

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.....	[2]
Abstract.....	2
Problem Statement.....	2
Queries.....	2
UML Diagram.....	3
Assumption.....	5
Phase 2.....	[6]
Entities and Attributes.....	6
Relationships.....	6
Week Entity.....	6
Generalization/Specializaion.....	6
EER Modeling.....	7
Relational Schema.....	8
Relational Algebra.....	9
Phase 3.....	[14]
Database server.....	14
Web Application Server.....	15
Phase 4.....	[15]
Script Details.....	15
Phase 5.....	[16]
Images.....	16
Conclusion.....	[21]

UB COURSE REGISTRATION SYSTEM

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.

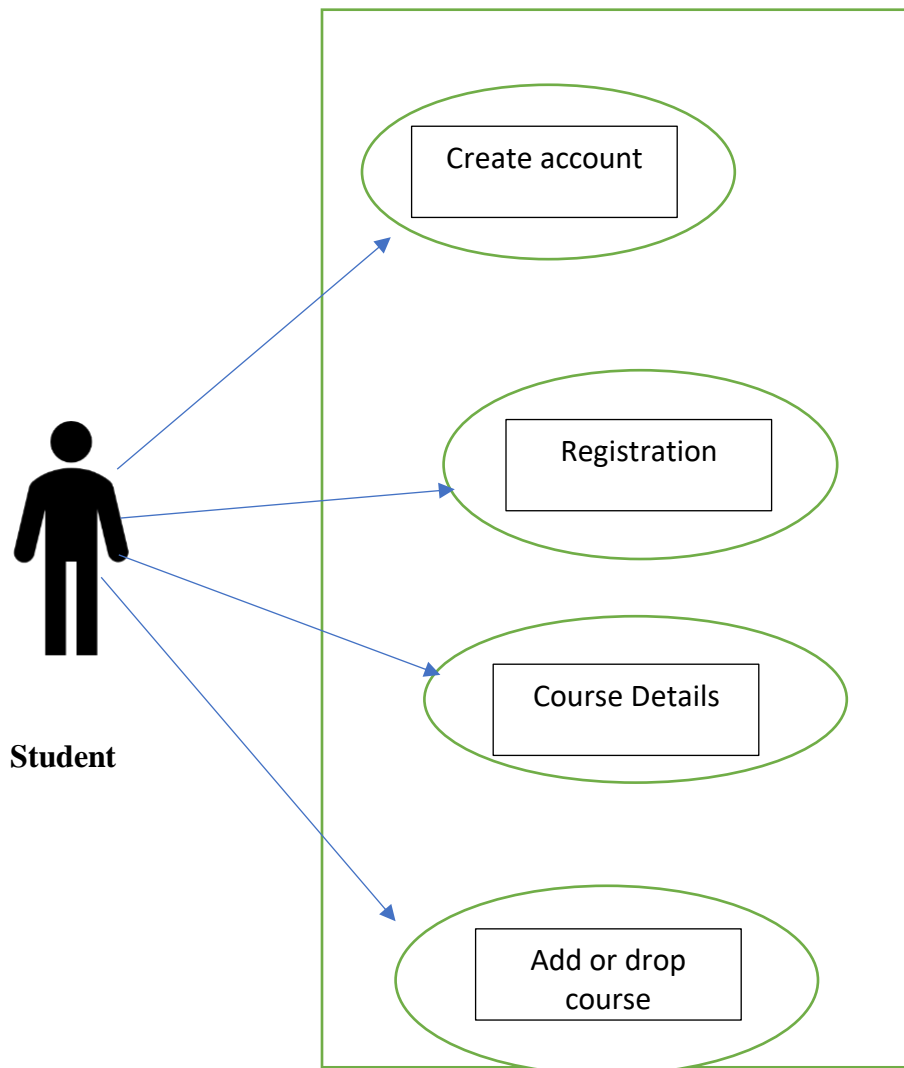
QUERIES:

- 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 userId, 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

UB COURSE REGISTRATION SYSTEM

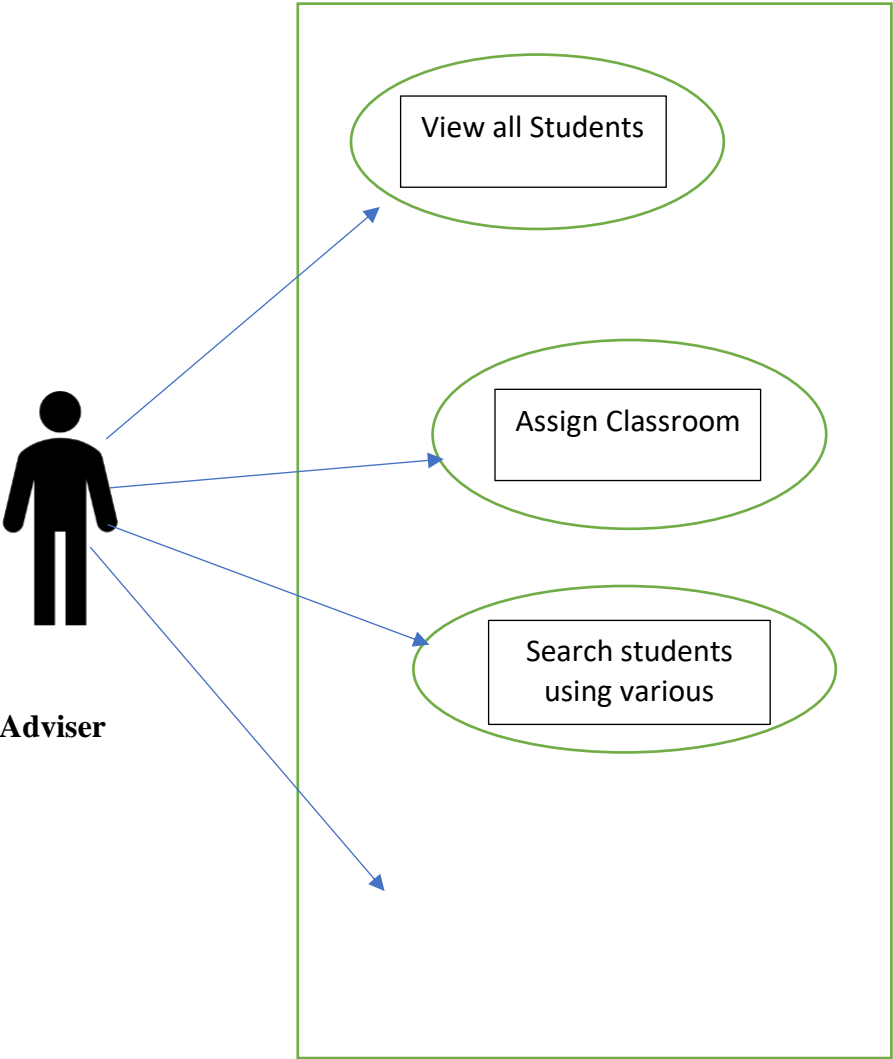
UML DIAGRAM:

Student:



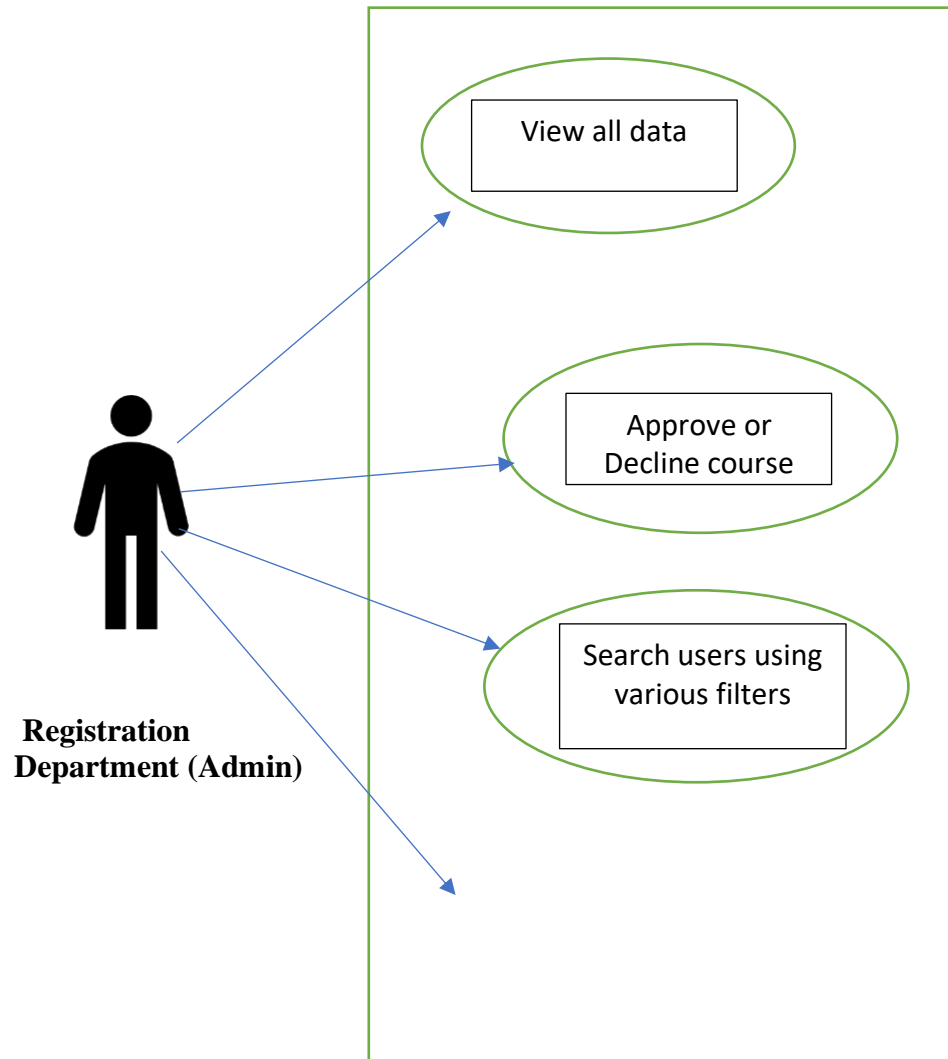
UB COURSE REGISTRATION SYSTEM

Adviser:



UB COURSE REGISTRATION SYSTEM

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:

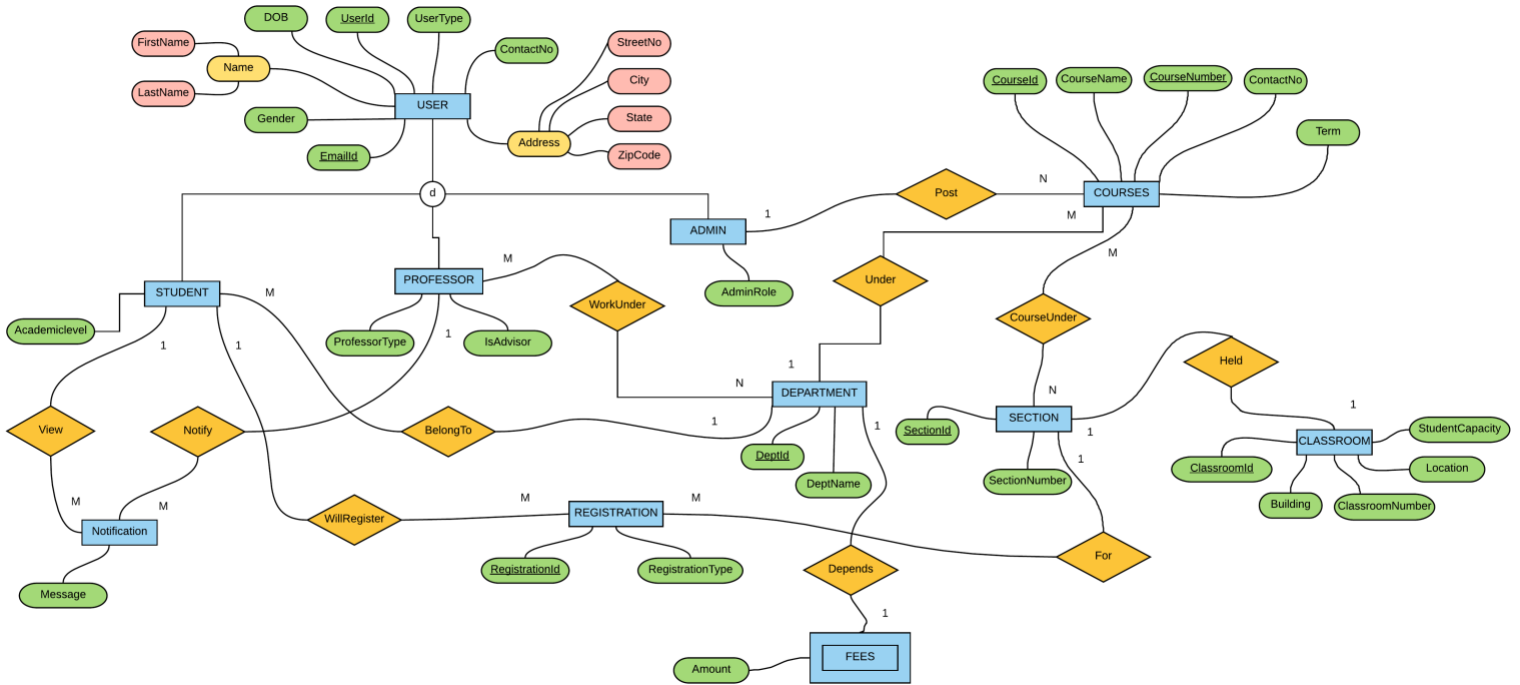
In our project we consider FEES as weak entity.

GENERALIZATION/SPECIALIZATION:

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

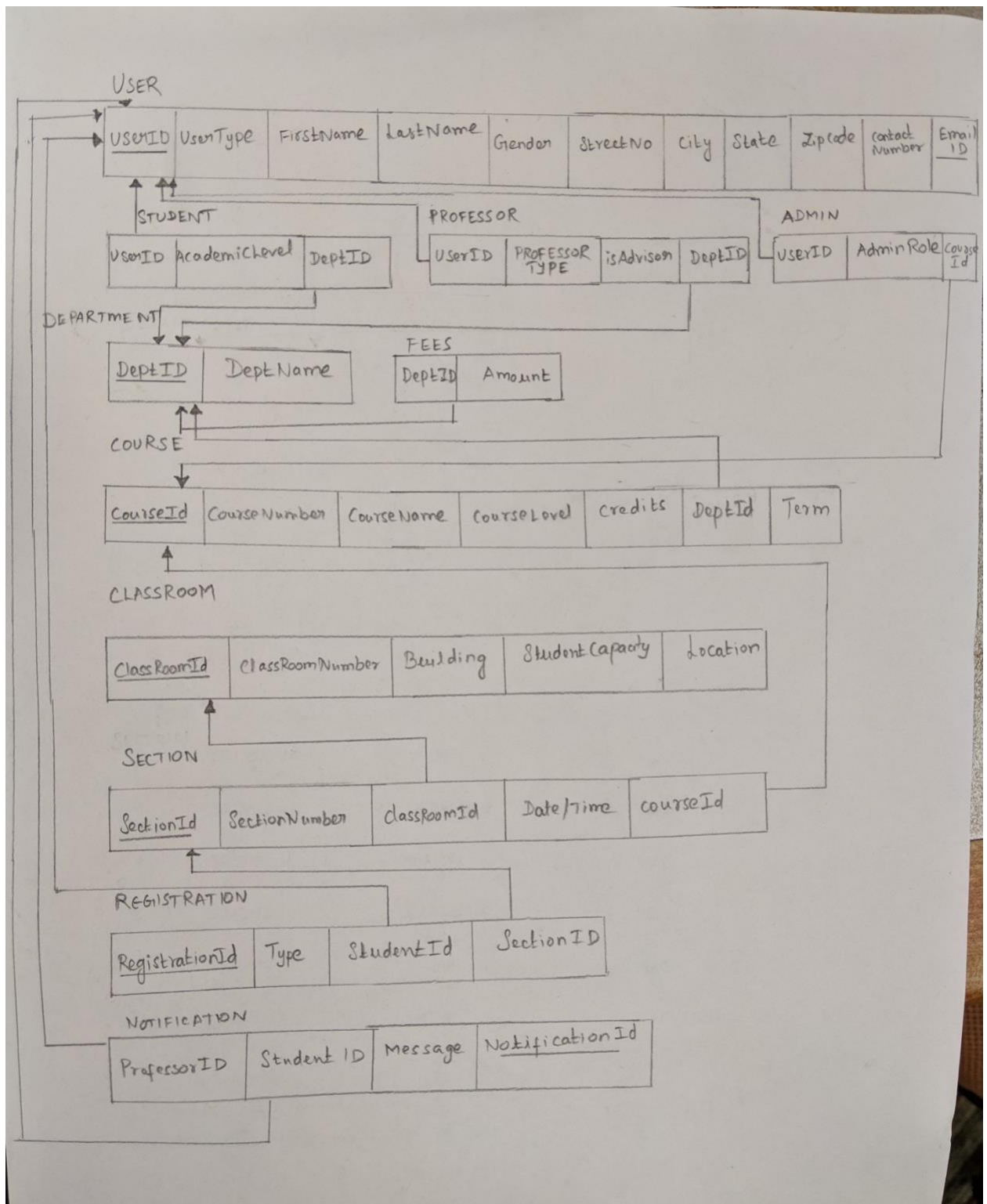
UB COURSE REGISTRATION SYSTEM

EER MODELING:



UB COURSE REGISTRATION SYSTEM

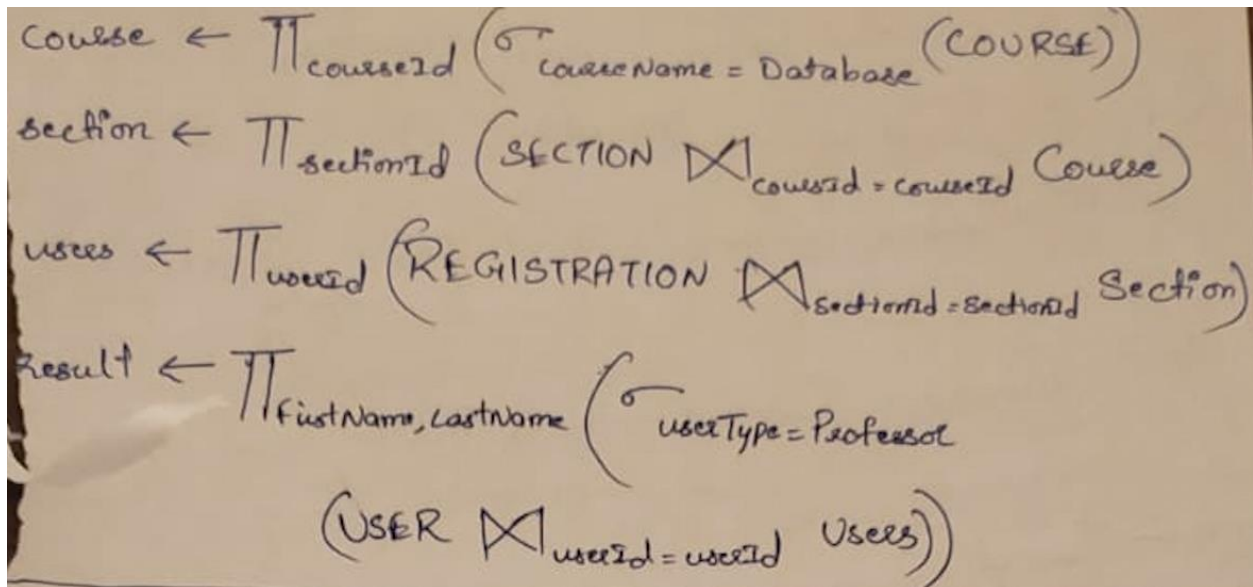
RELATIONAL SCHEMA:



UB COURSE REGISTRATION SYSTEM

RELATIONAL ALGEBRA:

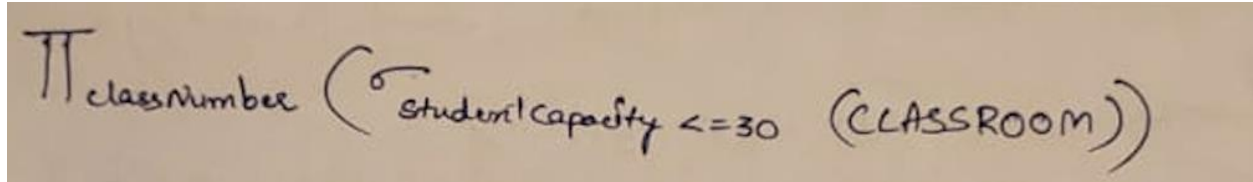
Query 1: Query to display all the professor's name who teaches "Database" course:



Handwritten Relational Algebra query for Query 1:

$$\begin{aligned} \text{course} &\leftarrow \Pi_{\text{courseId}} \left(\sigma_{\text{courseName} = \text{Database}} (\text{COURSE}) \right) \\ \text{section} &\leftarrow \Pi_{\text{sectionId}} \left(\text{SECTION} \bowtie_{\text{courseId} = \text{courseId}} \text{course} \right) \\ \text{users} &\leftarrow \Pi_{\text{userId}} \left(\text{REGISTRATION} \bowtie_{\text{sectionId} = \text{sectionId}} \text{section} \right) \\ \text{Result} &\leftarrow \Pi_{\text{firstName}, \text{lastName}} \left(\sigma_{\text{userType} = \text{Professor}} \right. \\ &\quad \left. (\text{USER} \bowtie_{\text{userId} = \text{userId}} \text{Users}) \right) \end{aligned}$$

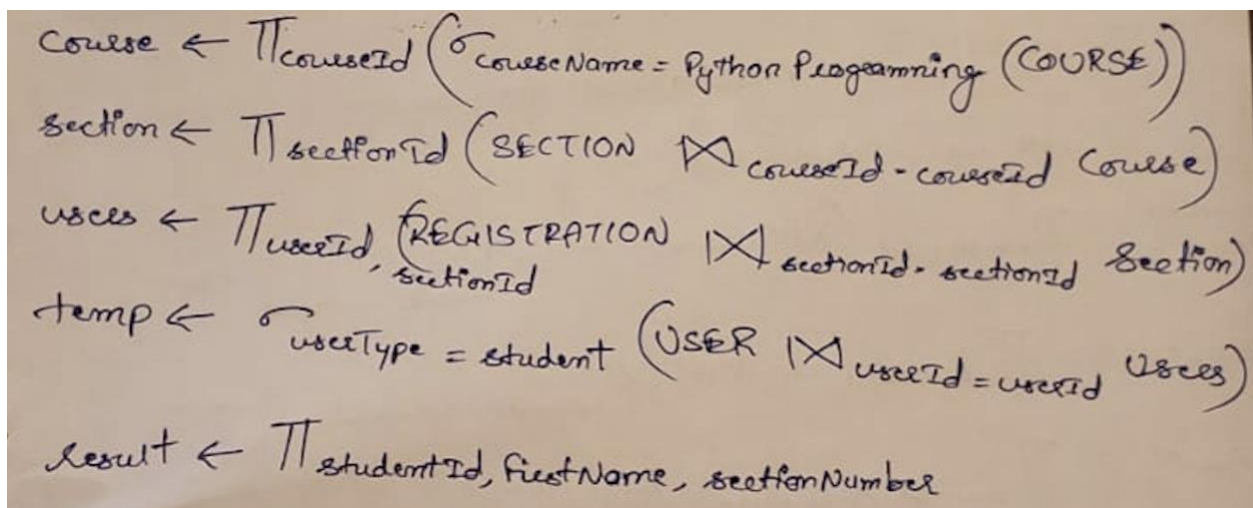
Query 2: Query to display all the rooms with capacity no more than 30 students



Handwritten Relational Algebra query for Query 2:

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

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



Handwritten Relational Algebra query for Query 3:

$$\begin{aligned} \text{course} &\leftarrow \Pi_{\text{courseId}} \left(\sigma_{\text{courseName} = \text{Python Programming}} (\text{COURSE}) \right) \\ \text{section} &\leftarrow \Pi_{\text{sectionId}} \left(\text{SECTION} \bowtie_{\text{courseId} = \text{courseId}} \text{course} \right) \\ \text{users} &\leftarrow \Pi_{\text{userId}} \left(\text{REGISTRATION} \bowtie_{\text{sectionId} = \text{sectionId}} \text{section} \right) \\ \text{temp} &\leftarrow \sigma_{\text{userType} = \text{student}} (\text{USER} \bowtie_{\text{userId} = \text{userId}} \text{Users}) \\ \text{result} &\leftarrow \Pi_{\text{studentId}, \text{firstName}, \text{sectionNumber}} \end{aligned}$$

UB COURSE REGISTRATION SYSTEM

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

department $\leftarrow \pi_{\text{DeptId}} (\sigma_{\text{DeptName} = \text{computer Science}})$
students $\leftarrow \sigma_{\text{userType} = \text{student}}$
result $\leftarrow \pi_{\text{userId}, \text{firstName}} (\text{department} \bowtie_{\text{DeptId} = \text{DeptId}} \text{students})$

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

count $\leftarrow \text{courseId} \text{ COUNT } \text{courseId} (\text{SECTION})$
temp $\leftarrow \sigma_{\text{course_count} > 1} (\text{count})$
result $\leftarrow \pi_{\text{courseId}, \text{CourseName}} (\text{COURSE} \bowtie_{\text{courseId} = \text{courseId}} \text{temp})$

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

user $\leftarrow \pi_{\text{userId}} (\sigma_{\text{FirstName} = \text{Shiva}} (\sigma_{\text{userType} = \text{student}} (\text{USER})))$
result $\leftarrow \pi_{\text{userId}, \text{Message}} (\text{NOTIFICATION} \bowtie_{\text{userId} = \text{studentId User}})$

UB COURSE REGISTRATION SYSTEM

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

```
user ←  $\Pi_{userId} (\sigma_{firstName = Shiva, lastName = Ganesan}$   
            $(\sigma_{userType = student (USER)})$ )  
Section ←  $\Pi_{sectionId} (REGISTRATION \bowtie_{userId = userId} user)$   
Courses ←  $\Pi_{courseId} (SECTION \bowtie_{courseId = courseId} Section)$   
result ←  $\Pi_{courseId, courseName} (COURSE \bowtie_{courseId = courseId} Courses)$ 
```

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

```
classrooms ←  $\Pi_{classroomId} (\sigma_{classNumber = 203, building = CarlsonHall}$   
               $(CLASSROOM)$ )  
Sections ←  $\Pi_{courseId} (SECTION \bowtie_{classroomId = classroomId} classrooms)$   
result ←  $\Pi_{courseId, courseName} (COURSE \bowtie_{courseId = courseId} Sections)$ 
```

Query 9: Courses which are taught as Distance Learning:

```
type ←  $\Pi_{sectionId} (\sigma_{type = DL} (REGISTRATION))$   
Courses ←  $\Pi_{courseId} (SECTION \bowtie_{sectionId = sectionId} type)$   
result ←  $\Pi_{courseId, courseName} (COURSE \bowtie_{courseId = courseId} Courses)$ 
```


UB COURSE REGISTRATION SYSTEM

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

$$\text{count} \leftarrow \text{deptId } f_{\text{count deptId}} (\text{COURSE})$$
$$\text{result} \leftarrow \Pi_{\text{deptName, count-dept}} \left(\text{DEPARTMENT} \bowtie_{\substack{\text{deptId} \\ = \text{deptId}}} \text{count} \right)$$

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

$$\text{course type} \leftarrow \Pi_{\text{userId}} \left(\sigma_{\text{Type}=\text{DL}} (\text{REGISTRATION}) \right)$$
$$\text{result} \leftarrow \Pi_{\text{userId, firstName}} \left(\sigma_{\text{usertype}=\text{Student}} \left(\text{USER} \bowtie_{\text{userId}=\text{userId}} \text{course type} \right) \right)$$

Query 12: Department names whose fees is more than 800\$:

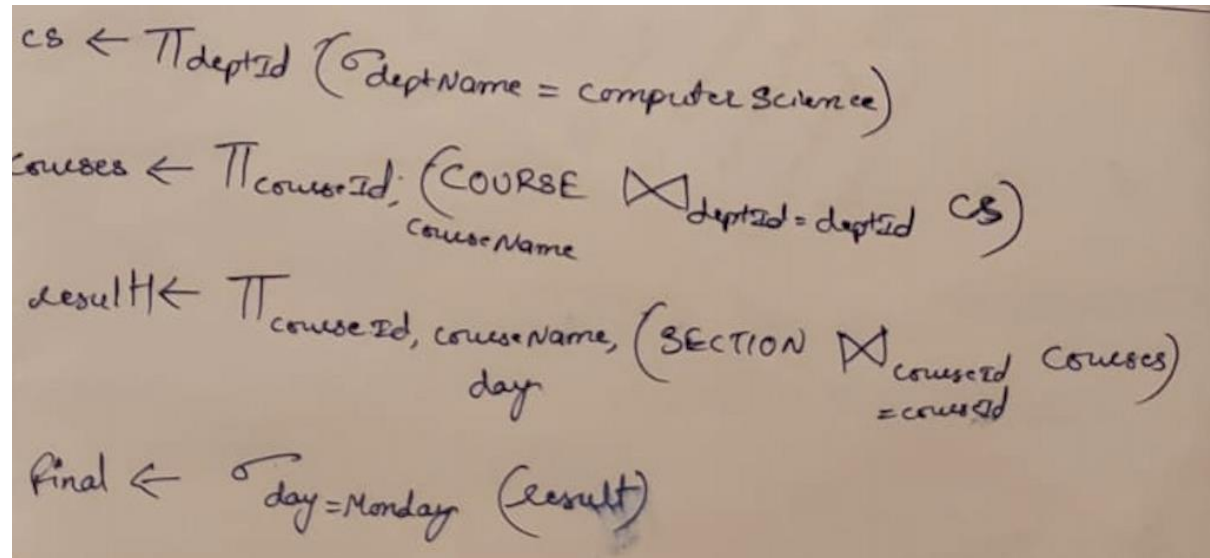
$$\text{departments} \leftarrow \Pi_{\text{DeptId}} \left(\sigma_{\text{Amount} > 800} (\text{FEES}) \right)$$
$$\text{result} \leftarrow \Pi_{\text{deptID, DeptName}} \left(\text{DEPARTMENT} \bowtie_{\substack{\text{deptId} \\ = \text{deptId}}} \text{departments} \right)$$

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

$$\text{credits} \leftarrow \Pi_{\text{courseId, courseName, deptId}} \left(\sigma_{\text{credits} = 1} (\text{COURSE}) \right)$$
$$\text{Result} \leftarrow \Pi_{\text{courseId, courseName, deptName}} \left(\text{DEPARTMENT} \bowtie_{\text{deptId} = \text{DeptId}} \text{Credits} \right)$$

UB COURSE REGISTRATION SYSTEM

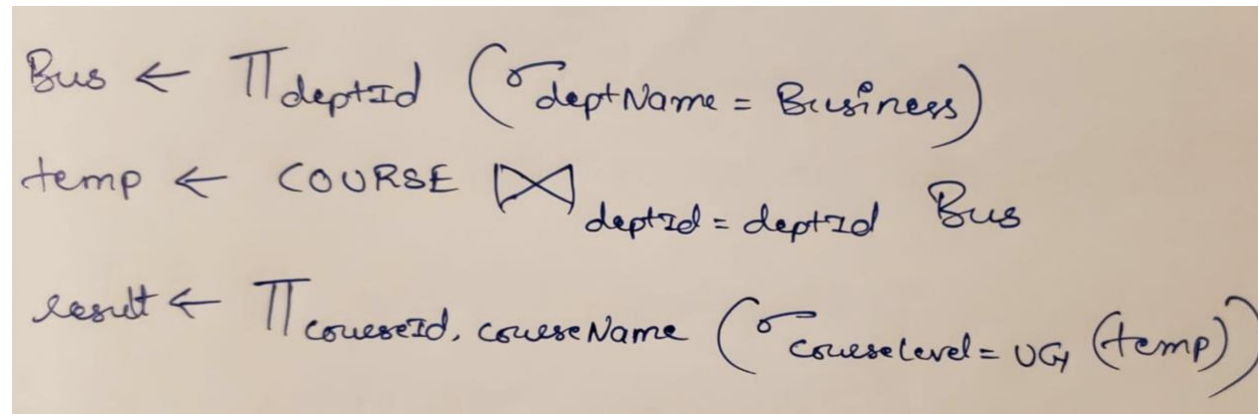
Query 14: Courses of Computer science department conducted on Mondays:



Handwritten SQL query for Query 14:

```
CS ← πdeptId (σdeptName = Computer Science)  
courses ← πcourseId, courseName (COURSE ⋈deptId = deptId CS)  
resultH ← πcourseId, courseName, day (SECTION ⋈courseId = courseId courses)  
Final ← σday = Monday (result)
```

Query 15: Undergrad courses names of Business department:



Handwritten SQL query for Query 15:

```
Bus ← πdeptId (σdeptName = Business)  
temp ← COURSE ⋈deptId = deptId Bus  
result ← πcourseId, courseName (σcourseLevel = UG (temp))
```

UB COURSE REGISTRATION SYSTEM

PHASE 3:

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

DATABASE SERVER

The screenshot displays the Google Cloud Platform console interface for a MySQL instance named 'project-adb'. The top navigation bar shows 'Google Cloud Platform' and 'ADB-Group11'. The left sidebar lists the instance details. The main content area shows the 'Instance details' tab, which includes a 'Connect to this instance' button, a 'CPU utilization' graph, and a 'Configuration' section. The 'Configuration' section lists various settings such as 'vCPUs', 'Memory', 'HDD storage', 'Database version', 'Auto storage increase', 'Automated backups', 'Binary logging', 'Location', 'No database flags', 'No labels', and 'Not highly available (zonal)'. The 'Operations and logs' section shows a table of backup operations with columns for 'Date/Time', 'Type', and 'Status'.

Instance details

project-adb
MySQL Second Generation master

Connect to this SQL instance from a Compute Engine VM. [Dismiss] [Start]

OVERVIEW CONNECTIONS USERS DATABASES BACKUPS REPLICAS OPERATIONS

CPU utilization 1 hour 6 hours 12 hours 1 day 2 days 4 days 7 days 14 days 30 days

Apr 12, 2019 5:51 PM

100%
80%
60%
40%
20%
0

5:05 5:10 5:15 5:20 5:25 5:30 5:35 5:40 5:45 5:50 5:55 6 PM

● CPU utilization (project-adb): 2%

Connect to this instance

Public IP address
35.231.241.254

Instance connection name
adb-ganesan-shivashankar:us-east1:project-adb

Connect using Cloud Shell

Connect from a Compute Engine VM instance

See all connection methods

Suggested actions

→ Create failover replica (enable high availability)

Service account

dt5mqdybznsrk5y7co7y04ya@speckle-umbrella-25.iam.gserviceaccount.com

Configuration

vCPUs 1 Memory 614 MB HDD storage 10 GB

Database version is MySQL 5.7

Auto storage increase is enabled

Automated backups are enabled

Binary logging is enabled

Located in us-east1-b

No database flags set

No labels set

Not highly available (zonal)

→ Edit configuration

Maintenance schedule

Maintenance window

Updates may occur any day of the week

Maintenance timing

Cloud SQL chooses the maintenance timing

→ Edit maintenance schedule

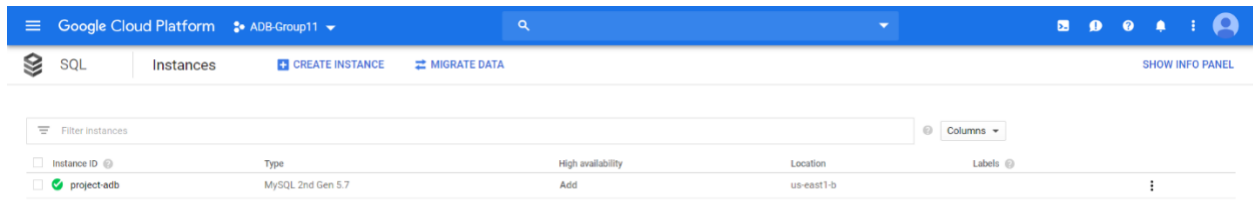
Operations and logs

Date/Time	Type	Status
Apr 12, 2019, 1:29:24 PM	Backup	Backup finished
Apr 11, 2019, 11:48:46 AM	Backup	Backup finished
Apr 10, 2019, 12:48:36 PM	Backup	Backup finished
Apr 9, 2019, 1:25:29 PM	Backup	Backup finished
Apr 8, 2019, 11:28:00 AM	Backup	Backup finished

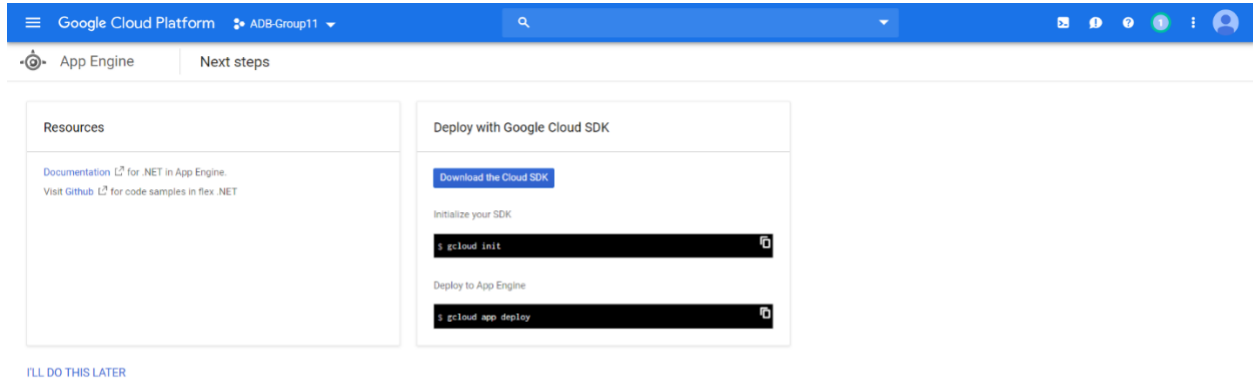
→ View all operations

→ View MySQL error logs

UB COURSE REGISTRATION SYSTEM



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

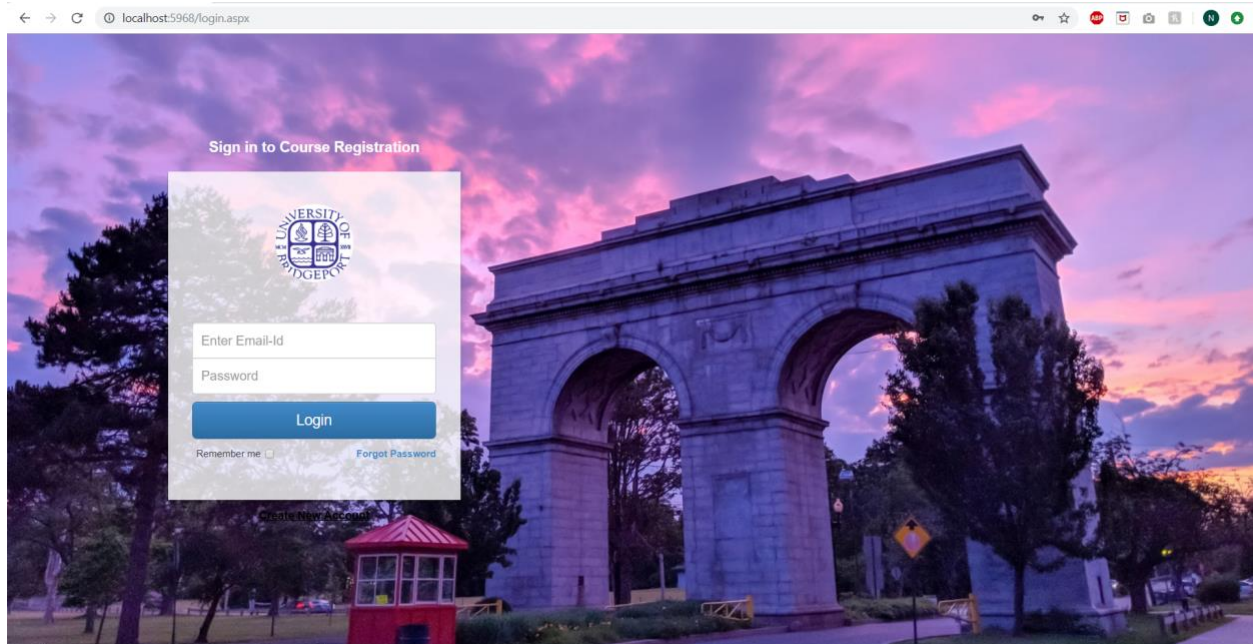
UB COURSE REGISTRATION SYSTEM

PHASE 5:

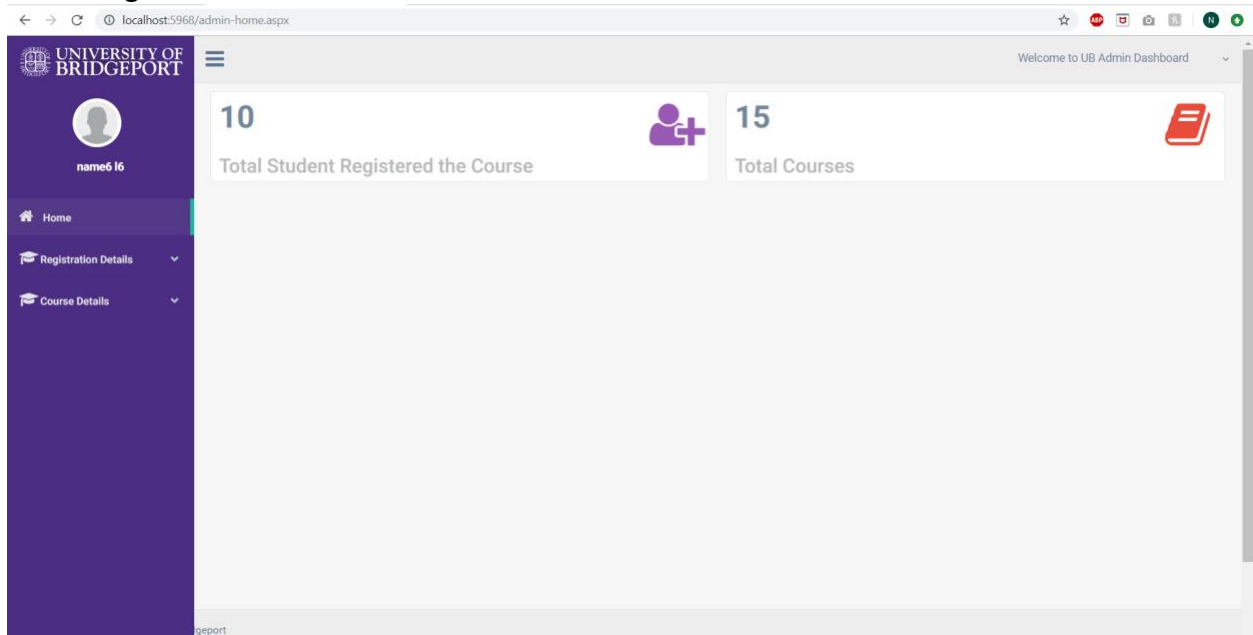
The codes are added in the zip folder.

Admin Dashboard:

Login:



Home Page:



UB COURSE REGISTRATION SYSTEM

Registered Student List:

Course Registered Student List

Search By:

[Preview](#) [Advance Search](#)

More Details	Course Id	User Id	First Name	Last Name	Email Id	Department	Course Level	Course Number	Course Name	Course Status
More Info...	2	15	Nitish	Soman	nsoman@my.bridgeport.edu	ComputerScience	Graduate	CPSC102	Database	1
More Info...	5	15	Nitish	Soman	nsoman@my.bridgeport.edu	ComputerScience	Graduate	CPSC105	OS	1
More Info...	3	3	name2	I2	mail2@gmail.com	ComputerScience	Graduate	CPSC103	Algo	1
More Info...	3	2	name1	I1	mail1@gmail.com	ComputerScience	Graduate	CPSC103	Algo	0
More Info...	3	1	Shiva	Ganesan	sganesan@my.bridgeport.edu	ComputerScience	Graduate	CPSC103	Algo	1

Each student Details:

User Information [Edit](#)

Column Names	Details
Name	Nitish Soman
User Type	1
Email Id	nsoman@my.bridgeport.edu
Password	test123
Academic Level	
Street Number	116
State	CT
City	Bridgeport
Zipcode	06604
Contact Number	2036857711

Course Information [Edit](#)

Column Names	Details
Course Number	CPSC102
Course Name	Database
Course Level	Graduate
Term	Fall2019

Course Registered

Course Id	User Id	First Name	Last Name	Email Id	Course Level	Course Number	Course Name	Course Status	Approve/Decline
2	15	Nitish	Soman	nsoman@my.bridgeport.edu	Graduate	CPSC102	Database	Approved	Decline
5	15	Nitish	Soman	nsoman@my.bridgeport.edu	Graduate	CPSC105	OS	Approved	Decline

Personal Message [Save](#)

Message :

UB COURSE REGISTRATION SYSTEM

Course List and search:

The screenshot shows the 'Course List' page in the University of Bridgeport Admin Dashboard. The page features a sidebar with navigation links: Home, Registration Details, Course Details, and Course List. The main content area displays a search bar with 'Course Name' entered, and three buttons: Preview, Advance Search, and Add Course. Below these is a table listing courses with columns for More Details, Course Id, Department, Course Level, Course Number, Course Name, Course Status, Credits, and Term.

More Details	Course Id	Department	Course Level	Course Number	Course Name	Course Status	Credits	Term
More Info...	5	ComputerScience	Graduate	CPSC105	OS	1	3	Fall2019
More Info...	4	ComputerScience	Graduate	CPSC104	OOP	1	3	Fall2019
More Info...	1	ComputerScience	Graduate	CPSC101	Python	1	3	Fall2019
More Info...	3	ComputerScience	Graduate	CPSC103	Algo	1	3	Fall2019
More Info...	2	ComputerScience	Graduate	CPSC102	Database	1	3	Fall2019
More Info...	6	ComputerScience	Under-Graduate	CPSC106	Calculus	1	3	Summer2019

Add new course:

The screenshot shows the 'Add Course' modal form in the University of Bridgeport Admin Dashboard. The form is overlaid on the Course List page. It contains input fields for Course Number, Course Name, Course Level (a dropdown menu), Credits (a dropdown menu), Department (a dropdown menu), and Term (a dropdown menu). An 'Add Course' button is located at the bottom of the modal.

UB COURSE REGISTRATION SYSTEM

Each Course Details/ Update and drop course

The screenshot displays the 'UB Admin Dashboard' interface. On the left is a purple sidebar with the University of Bridgeport logo and navigation links: Home, Registration Details, and Course Details. The main content area is titled 'Welcome to UB Admin Dashboard'. It features two sections: 'Course Update' and 'Course Drop'. The 'Course Update' section contains several dropdown menus for Department (ComputerScience), Course Level (Graduate), Course Number (CPSC104), Course Name (OOP), Course Status (Approved), Credits (3), Term (Fall 2019), and Class Room Number (c-106). Below these is a blue 'Update' button. The 'Course Drop' section shows a table with columns for Course Activity, Current Status, and Change Status. The table has one row with 'User Status' and 'Active', and a red 'DEACTIVE' button.

Course Activity	Current Status	Change Status
User Status	Active	DEACTIVE

Advisor Dashboard:

Homepage:

The screenshot shows the 'UB Advisor Dashboard' homepage. The left sidebar is purple with the University of Bridgeport logo and navigation links: Home and Details. The main content area is titled 'Welcome to UB Advisor Dashboard'. It features two large white boxes with purple and red icons. The first box shows '10 Total Student Registered the Course' with a purple icon of two people. The second box shows '15 Total Courses' with a red icon of a book. Below these boxes is a large empty white space.

UB COURSE REGISTRATION SYSTEM

Registered Student list & Search:

The screenshot displays the 'Prospected CDF List' interface. On the left is a sidebar with the University of Bridgeport logo, a user profile icon labeled 'name3 13', and navigation links for 'Home', 'Details', and 'Student-List'. The main content area has a header 'Welcome to UB Advisor Dashboard' and a sub-header 'Prospected CDF List'. Below this is a search bar with the placeholder 'First Name or Last Name or Email or Contact No.' and two buttons: 'Preview' and 'Advance Search'. A table lists five prospective students with columns for 'More Details', 'Course Id', 'User Id', 'First Name', 'Last Name', 'Email Id', 'Department', 'Course Level', 'Course Number', 'Course Name', and 'Course Status'.

More Details	Course Id	User Id	First Name	Last Name	Email Id	Department	Course Level	Course Number	Course Name	Course Status
More Info...	2	15	Nitish	Soman	nsoman@my.bridgeport.edu	ComputerScience	Graduate	CPSC102	Database	1
More Info...	5	15	Nitish	Soman	nsoman@my.bridgeport.edu	ComputerScience	Graduate	CPSC105	OS	1
More Info...	3	3	name2	I2	mail2@gmail.com	ComputerScience	Graduate	CPSC103	Algo	1
More Info...	3	2	name1	I1	mail1@gmail.com	ComputerScience	Graduate	CPSC103	Algo	0
More Info...	3	1	Shiva	Ganesan	sganesan@my.bridgeport.edu	ComputerScience	Graduate	CPSC103	Algo	1

Each course registered and approve/decline course and send message

The screenshot shows the 'Course Registered' section of the dashboard. The sidebar is identical to the previous view. The main content area has a header 'Welcome to UB Advisor Dashboard' and a sub-header 'Course Registered'. Below this is a table with columns: 'Course Id', 'User Id', 'First Name', 'Last Name', 'Email Id', 'Course Level', 'Course Number', 'Course Name', 'Course Status', and 'Approve/Decline'. Two courses are listed, both with 'Approved' status and a 'Decline' button. Below the table is a 'Personal Message' section with a text input field and a 'Save' button. At the bottom is a 'User Information' section containing a table with user details.

Course Id	User Id	First Name	Last Name	Email Id	Course Level	Course Number	Course Name	Course Status	Approve/Decline
2	15	Nitish	Soman	nsoman@my.bridgeport.edu	Graduate	CPSC102	Database	Approved	<button>Decline</button>
5	15	Nitish	Soman	nsoman@my.bridgeport.edu	Graduate	CPSC105	OS	Approved	<button>Decline</button>

Personal Message

Message :

Save

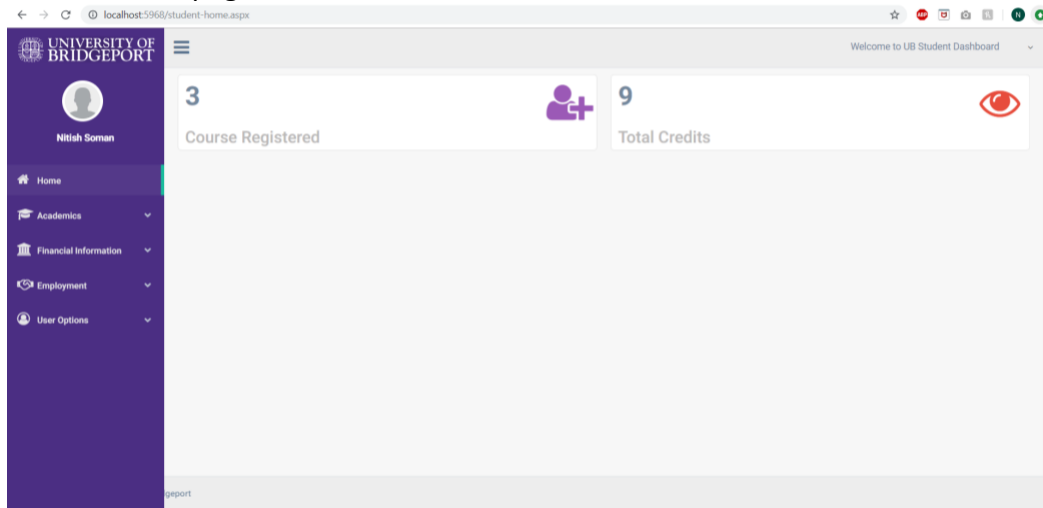
User Information

Column Names	Details
Name	Nitish Soman
Email Id	nsoman@my.bridgeport.edu
Course Id	1

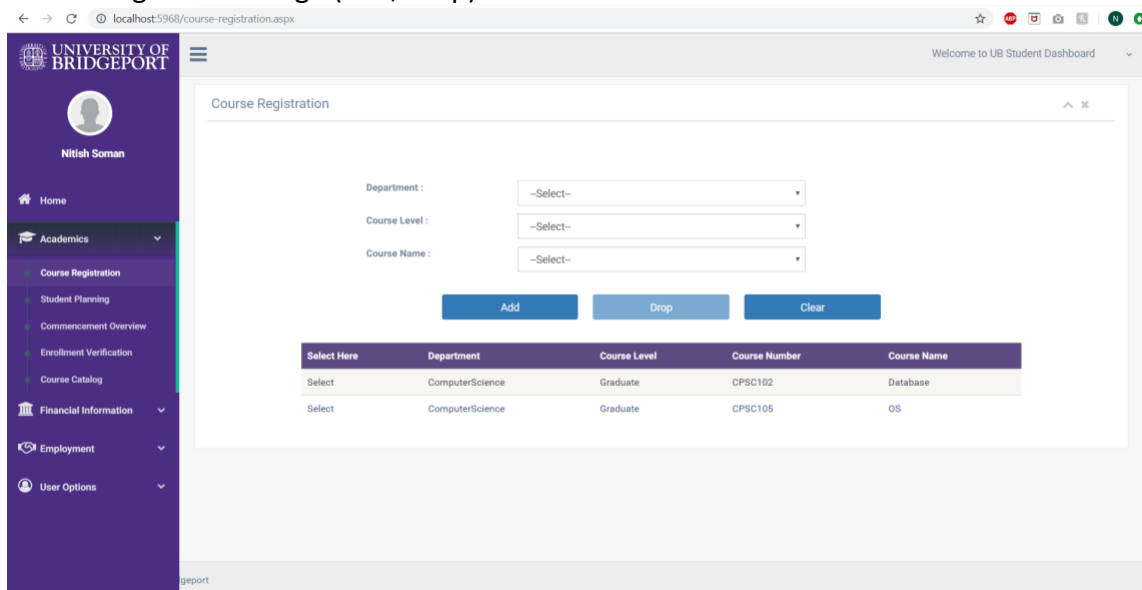
UB COURSE REGISTRATION SYSTEM

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