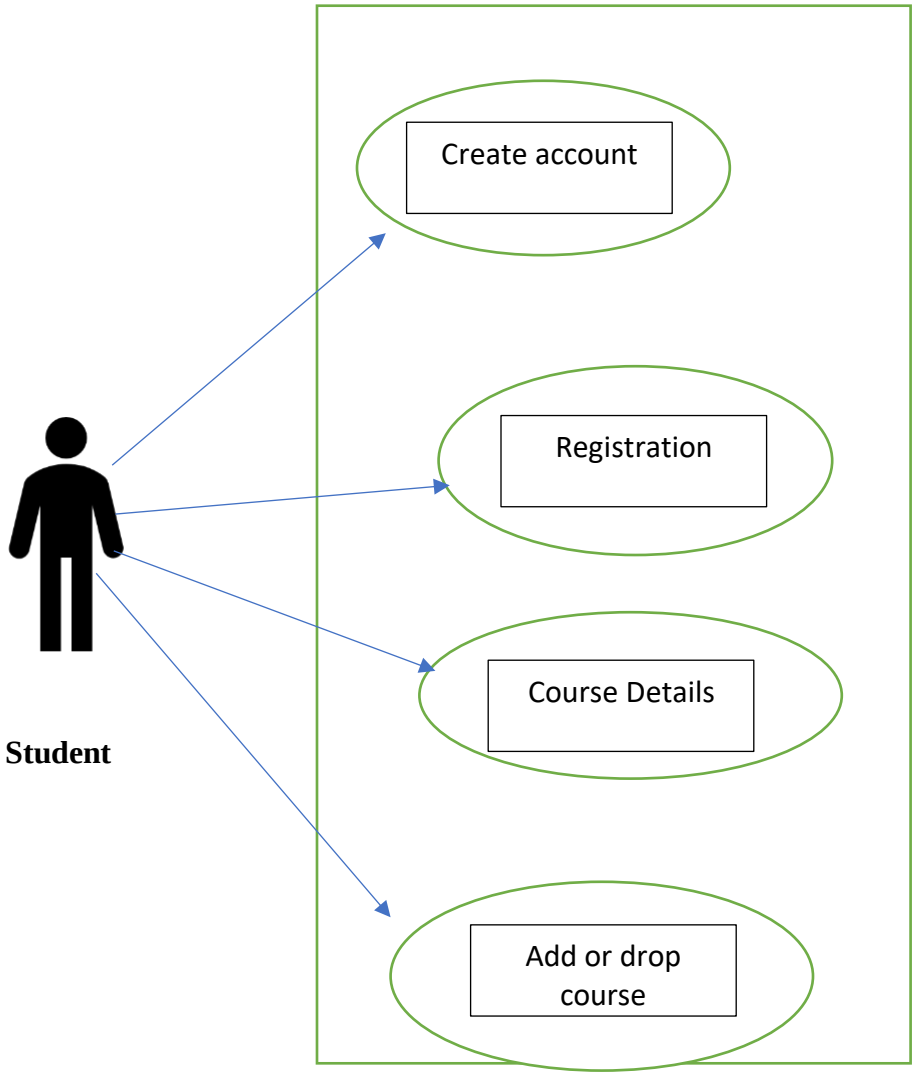
UB web-based course registration for students who can sign-up and sign-in to the portal to register & manage courses. Also, student can update their profile data and add or drop their courses. Along with registration department can sign-in to the portal to manage, modify courses and manage room assignment. Registration department act as admin who have all rights like add, delete, update courses. Moreover, Adviser can have access to approve or decline student's course registration. Adviser also has access to view course registrations for all students and they can send message to students regarding any issue.

QUERIES:

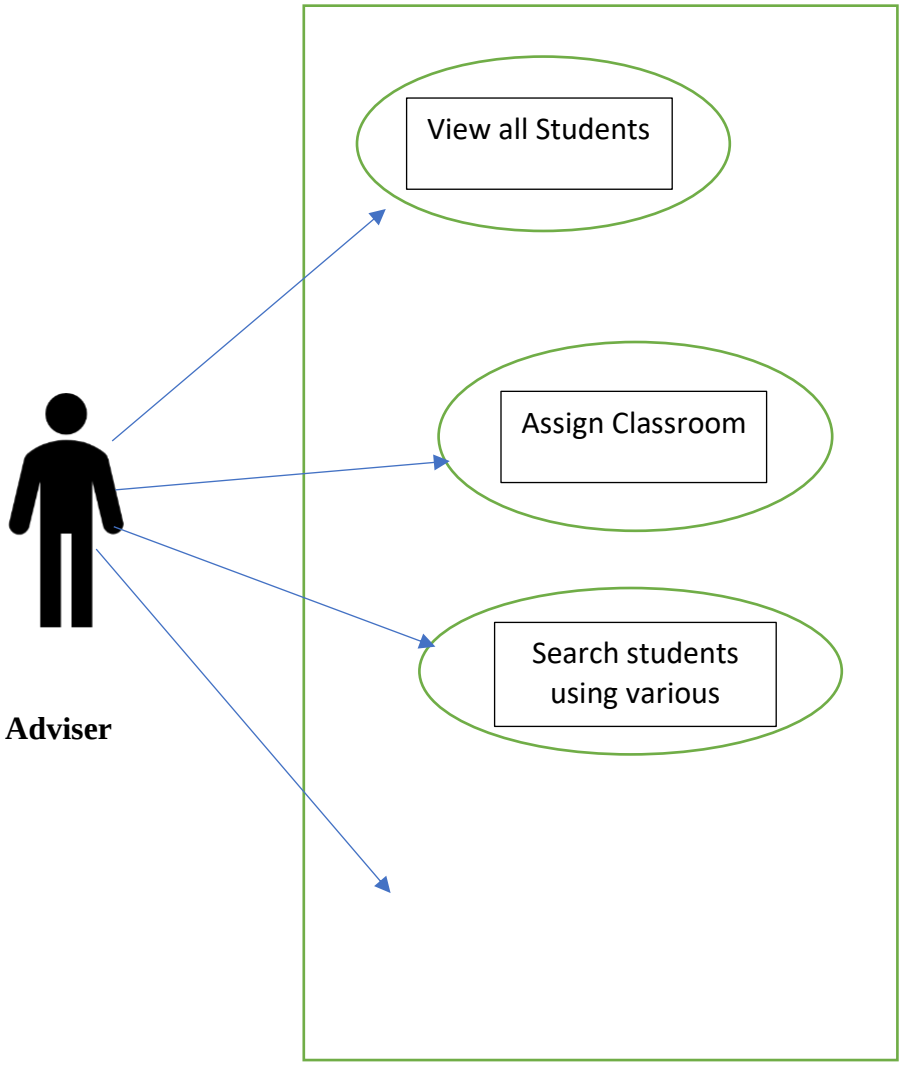
- Query to display all the professor's name who teaches "Database" course:
- Query to display all the rooms with capacity no more than 30 students
- Query to fetch student ID , student name , section number of course python programming:
- Query to fetch List of userdId, students 'First Name from computer science department
- Query to fetch course names which has more than 1 sections
- Query to list messages received by student name "Shiva"
- Query to list of courses in which Shiva Ganesan is enrolled
- Query to list of courses conducted in class room number 203 of Carlson hall
- Query to fetch courses which are taught as Distance Learning:
- Query to display number of courses under each department
- Query to fetched Student ID and Name enrolled in "Distance Learning"
- Query to fetch Department names whose fess is more than 800
- Query to list of course, department who credit is 1
- Query to fetch courses of computer science department which is conducted on Mondays
- Query to fetch undergrad courses names of Business department

UML DIAGRAM:

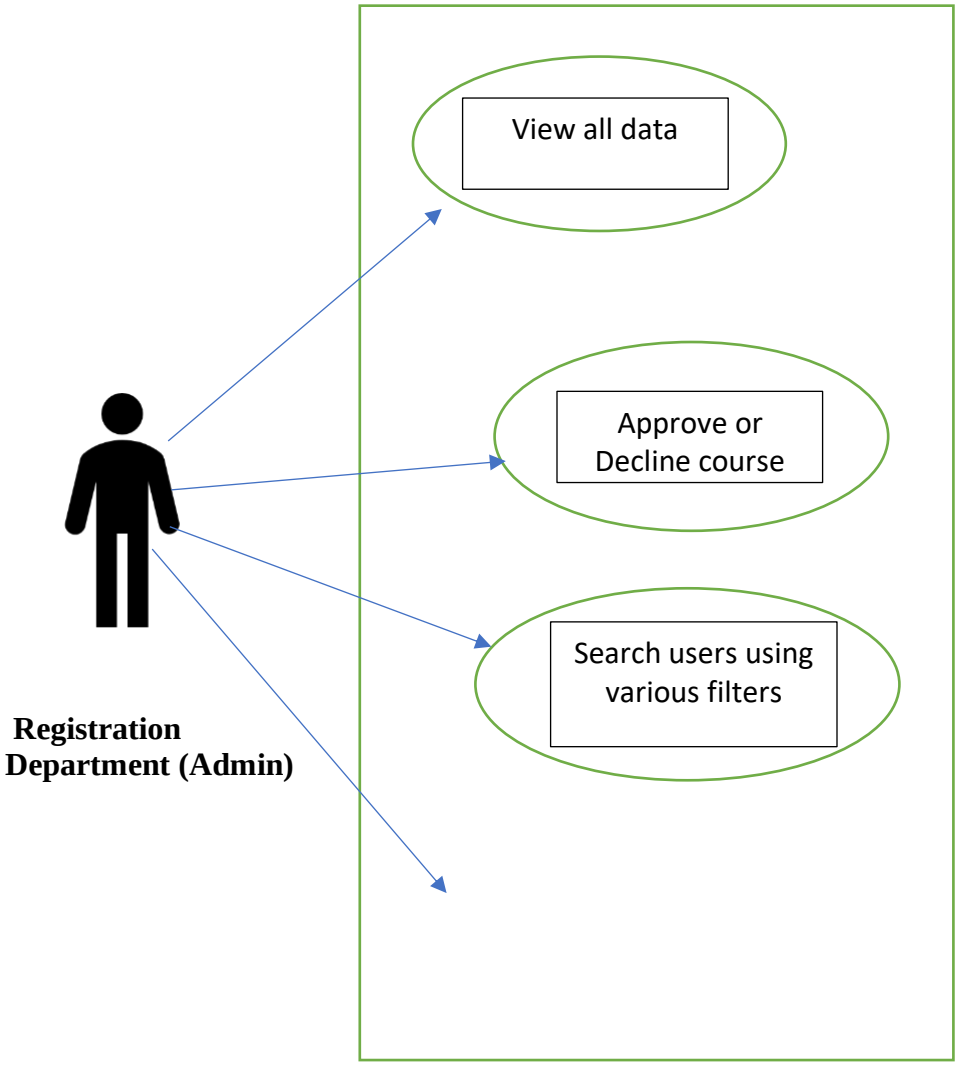
Student:



Adviser:


View all Students
Assign Classroom
Search students using various
Adviser

Registration Department:



ASSUMPTIONS:

We will try to do all the stated above tasks and even add some more features in future. But mainly we focus to fulfill all the stated functionalities. We will try to create database in such way that it fulfills relationship between all the entities. Major assumptions is student can register to max 4 courses per semester.

PHASE 2:

ENTITIES AND ATTRIBUTES:

1. USER is an entity with attributes- UserID, UserType, EmailId, Password, Gender, Name which has First Name, Last Name, DOB, ContactNo, Address which has StreetNo, City, State, ZIPCode.
2. STUDENT entity with AcademicLevel attribute.
3. PROFESSOR entity with attributes ProfessotType and IsAdvisor.
4. ADMIN entity with attributes AdminRole.
5. COURSES entity with attributes CourseId, CourseName, CourseNumber, ContactNo, Term.
6. SECTION entity with attributes SectionId, SectionNumber.
7. DEPARTMENT entity with attributes DeptId, DeptName.
8. REGISTRATION entity with attributes RegistrationId, RegistrationType.
9. CLASSROOM entity with attributes ClassroomId, Building, ClassNumber, Location, StudentCapacity.
10. NOTIFICATION entity with attribute Message.
11. FEES entity with attribute Amount.

RELATIONSHIPS:

1. STUDENT can VIEW many NOTIFICATION (1:M).
2. PROFESSOR can Notify many NOTIFICATION (1:M).
3. STUDENT is BELOGTO DEPARTMENT (M:1).
4. STUDENT WILLREGISTER for course REGISTRATION (1:M).
5. PROFESSOR can WORKUNDER many DEPARTMENT (M:N).
6. ADMIN can POST many COURSES (1:M).
7. COURSE is UNDER DEPARTMENT (M:1).
8. COURSE is UNDER many SECTIONS (M:N).
9. SECTION is HELD at CLASSROOM (1:1).
10. SECTION is FOR REGISTRATION (1:M).

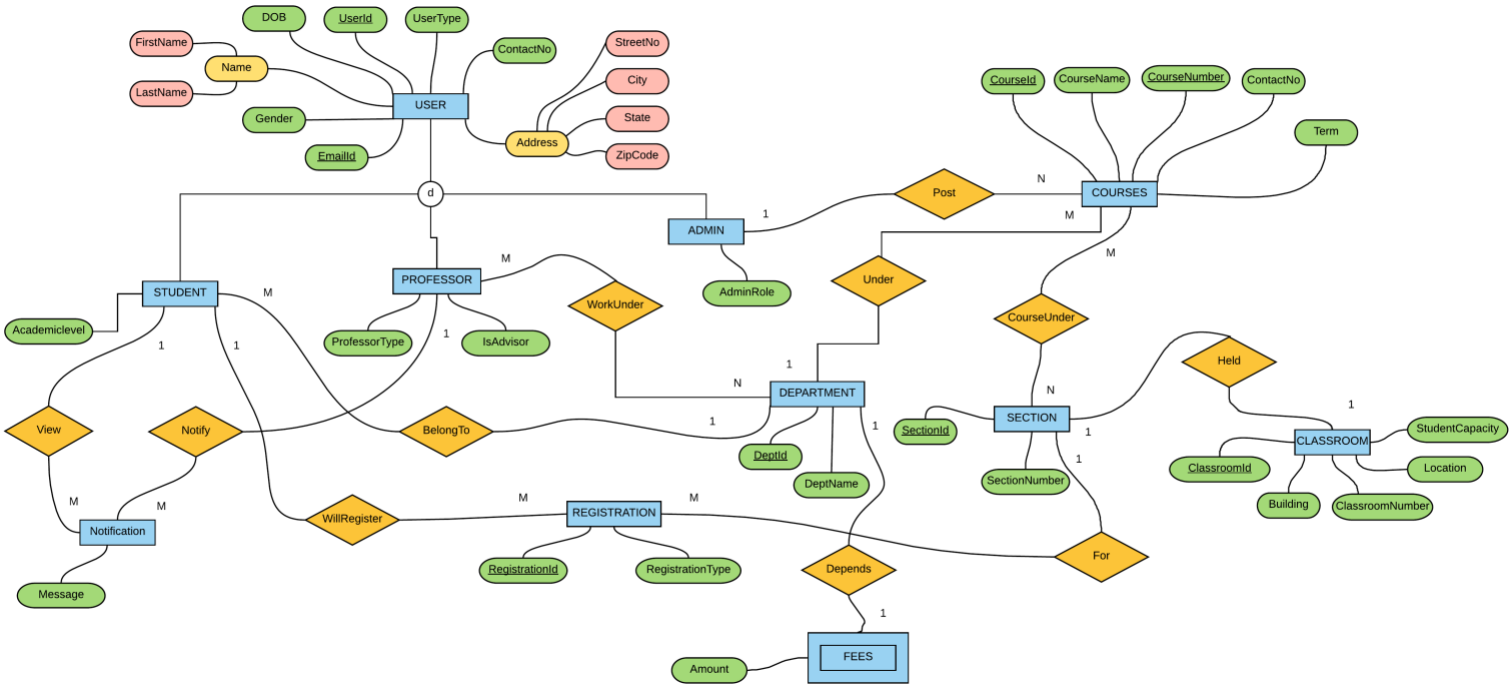
WEAK ENTITY:

In our project we consider FEES as weak entity.

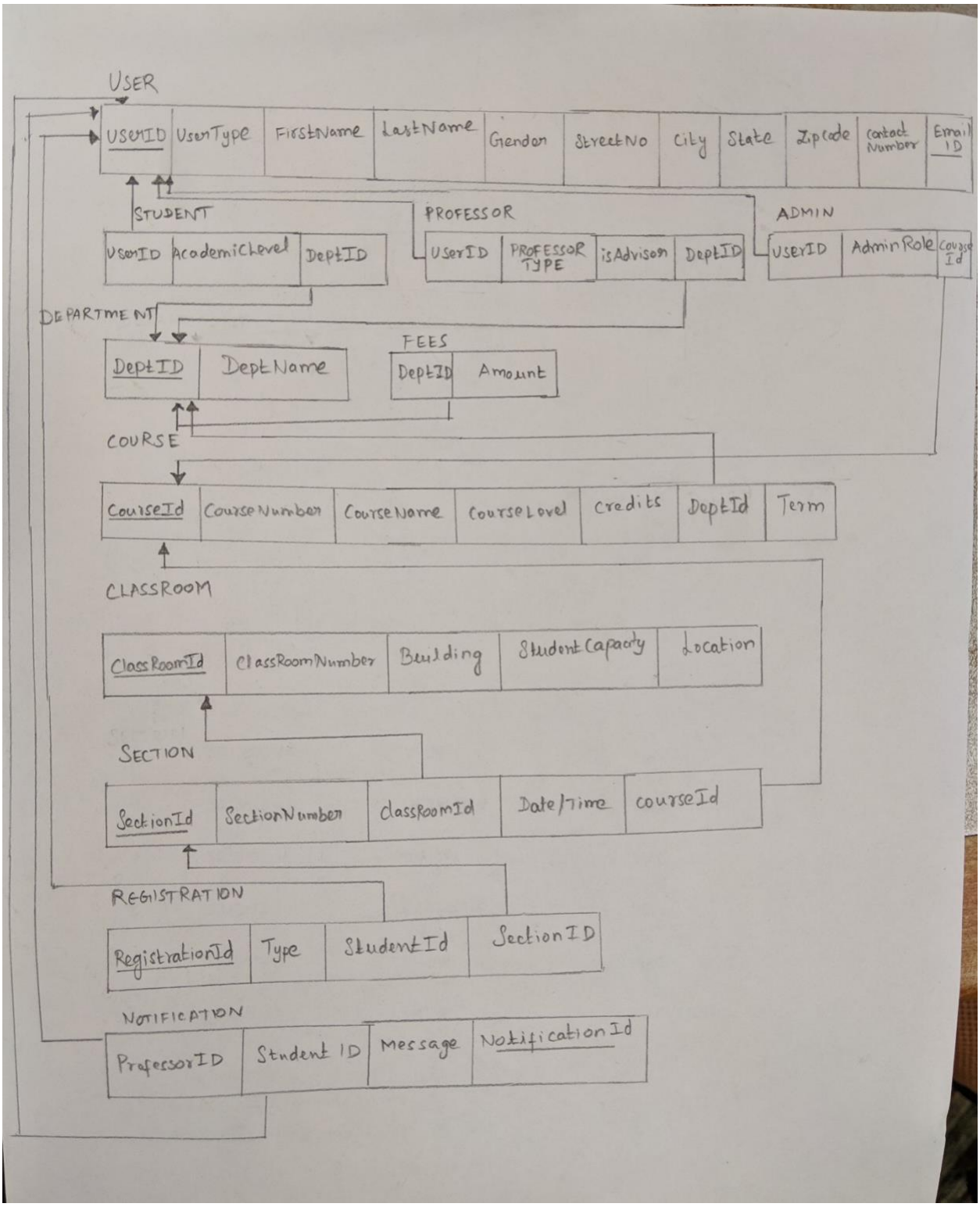
GENERALIZATION/SPECIALIZATION:

The USERS can classified into STUDENT, PROFESSOR and ADMIN which have different roles and permissions.

EER MODELING:



RELATIONAL SCHEMA:



RELATIONAL ALGEBRA:

Query 1: Query to display all the professor's name who teaches "Database" course:

$$\text{course} \leftarrow \Pi_{\text{courseId}}\left(\sigma_{\text{courseName} = \text{Database}}(\text{COURSE})\right)$$

$$\text{section} \leftarrow \Pi_{\text{sectionId}}\left(\text{SECTION} \bowtie_{\text{courseId} = \text{courseId}} \text{Course}\right)$$

$$\text{uses} \leftarrow \Pi_{\text{userId}}\left(\text{REGISTRATION} \bowtie_{\text{sectionId} = \text{sectionId}} \text{Section}\right)$$

$$\text{result} \leftarrow \Pi_{\text{firstName}, \text{lastName}}\left(\sigma_{\text{userType} = \text{Professor}}\left(\text{USER} \bowtie_{\text{userId} = \text{userId}} \text{Uses}\right)\right)$$

Query 2: Query to display all the rooms with capacity no more than 30 students

$$\Pi_{\text{classNumber}}\left(\sigma_{\text{studentCapacity} <= 30}(\text{CLASSROOM})\right)$$

Query 3: Query to fetch student ID , student name , section number of course python programming:

$$\text{course} \leftarrow \Pi_{\text{courseId}}\left(\sigma_{\text{courseName} = \text{Python Programming}}(\text{COURSE})\right)$$

$$\text{section} \leftarrow \Pi_{\text{sectionId}}\left(\text{SECTION} \bowtie_{\text{courseId} = \text{courseId}} \text{Course}\right)$$

$$\text{uses} \leftarrow \Pi_{\text{userId}, \text{sectionId}}\left(\text{REGISTRATION} \bowtie_{\text{sectionId} = \text{sectionId}} \text{Section}\right)$$

$$\text{temp} \leftarrow \sigma_{\text{userType} = \text{student}}\left(\text{USER} \bowtie_{\text{userId} = \text{userId}} \text{Uses}\right)$$

$$\text{result} \leftarrow \Pi_{\text{studentId}, \text{firstName}, \text{sectionNumber}}$$

Query 4: List of userdId, students 'First Name from computer science department:

$$department \leftarrow \pi_{DeptId}(\sigma_{DeptName = Computer\ Science})$$

$$students \leftarrow \sigma_{userType = student}$$

$$result \leftarrow \pi_{userId,\ firstName}(department \bowtie_{DeptId = DeptId} students)$$

Query 5: Course names which has more than 1 sections:

$$count \leftarrow courseId\ \mathcal{F}_{COUNT\ courseId}(SECTION)$$

$$temp \leftarrow \sigma_{course\_count > 1}(count)$$

$$result \leftarrow \pi_{courseId,\ courseName}(COURSE \bowtie_{courseId = courseId} temp)$$

Query 6: List messages received by student name "Shiva"

$$user \leftarrow \pi_{userId}(\sigma_{FirstName = Shiva}(\sigma_{userType = student}(USER)))$$

$$result \leftarrow \pi_{userId,\ Message}(NOTIFICATION \bowtie_{userId = studentId} User)$$

Query 7: List of courses in which Shiva Ganesan is enrolled

$$User \leftarrow \Pi_{userId}\left(\sigma_{firstName = Shiva,\ lastName = Ganesan}\left(\sigma_{userType = student}(USER)\right)\right)$$

$$Section \leftarrow \Pi_{sectionId}\left(REGISTRATION \bowtie_{userId = userId} User\right)$$

$$Courses \leftarrow \Pi_{courseId}\left(SECTION \bowtie_{courseId = courseId} Section\right)$$

$$result \leftarrow \Pi_{courseId,\ courseName}\left(COURSE \bowtie_{courseId = courseId} Courses\right)$$

Query 8: List of courses conducted in class room number 203 of Carlson hall:

$$classrooms \leftarrow \Pi_{classroomId}\left(\sigma_{classNumber = 203,\ building = carlsonHall}(CLASSROOM)\right)$$

$$sections \leftarrow \Pi_{courseId}\left(SECTION \bowtie_{classroomId = classroomId} classrooms\right)$$

$$result \leftarrow \Pi_{courseId,\ courseName}\left(COURSE \bowtie_{courseId = courseId} sections\right)$$

Query 9: Courses which are taught as Distance Learning:

$$type \leftarrow \Pi_{sectionId}\left(\sigma_{type = DL}(REGISTRATION)\right)$$

$$courses \leftarrow \Pi_{courseId}\left(SECTION \bowtie_{sectionId = sectionId} type\right)$$

$$result \leftarrow \Pi_{courseId,\ courseName}\left(COURSE \bowtie_{courseId = courseId} courses\right)$$

Query 10: Query to display number of courses under each department

$$count \leftarrow {}_{deptId}\mathcal{F}_{COUNT\ deptId}(COURSE)$$

$$result \leftarrow \Pi_{deptName,\ count\_dept}(DEPARTMENT \bowtie_{deptId = deptId} count)$$

Query 11: Student ID and Name enrolled in "Distance Learning":

$$courses\ type \leftarrow \Pi_{userId}(\sigma_{Type = DL}(REGISTRATION))$$

$$result \leftarrow \Pi_{userId,\ FirstName}(\sigma_{usertype = Student}(USER \bowtie_{userId = userId} Courses\ type))$$

Query 12: Department names whose fess is more than 800$:

$$departments \leftarrow \Pi_{DeptId}(\sigma_{Amount > 800}(FEES))$$

$$result \leftarrow \Pi_{deptID,\ DeptName}(DEPARTMENT \bowtie_{deptId = deptId} departments)$$

Query 13: List of course, department who credit is 1:

$$credits \leftarrow \Pi_{courseId,\ courseName,\ deptId}(\sigma_{credits = 1}(COURSE))$$

$$result \leftarrow \Pi_{courseId,\ courseName,\ deptName}(DEPARTMENT \bowtie_{deptId = DeptId} Credits)$$

Query 14: Courses of Computer science department conducted on Mondays:

$$CS \leftarrow \Pi_{deptId}\left(\sigma_{deptName = \text{computer Science}}\right)$$

$$Courses \leftarrow \Pi_{courseId, courseName}\left(COURSE \bowtie_{deptId = deptId} CS\right)$$

$$result \leftarrow \Pi_{courseId, courseName, day}\left(SECTION \bowtie_{courseId = courseId} Courses\right)$$

$$final \leftarrow \sigma_{day = Monday}\left(result\right)$$

Query 15: Undergrad courses names of Business department:

$$Bus \leftarrow \Pi_{deptId}\left(\sigma_{deptName = Business}\right)$$

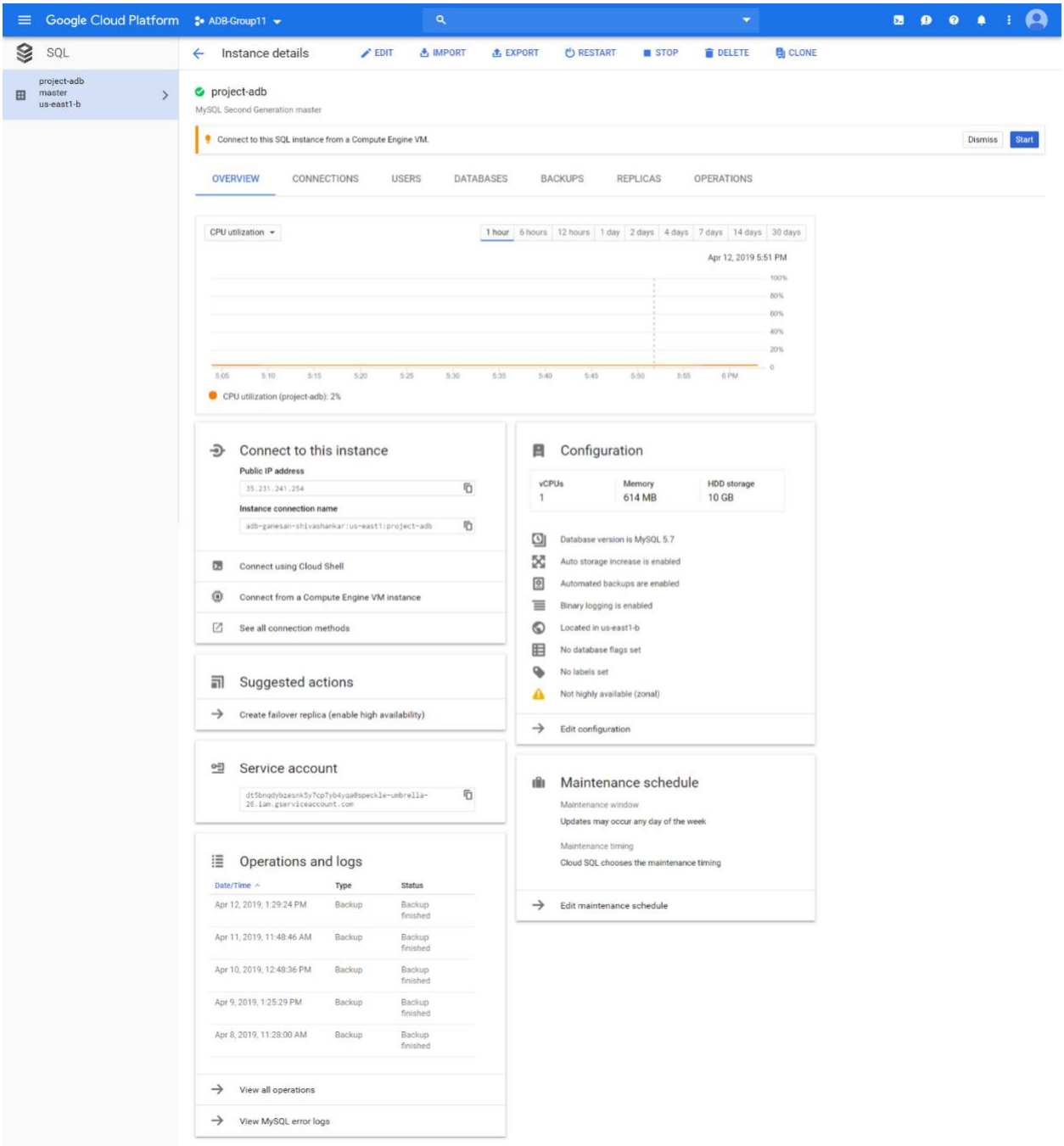
$$temp \leftarrow COURSE \bowtie_{deptId = deptId} Bus$$

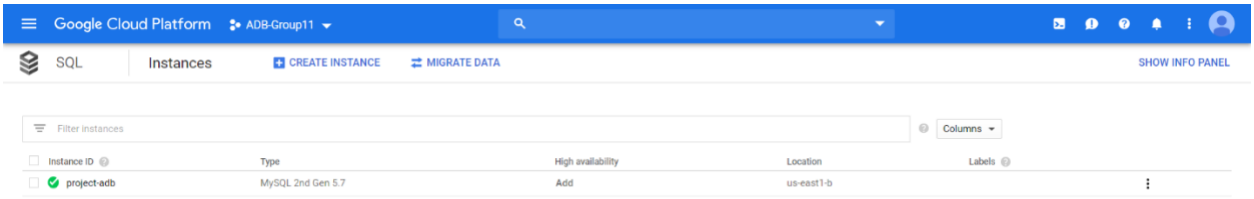
$$result \leftarrow \Pi_{courseId, courseName}\left(\sigma_{courseLevel = UG}\left(temp\right)\right)$$

PHASE 3:

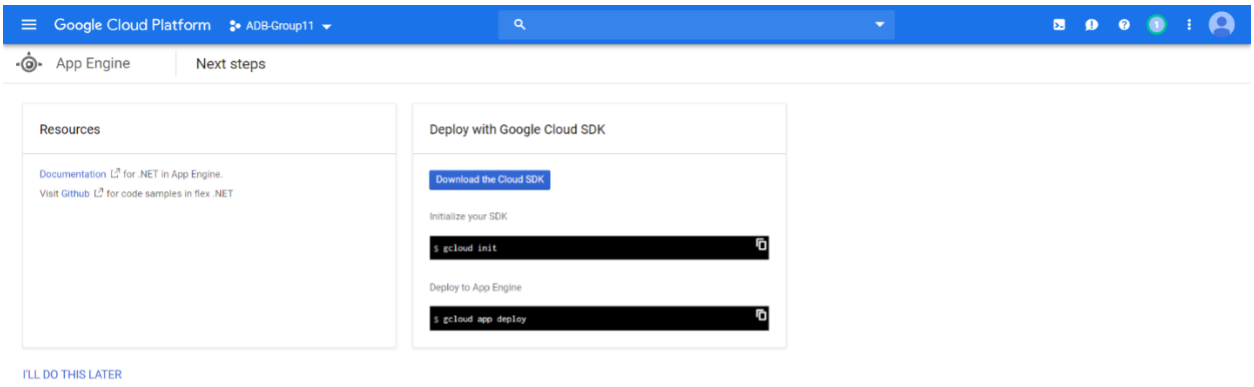
In this phase we have created the database server and web application server in the GCP.

DATABASE SERVER





WEB APPLICATION SERVER:



PHASE 4:

In this phase we have implemented the database tables.

- "dbDDL.sql" contains the script for creating DDL script like creating database schema, views, constraints, tables, triggers, etc.,
- dbDML.sql contains DML script for insert, update statements
- dbDrop.sql contains Drop script for dropping the tables, view etc.,
- dbSQL contains SQL script for creating join queries etc

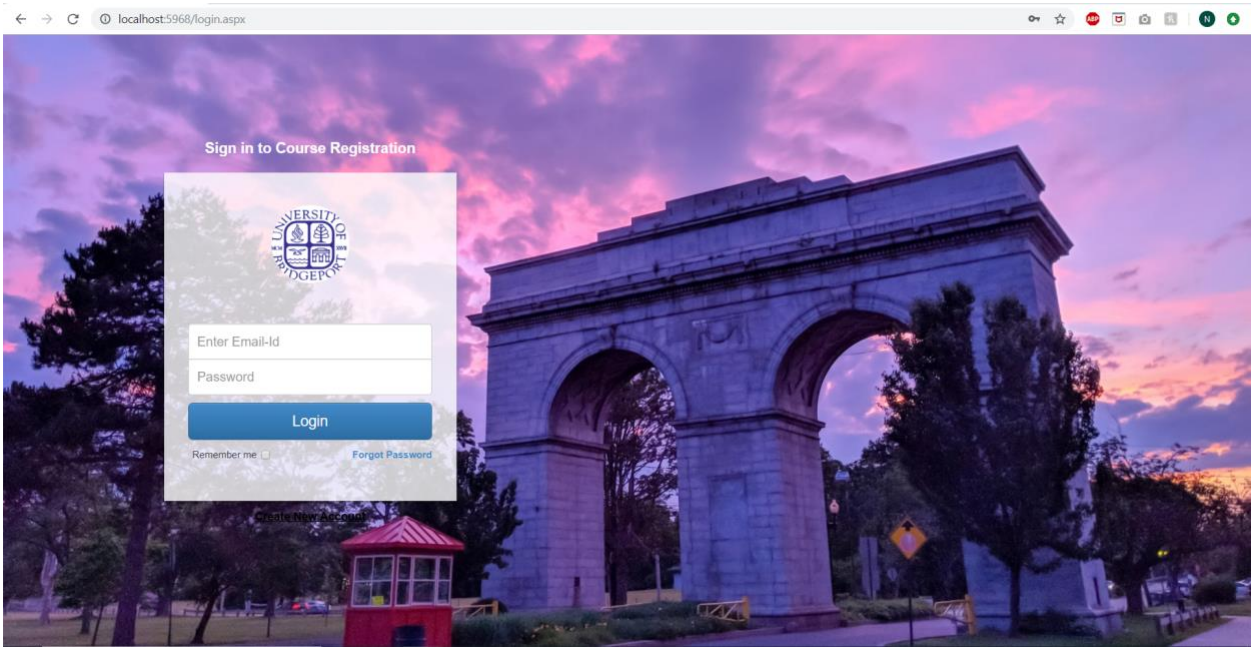
These files are added in the folder

PHASE 5:

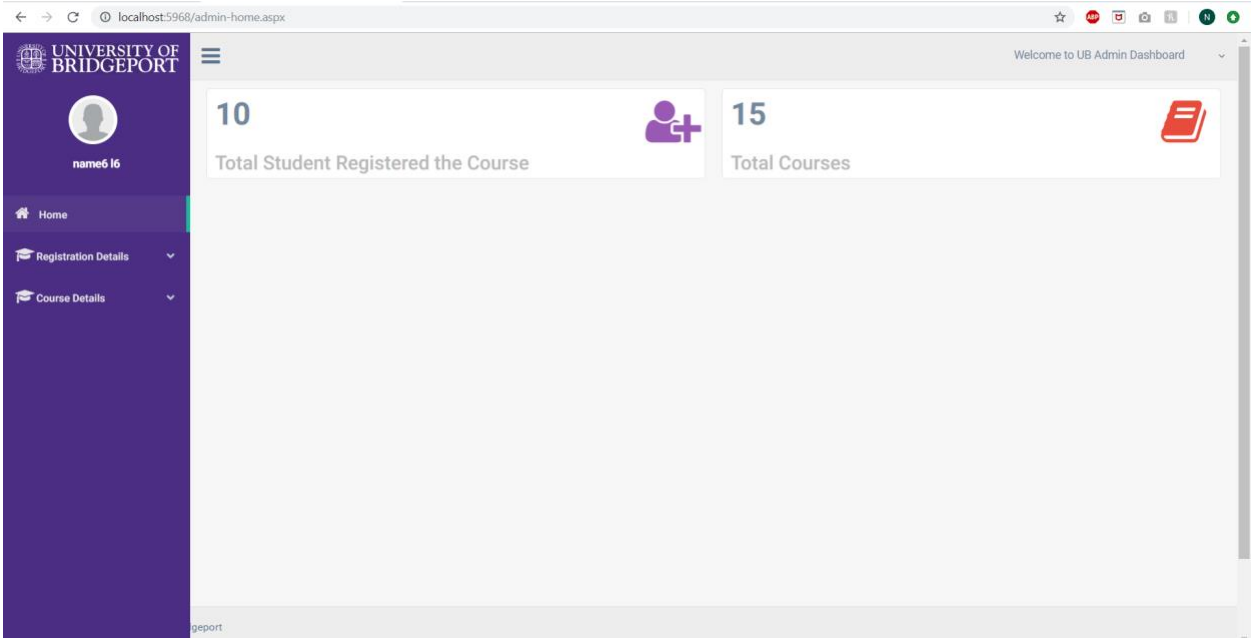
The codes are added in the zip folder.

Admin Dashboard:

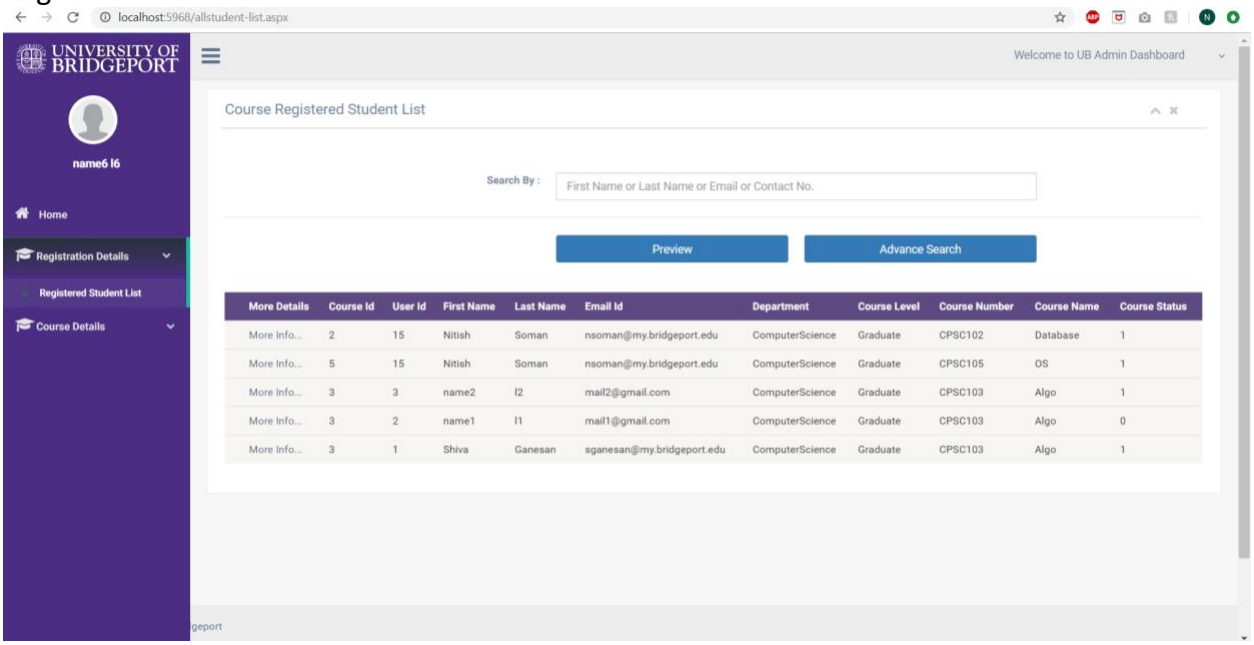
Login:



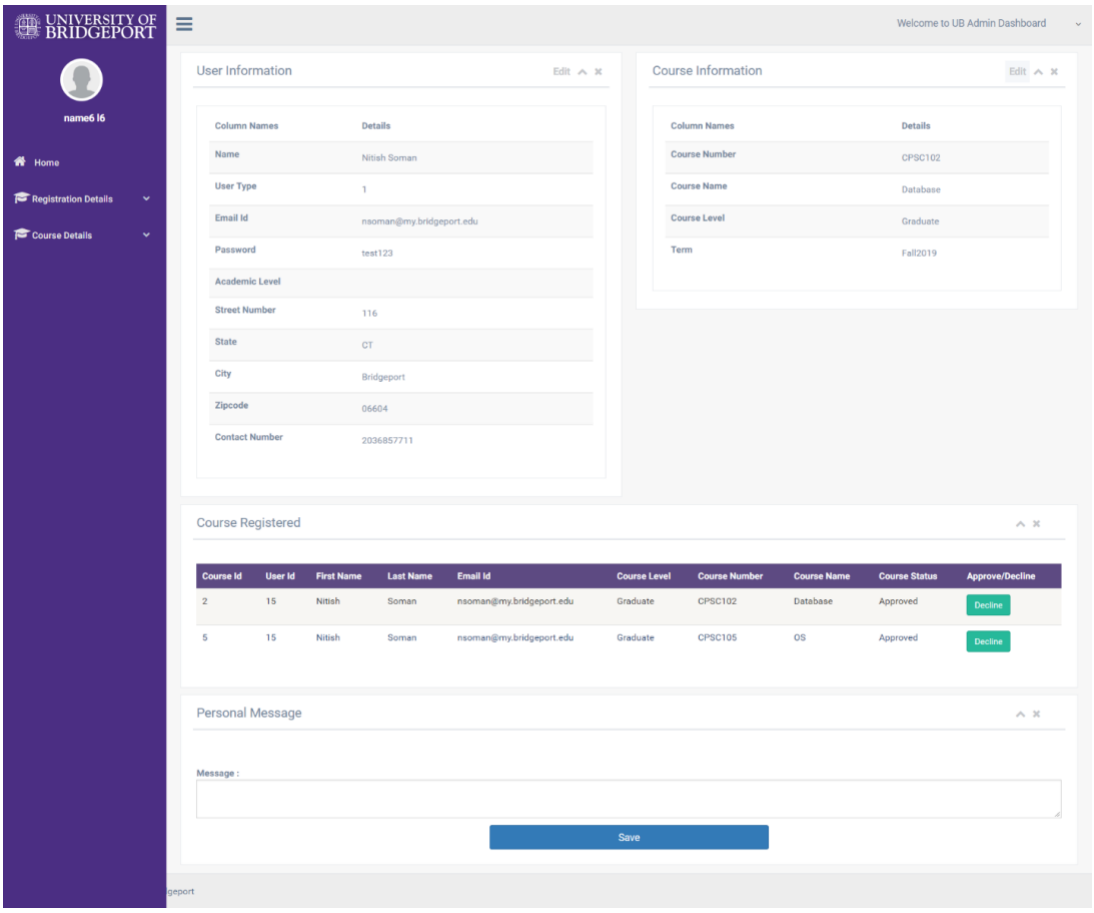
Home Page:



Registered Student List:

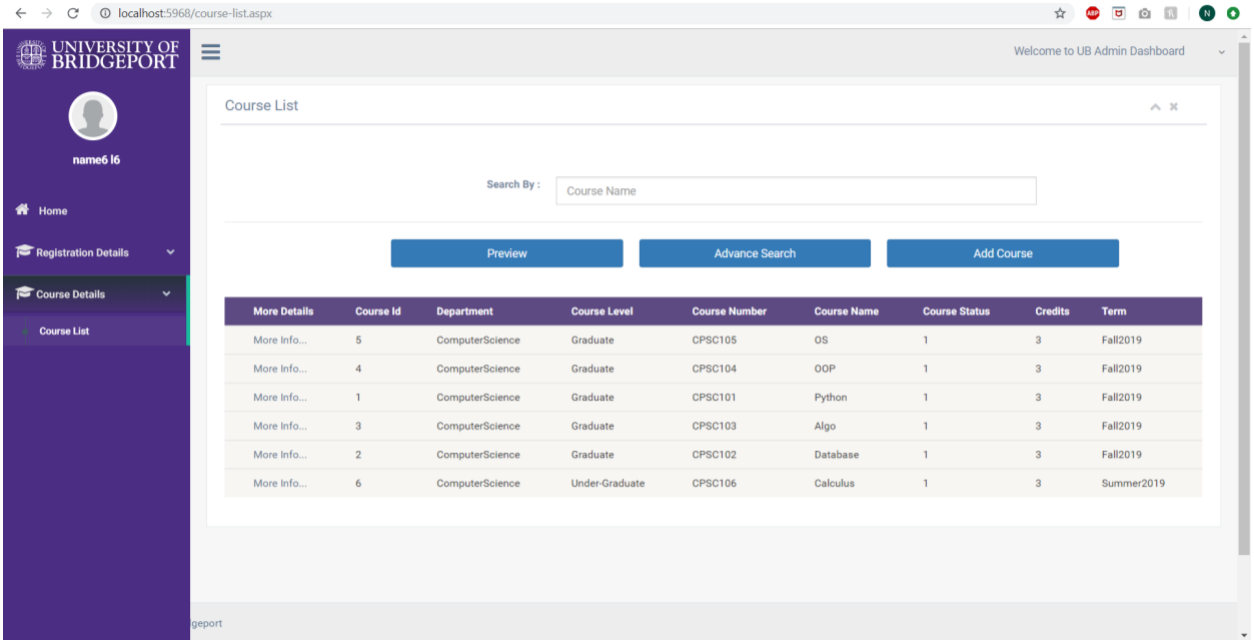


Each student Details:

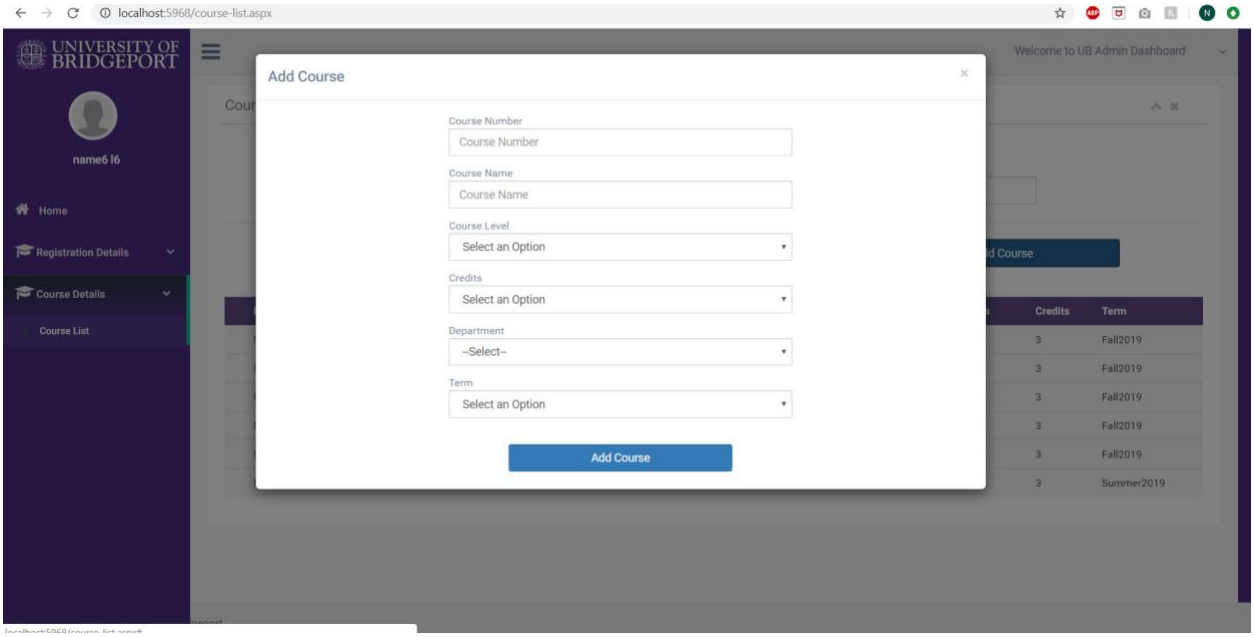


UB COURSE REGISTRATION SYSTEM

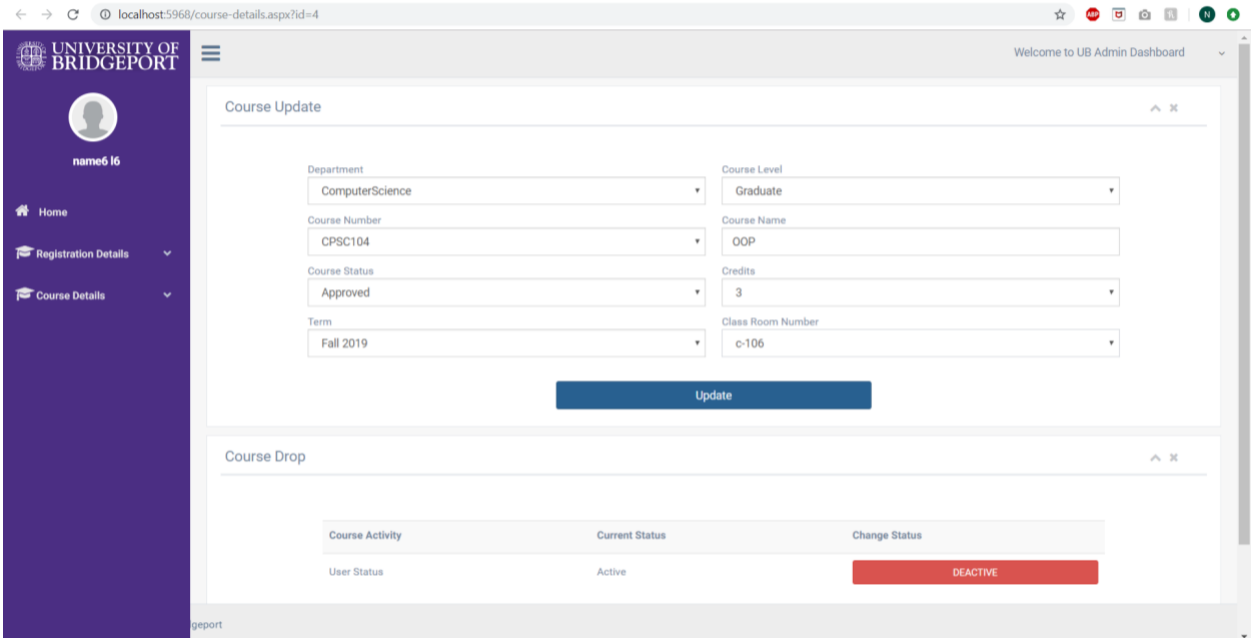
Course List and search:



Add new course:

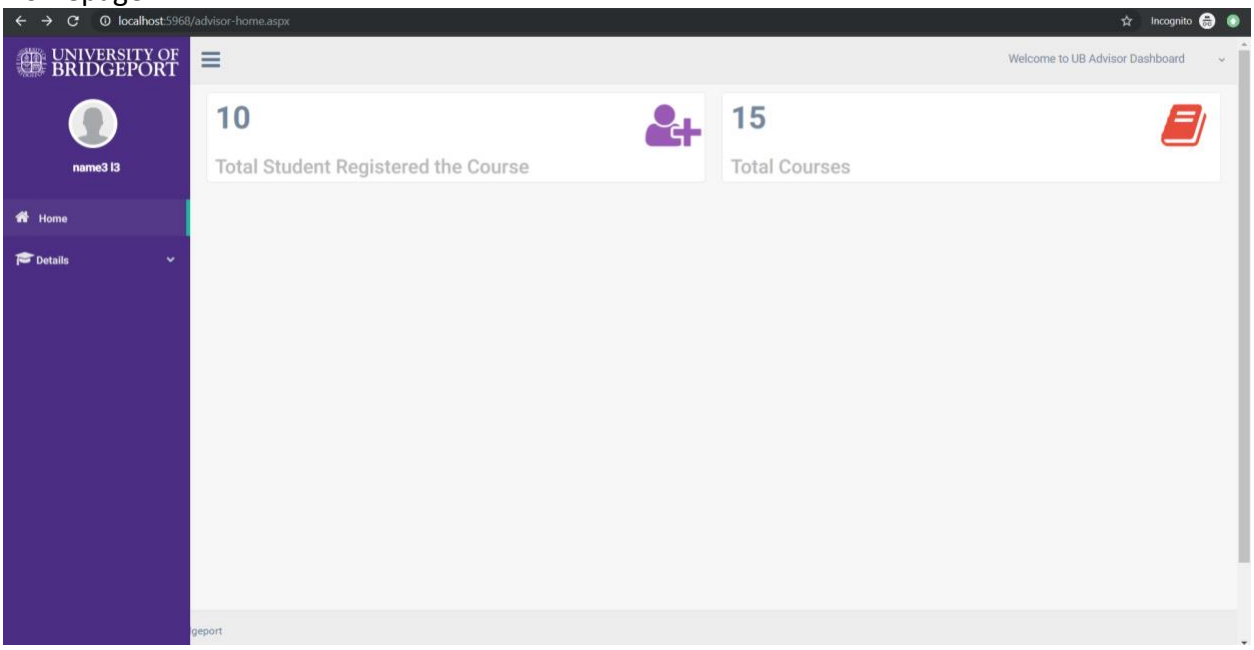


Each Course Details/ Update and drop course



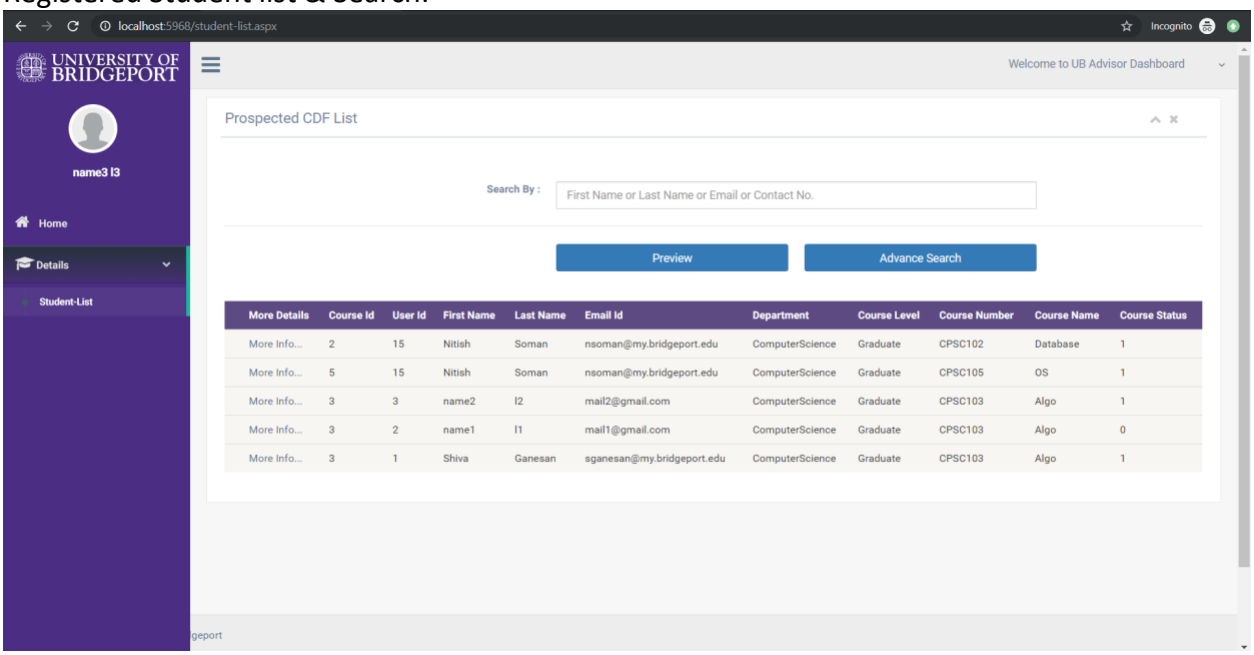
Advisor Dashboard:

Homepage:

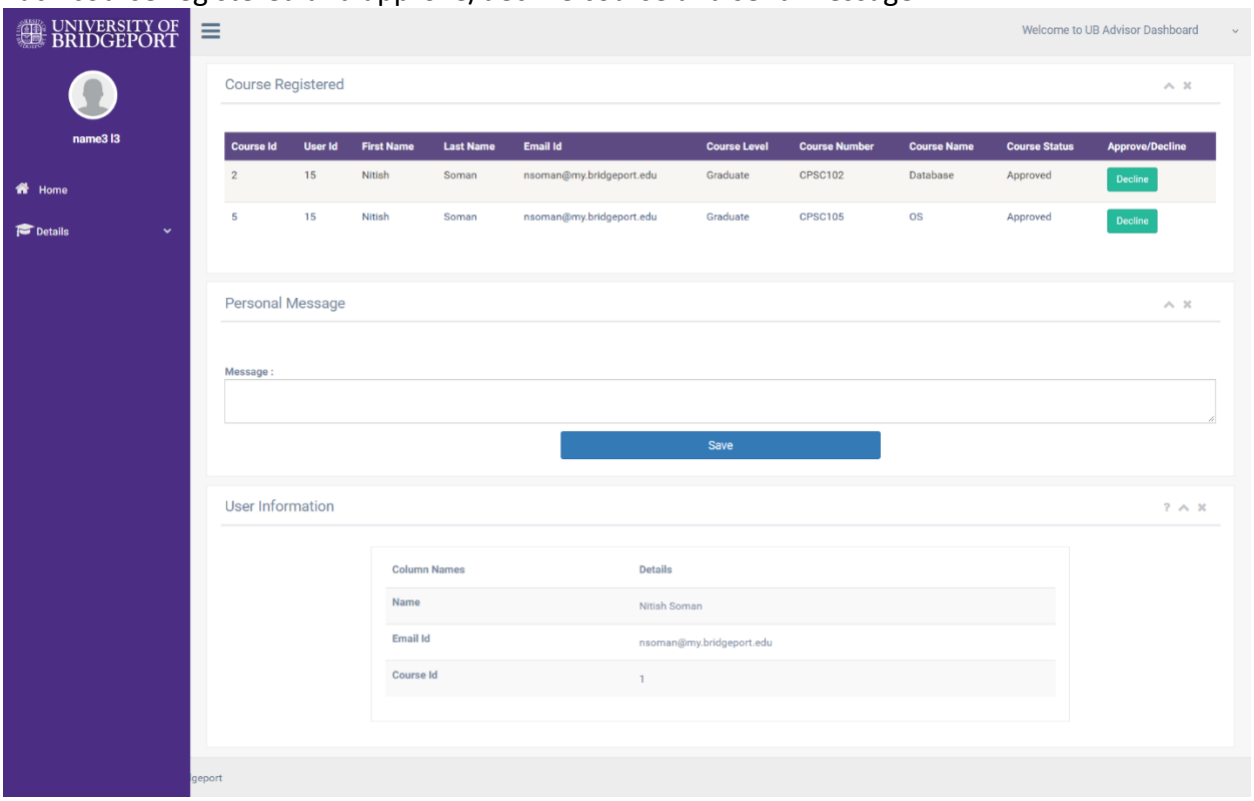


UB COURSE REGISTRATION SYSTEM

Registered Student list & Search:

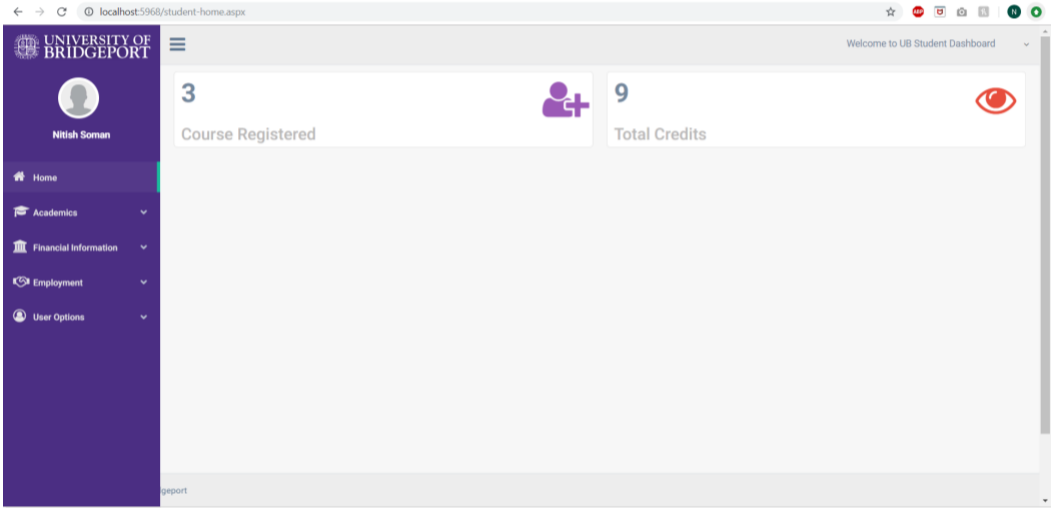


Each course registered and approve/decline course and send message

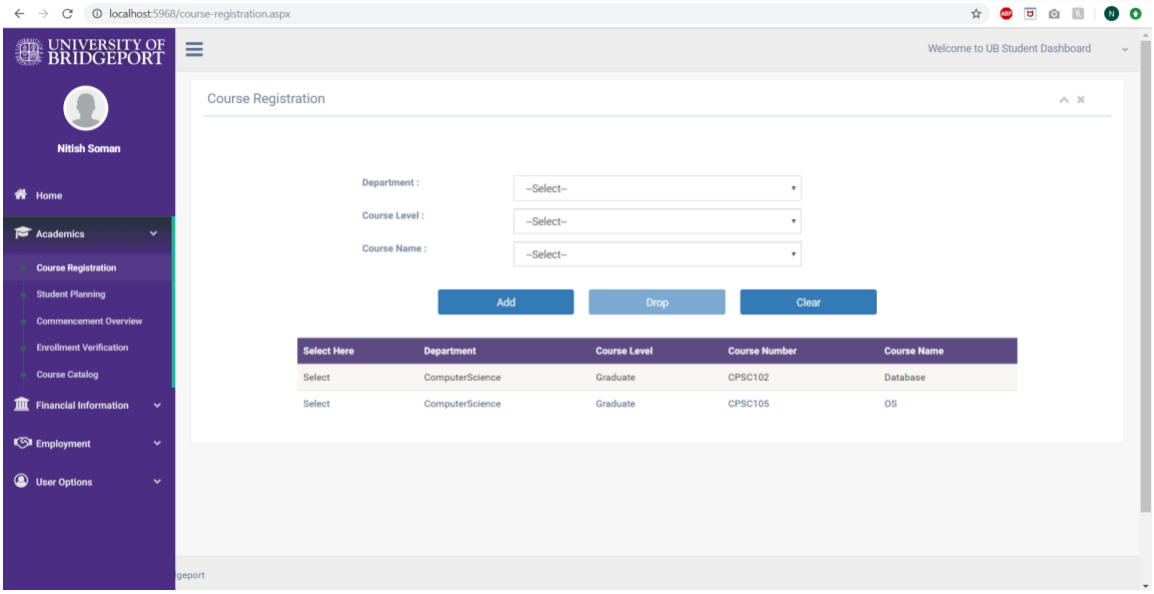


Student Dashboard:

Student homepage:



Course registration Page (add, drop):



CONCLUSION:

In this project we have learnt the following

1. GCP: Learnt how to operate and function within GCP
2. Database concepts
3. Web development
4. Integrating front end with database