Table of Contents

[**University Web Based Registration System** 2](#_Toc82731009)

[**Purpose & Problem Statement** 4](#_Toc82731010)

[**Homepage** 4](#_Toc82731011)

[**Functional Requirements** 4](#_Toc82731012)

[**Non-Functional Requirements** 4](#_Toc82731013)

[**Performance Requirements** 5](#_Toc82731014)

[**Reliability** 5](#_Toc82731015)

[**Availability** 5](#_Toc82731016)

[**Security** 5](#_Toc82731017)

[**Business Rules** 5](#_Toc82731018)

[**General User Business rules** 5](#_Toc82731019)

[**Student user Business Rules:** 6](#_Toc82731020)

[**Faculty User Business Rules** 7](#_Toc82731021)

[**Admin User Business Rules** 8](#_Toc82731022)

[**Researcher User Business Rules** 8](#_Toc82731023)

[**Entities & Attributes of Entities** 9](#_Toc82731024)

[**Relationships:** 10](#_Toc82731025)

[**Relational Schema:** 12](#_Toc82731026)

[**Entity Relationship Diagram (EERD)** 14](#_Toc82731027)

[**Use Cases:** 16](#_Toc82731028)

[**General User Use Cases:** 16](#_Toc82731029)

[**Student User Use Cases:** 17](#_Toc82731030)

[**Researcher:** 21](#_Toc82731031)

[**Faculty Professor** 22](#_Toc82731032)

[**Administrator** 23](#_Toc82731033)

[**Technologies to be use** 26](#_Toc82731034)

[**Front-End Technology** 26](#_Toc82731035)

[**Back-End Technology** 26](#_Toc82731036)

[**Database:** 26](#_Toc82731037)

[**Feasibility:** 26](#_Toc82731038)

[**Time Constraints** 26](#_Toc82731039)

[**Task Breakdown:** 27](#_Toc82731040)

# 

# **New York University Web Based Registration System**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Meeting Log** | | | | | | |
| **Date & Time** | **Meeting Platform** | **Attendance** | | | **Topic** | **Description** |
|  |  | **Saad kirmani** | **Zaid Khan** | **Member 3** |  |  |
| 9/8/21 2:30 PM | Zoom online Meeting App | P | P | P | Purpose & Problem Statement Gathering Requirements | Discussing Purpose and defining Problem Statement about |
| 9/9/21 2:30 PM | Zoom online Meeting App | P | P | P | Business Rules Functional Requirements Non-Functional Requirements | Defining Business Rules and identifying functional and Non-Functional Requirements |
| 9/10/21 2:30 PM | Zoom online Meeting App | P | P | P | Use Cases Software Usage | Identifying Use Cases from requirement gathered and usage of software |
| 9/11/21 2:30 PM | Zoom online Meeting App | P | P | P | Entities Attributes Relationships | Identifying Entities from requirement gathered, relationships and its attributes |
| 9/12/21 2:30 PM | Zoom online Meeting App | P | P | P | Relational Schema EERD | Defining Relational Schema and Designing EERD |
| 9/13/21 2:30 PM | Zoom online Meeting App | P | P | P | Design Documents | Documenting all the things identified and defined in above meetings |
| 9/14/21 2:30 PM | Zoom online Meeting App | P | P | P | Design Mock ups | Designing Mockup of System |
| 9/15/21 2:30 PM | Zoom online Meeting App | P | P | P | Finalizing Documents | Creating and Finalizing Documents |
| 9/16/21 2:30 PM | Zoom online Meeting App | P | P | P | Error Fixing | Fixing any error appear in above design |

## 

## **Purpose & Problem Statement**

New York University wants to create a web-based registration system which can handle real time data. The data is related to its students, faculty either teacher or other staff members. The main problem of the university is dealing with files and paper work and NY University wants to convert all this an automated system in which students can easily register themselves in courses and teacher can update and post all the required things and also results and many more.

## **Homepage**

Homepage is the very first page which appear when a website is searched and home page will contain the very first thing which is *academic calendar* which will contain each and every detail related to holidays, events, start and ending dates of semester and also *master schedule* for next and present semester. From the master schedule user should be able to look for courses names the professors who are going to teach, major and the department. The most important here is that it will all accessible without login and there must be a login section on the top of the site (usually) from when the authenticate students can login and get more options.

## **Functional Requirements**

The requirements below mentioned must be offered by the system as these are also the objectives of the application.

* The application should be able to add, update and monitor every information related to NY university
* The users should be divided into four categories, Administrator, Faculty, Researchers and Students. Each of the group should have certain rights to access only the sections which are defined for them.
* Course Schedule, Master Schedule, login Section and Catalog to the college should be available to home page.
* Each user must have a login and password to access their portal.
* There are certain actions need to be performed again each user and those actions will be defined in latter sections of SRS.

## **Non-Functional Requirements**

These requirements are need to measure the efficiency of the application system. They are of different types.

### **Performance Requirements**

The System should be able to support a specific number of users at one time and the time to load a single page should not exceed than a number of defined seconds. The performance requirements are specially measure the time specified for each reaction performed on a specific action.

### **Reliability**

The system should be reliable enough and each transaction made should be rolled back if the transaction disturbs or interrupt in half of way. The failure ratio should be 1 out of 1000 times. If used leaves without saving the changes the changes shouldn’t be made. The data should set required while entering information. The system should be operational 99% of Time.

### **Availability**

The system should be available 24/7 and if in case of any disaster there should be a backup server to support routine transactions.

### **Security**

The data related to every user shouldn’t be compromised. Only the authentication users will be able to access the portal and only their portal. If there is try to add passwords more than 4 times then the user should be locked out and after locking the user the password needs to be reset with proper verification of email-method.

## **Business Rules**

Below are business rules defined for each user and the rules are of two type one is required and other is forbidden. Required rules are shown and defined which a user is able to manipulate and forbidden rules are not shown to users.

### 

### **General User Business rules**

|  |  |
| --- | --- |
| **Required** | **Forbidden** |
| * Usernames/emails and IDs are unique * Every user has a first name, last name, date of birth, username and a password * Every user will be able to change their password or request a reset * Every user will have access to the master schedule, catalog and academic calendar * Every user has three security questions. * If a user cannot log in, they must contact administration. | * User cannot enter invalid password and is required to do a password reset * User cannot view other users’ information |

### **Admin User Business Rules**

|  |  |
| --- | --- |
| **Required** | **Forbidden** |
| * Admin can create/update the master schedule * Admin can update holds * Admin can cancel classes but he must inform the students/faculty about the changes. * Admin can add/delete users * Admin has access to the user’s list * Admin can view course list * Admin can search for courses and students * Admin can assign professors to classes * Admin can approve the drop of classes * Admin can update change in major/minor * Admin can reset passwords | * When creating the master schedule admin cannot assign two classes to the same room and time * Admin cannot assign two professors to teach two classes at the same time. |

### **Faculty User Business Rules**

|  |  |
| --- | --- |
| **Required** | **Forbidden** |
| * Faculty can assign grades * Faculty can update grades within a time window * Faculty can view all information about their students. * Faculty can view a student’s advisor * Faculty can be both a professor and advisor * Faculty can access their students transcripts7 * Faculty may teach many classes * A faculty only teaches classes within his/her department * Faculty can be either full-time or part-time * Faculty can view their classes * An advisor can advise many students * Faculty can view schedules. * Faculty can view their class list. * Faculty can keep attendance. * Faculty can keep teaching history. | * Faculty cannot update grades outside the time window * Faulty cannot update or assign grades for classes they don’t teach * Faculty cannot alter students’ personal information. * Faculty cannot clear students’ holds. |

### **Student user Business Rules:**

|  |  |
| --- | --- |
| **Required** | **Forbidden** |
| * Students can add/drop classes to their schedule within a time period. * Students can request to overtly a class * A student may be part time or full time * Students can be graduate or undergraduate. * Students can view their grades. * Students can view their unofficial transcript * Students can declare or change major/minor * Students can check major/minor requirements. * Students can view degree audit * Students must meet prerequisites before registering for a course * Students can have one or two majors6 * Students can have zero or two minors * Students can have up two advisors * Students will be able to view their advisor. * Students can view their holds. * Students can have a major and a minor * Students can view their present semester schedule. | * Students cannot change their grades * Students cannot register for classes if they have a hold on their account * Students cannot register for classes without meeting the prerequisites * Students cannot register for multiple classes’ occurring at the same time. * Students cannot clear holds. * Students cannot register for classes if they are overlapping with already * Registered classes. * Students cannot drop classes after a certain date without approvals * Students cannot register for classes that does not have seats available * Students cannot view other student’s transcript or degree audit * Part-time students cannot take more than 12 credits * Fulltime students cannot take more of 18 credits or less than 13 credits. * Students cannot have zero courses * Students cannot create new classes |

### 

### **Researcher User Business Rules**

|  |  |
| --- | --- |
| **Required** | **Forbidden** |
| * Researchers can see all the data with no identification. * Researchers can view details about classes such as size, time offered, etc. * Researchers can prepare college stats. | * Researchers cannot access individual’s personal data. * Researchers cannot update any user’s information. |

## **Entities & Attributes of Entities**

Below is the table which given the information related to entities and their attributes, the primary key of each entity will be in bold and the foreign key will be in italic format.

|  |  |
| --- | --- |
| **Entity** | **Attribute** |
| USER | **User\_ID,** F\_Name, L\_Name, DOB, Email, Phone, Date\_Created, Username, Password, User\_Type |
| STUDENT | ***Student\_ID*,** Student\_GPA, Student\_Type, Credits\_Earned |
| UNDERGRADUATE | ***Student\_ID,*** Type, Standing |
| GRADUATE | ***Student\_ID,*** Type, Graduate\_Progam |
| STUDENT\_ FULLTIME | ***Student\_ID,*** Type |
| STUDENT\_ PARTTIME | ***Student\_ID,*** Type |
| ENROLLMENT | ***Student\_ID, Class\_ID,*** *Semester\_ID*, Grade |
| ATTENDANCE | ***Student\_ID, Class\_ID*,** Date\_Att, Status |
| HOLD | **Hold\_ID,** Hold\_Type |
| STUDENT\_HOLDS | ***Student\_ID, Hold\_ID*** |
| FACULTY | ***Faculty\_ID*,** *Room\_ID*, Speciality, Rank, Type |
| FULLTIME\_FACULTY | ***Faculty\_ID,*** Type |
| PARTTIME\_FACULTY | ***Faculty\_ID,*** Type |
| ADVISOR\_STUDENT | ***Student\_ID, Faculty\_ID,*** Assign\_Date |
| ADMINISTRATOR | ***Admin\_ID*,** Privilege\_Level |
| RESEARCHER | ***Researcher\_ID*** |
| BUILDING | **Building\_ID,** Name |
| ROOM | **Room\_ID,** *Building\_ID*, Room\_Type, Room\_Number |
| LAB | ***Room\_ID*,** Capacity |
| LECTURE\_HALL | ***Room\_ID*,** Capacity |
| OFFICE | ***Room\_ID*,** Capacity |
| DEPT\_FACULTY | **Dept\_ID, Faculity\_ID,** Join\_Date |
| DEPARTMENT | **Dept\_ID,** Name**,** Phone, *Manager\_ID* |
| COURSE | **Course\_ID,** Title, Credit\_Hours, PassGrade, *Dept\_ID, Prereq\_ID* |
| CLASS | **Class\_ID,** *Course\_ID***,** *Faculty\_ID*, *Room\_ID, Semester\_ID, Timeslot\_ID* |
| TIME-SLOT | **Timeslot\_ID,** Date, Day, Period |
| TIME\_SLOT\_PERIOD | ***Timeslot\_ID, Period\_ID*** |
| PERIOD | **PeriodID,** Start\_Time, End\_Time |
| SEMESTER | **Semester\_ID,** Date\_Start, Date\_End |
| MAJOR | **Major\_ID,** Name, *Dept\_ID* |
| MINOR | **Minor\_ID,** Namr, *Dept\_ID* |
| STUDENT\_MAJOR | ***Student\_ID, Major\_ID*** |
| STUDENT\_MINOR | **Student\_ID, Minor\_ID** |
| MINOR\_REQUIREMENTS | ***CourseID, Major\_ID*** |
| MAJOR\_REQUIREMENTS | ***CourseID, Minor\_ID*** |
| STUDENT\_HISTORY | ***Student\_ID, Class\_ID, Semester\_ID*,** Grade |
| FACULTY\_HISTORY | ***Fculty\_ID, Class\_ID, Semester\_ID*** |

### **Relationships:**

* One Student can be taught by many professors and one Professor can teach many Students. So, the relation between professor faculty and the student is many to many and this needs to be break down to one to many. To Manage this the relation **Advisor** is created which contains both student and faculty assigned to each other.
* One Student can take many Classes and one Class can be taken by many Students. There is again many to many relationships but between students and classes to break down to one to many as the relation created **Enrollment** here.
* A Faculty professor may or may not be an Advisor and an Advisor is always a professor. Professor and Advisor have a one or zero relationship.
* Professors can teach to many Classes and one Class is taught by one and only one Professor. Professor and Class have a one to many relationship.
* Students can have one or many majors and one major can be taken by many students. Students and Major have a many to many relationships. Since the relation of majors and students is many to many it must be broken down to one to many. **Student\_major** is Created as new relation
* Students can have one or many minors and one minor can be taken by one or many students. Students and Minor have a many to many relationship. Since the relation of minors and students is many to many it must be broken down to one to many. **Student\_minor** is created as new relation
* One Student can have zero or many holds and a hold is held by zero or many Students. Students and holds have a many to many relationship. Since the relation of holds and students is many to many it must be broken down to one to many. Student\_holds is created as new relation
* Faculty Member may or may not be Manager of a department and a department has one Manager. Professors and departments have a zero to one relationship.
* Departments can have one or many majors and a major belongs to one and only one department. Departments and majors have a many to one relationship.
* Departments can have many minors and a minor belongs to one department. Departments and minors have a many to one relationship.
* Departments offer many courses but a course belongs to only one department. Department and Course have a many to one relation
* A class takes place in one room and one room can host many classes. The lation is one to many
* A building has many rooms but a room belongs to only one building. Room and Building have a one-to-many relation
* A course generates many classes and a class belongs to only one course. Course and Classes have a one-to-many relation
* A course can have only one pre requisite.
* Semester has many classes and a class belongs to many semesters. The relation between Semester and classes is many to many and break down in **Enrollment**.
* Time\_Slot\_Period and Class share a one-to-many relationship.

### **Relational Schema:**

USER (**User\_ID**, F\_Name, L\_Name, DOB, Email, Phone, Date\_Created, Username, Password, User\_Type)

STUDENT (***Student\_ID***, Student\_GPA, Student\_Type, Credits\_Earned)

UNDERGRADUATE (***Student\_ID***, Type, Standing)

GRADUATE (***Student\_ID***, Type, Graduate\_Progam)

STUDENT\_ FULLTIME (***Student\_ID***, Type)

STUDENT\_ PARTTIME (***Student\_ID***, Type)

ENROLLMENT (***Student\_ID***, ***Class\_ID***, ***Semester\_ID***, Grade)

ATTENDANCE (***Student\_ID***, ***Class\_ID***, Date\_Att, Status)

HOLD (**Hold\_ID**, Hold\_Type)

STUDENT\_HOLDS (***Student\_ID***, ***Hold\_ID***)

FACULTY (***Faculty\_ID, Room\_ID***, Speciality, Rank, Type)

FULLTIME\_FACULTY (***Faculty\_ID***, Type)

PARTTIME\_FACULTY (***Faculty\_ID***, Type)

ADVISOR\_STUDENT (***Student\_ID***, ***Faculty\_ID***, Assign\_Date)

ADMINISTRATOR (**Admin\_ID**, Privilege\_Level )

RESEARCHER (***Researcher\_ID***)

BUILDING (***Building\_ID***, Name)

ROOM (***Room\_ID,*** Building\_ID, Room\_Type, Room\_Number)

LAB (***Room\_ID***, Capacity)

LECTURE\_HALL (***Room\_ID,*** Capacity)

OFFICE (***Room\_ID***, Capacity)

DEPT\_FACULTY (**Dept\_ID**, Faculity\_ID, Join\_Date, Percentage\_Time)

DEPARTMENT (**Dept\_ID,** Name, Phone, Manager\_ID)

COURSE (**Course\_ID**, Title, Credit\_Hours, PassGrade, Dept\_ID, Prereq\_ID)

CLASS (**Class\_ID**, Course\_ID, Faculty\_ID, Room\_ID, Semester\_ID, Timeslot\_ID)

TIME-SLOT (**Timeslot\_ID**, Date, Day, Period)

TIME\_SLOT\_PERIOD (Timeslot\_ID, Period\_ID)

TIME\_SLOT\_Day (Timeslot\_ID, Day)

Day(DayOfWeek)

PERIOD (**Period\_ID**, Start\_Time, End\_Time)

SEMESTER (**Semester\_ID**, Date\_Start, Date\_End)

MAJOR (**Major\_ID**, Name, Dept\_ID)

MINOR (**Minor\_ID**, Namr, Dept\_ID)

STUDENT\_MAJOR (***Student\_ID***, ***Major\_ID***)

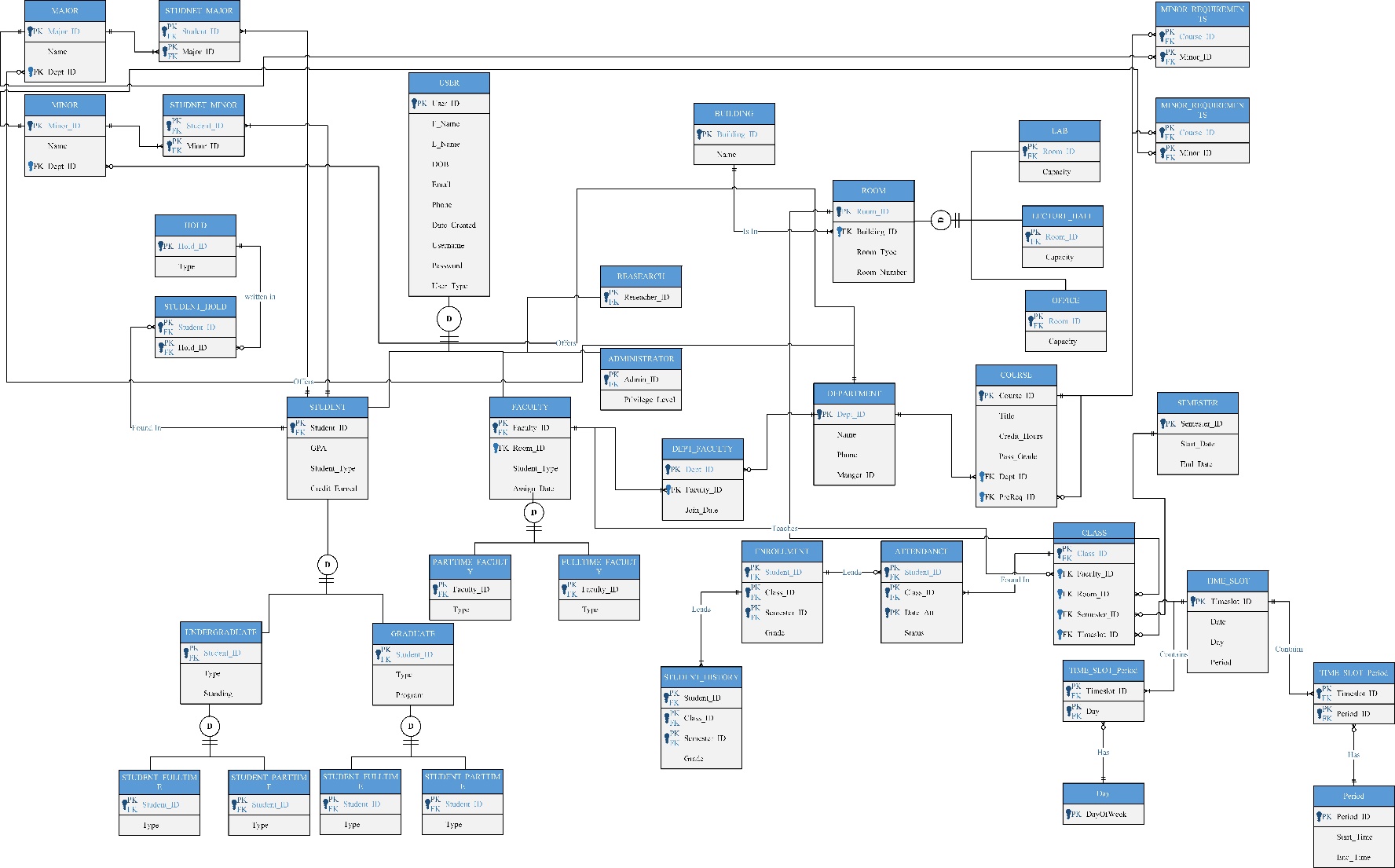
STUDENT\_MINOR (***Student\_ID***, ***Minor\_ID***)

MINOR\_REQUIREMENTS (***CourseID, Major\_ID***)

MAJOR\_REQUIREMENTS (***CourseID, Minor\_ID***)

STUDENT\_HISTORY (***Student\_ID***, Class\_ID, Semester\_ID, Grade)

### **Entity Relationship Diagram (EERD)**



## 

## **Use Cases:**

### **General User Use Cases:**

|  |  |
| --- | --- |
| **Use Case ID** | UC-1 |
| **Use Case Name** | Master Schedule Access |
| **Primary Actor** | Generic User |
| **Pre-Conditions** | Open the Home Page of site and may or may not be logged in and wants to check the semester he needs. |
| **Post-Conditions** | User successfully access the master schedule |
| **Main Success Scenario** | 1. User Browse the Home Page 2. User Selects the Master Schedule 3. User Selects the semester he needs to check |
| **Frequency of Use** | Several times a day |

|  |  |
| --- | --- |
| **Use Case ID** | UC-2 |
| **Use Case Name** | Catalog Access |
| **Primary Actor** | Generic User |
| **Pre-Conditions** | Open the Home Page of site and may or may not be logged in and wants to check the catalog he needs. |
| **Post-Conditions** | User successfully access the catalog |
| **Main Success Scenario** | 1. User Browse the Home Page 2. User Selects the Catalog of college 3. Browse further to the catalog selected. |
| **Frequency of Use** | Several times a day |

|  |  |
| --- | --- |
| **Use Case ID** | UC-3 |
| **Use Case Name** | Request Password Reset |
| **Primary Actor** | Generic User |
| **Pre-Conditions** | User forget/ lost or wants to change his password/email and wants to reset or recover. |
| **Post-Conditions** | User successfully reset |
| **Main Success Scenario** | 1. User Browse the Home Page 2. Select Login 3. Select forgot password 4. Request Reset Link. |
| **Frequency of Use** | When user needs |

|  |  |
| --- | --- |
| **Use Case ID** | UC-4 |
| **Use Case Name** | Access Academic Colander |
| **Primary Actor** | Generic User |
| **Pre-Conditions** | Open the Home Page of site and may or may not be logged in and wants to check the academic calendar |
| **Post-Conditions** | User successfully access the academic calendar |
| **Main Success Scenario** | 1. User Browse the Home Page 2. User Selects the academic calendar |
| **Frequency of Use** | Several times a day |

### **Student User Use Cases:**

|  |  |
| --- | --- |
| **Use Case ID** | UC-5 |
| **Use Case Name** | View current semester schedule |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in, into his portal |
| **Post-Conditions** | Student successfully views the semester schedule |
| **Main Success Scenario** | 1. Student Logs into the system 2. On the home page student can select and view the current semester schedule. |
| **Frequency of Use** | Several times a day |

|  |  |
| --- | --- |
| **Use Case ID** | UC-6 |
| **Use Case Name** | View Profile |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in, into his portal |
| **Post-Conditions** | Student successfully views His/her Profile |
| **Main Success Scenario** | 1. Student Select his Profile 2. System Opens his/her detail information |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-7 |
| **Use Case Name** | Add Class |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in, into his portal and know his class id in which he wants to add himself.  Student must be within time frame, must have no holds, must meet pre-req and cannot reach the defined classes limit. |
| **Post-Conditions** | Student successfully Adds the class |
| **Main Success Scenario** | 1. Student Selects Add/Drop Classes 2. Student Search for Class ID 3. Click “Add” button to enroll in the class |
| **Frequency of Use** | On Semester start |

|  |  |
| --- | --- |
| **Use Case ID** | UC-8 |
| **Use Case Name** | Drop Class |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in, into his portal and must be within time frame. |
| **Post-Conditions** | Student successfully Drops the class |
| **Main Success Scenario** | 1. Student Selects Add/Drop Classes 2. Selects from the added classes 3. Submit the request for Drop the Class |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-9 |
| **Use Case Name** | View Advisor |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in. |
| **Post-Conditions** | Student successfully views advisor |
| **Main Success Scenario** | 1. Student Selects Degree Information 2. System Shows all the information including Advisor |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-10 |
| **Use Case Name** | View Grades |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in. |
| **Post-Conditions** | Student successfully views Grades |
| **Main Success Scenario** | 1. Student Selects the “Grades” 2. Student Select the needed semester and the grades will appear. |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-11 |
| **Use Case Name** | View Holds |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in. |
| **Post-Conditions** | Student successfully views Holds. |
| **Main Success Scenario** | 1. Student Selects the “Holds” option. |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-11 |
| **Use Case Name** | View Transcripts |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in. |
| **Post-Conditions** | Student successfully views Transcripts. |
| **Main Success Scenario** | 1. Student Selects the “Transcripts” option. |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-12 |
| **Use Case Name** | Declaring Major |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in. |
| **Post-Conditions** | Student successfully Declare his major form. |
| **Main Success Scenario** | 1. Student Selects Form Section 2. Student Select the Major Declaration Form. 3. Fills and Submit form. |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-13 |
| **Use Case Name** | Change Major or Minor |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in. |
| **Post-Conditions** | Student successfully Applied for change his major or minor. |
| **Main Success Scenario** | 1. Student Selects Form Section 2. Student Select the Major or Minor Change Form. 3. Fills and Submit form. |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-14 |
| **Use Case Name** | View Major Minor Requirements |
| **Primary Actor** | Student |
| **Pre-Conditions** | Student Must be logged in. |
| **Post-Conditions** | Student successfully views the requirements information. |
| **Main Success Scenario** | 1. Student Selects “Degree Information” Option 2. Student will be able to view the major/minor requirements information |
| **Frequency of Use** | On Need Basis |

### **Researcher:**

|  |  |
| --- | --- |
| **Use Case ID** | UC-15 |
| **Use Case Name** | Access Data |
| **Primary Actor** | Researcher |
| **Pre-Conditions** | Researcher Must be logged in. |
| **Post-Conditions** | Researcher successfully able to access the data. |
| **Main Success Scenario** | 1. Researcher Selects the desired option in menu provided for research. |
| **Frequency of Use** | On Need Basis |

|  |  |
| --- | --- |
| **Use Case ID** | UC-16 |
| **Use Case Name** | View Stats |
| **Primary Actor** | Researcher |
| **Pre-Conditions** | Researcher Must be logged in. |
| **Post-Conditions** | Researcher successfully able to access the stats. |
| **Main Success Scenario** | 1. Researcher Selects “Get Stats” 2. Selects the type of data 3. Click the “Get results” button |
| **Frequency of Use** | On Need Basis |

### **Faculty Professor**

|  |  |
| --- | --- |
| **Use Case ID** | UC-17 |
| **Use Case Name** | Access Master Schedule |
| **Primary Actor** | Professor |
| **Pre-Conditions** | Professor Must be logged in. |
| **Post-Conditions** | Professor successfully able to access the Master Schedule. |
| **Main Success Scenario** | 1. Professor logins to system 2. Select the master schedule |
| **Frequency of Use** | Several times a day. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-18 |
| **Use Case Name** | Access Student Transcript |
| **Primary Actor** | Professor |
| **Pre-Conditions** | Professor Must be logged in. |
| **Post-Conditions** | Professor successfully able to access the desired student transcript. |
| **Main Success Scenario** | 1. Professor Selects the class he/she teaches 2. Select the student from the student list. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-19 |
| **Use Case Name** | Upload Grades |
| **Primary Actor** | Professor |
| **Pre-Conditions** | Professor Must be logged in. |
| **Post-Conditions** | Professor successfully able to upload the grades. |
| **Main Success Scenario** | 1. Professor Selects the class he/she teaches 2. Select the upload grades 3. Select the student from the list 4. Input the grades for the student 5. Submit the grades. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-20 |
| **Use Case Name** | View Student List |
| **Primary Actor** | Professor |
| **Pre-Conditions** | Professor Must be logged in. |
| **Post-Conditions** | Professor successfully able to access student list |
| **Main Success Scenario** | 1. Professor Selects the class he/she teaches 2. Select the student list. |
| **Frequency of Use** | On Need Basis. |

### **Administrator**

|  |  |
| --- | --- |
| **Use Case ID** | UC-21 |
| **Use Case Name** | Clear/Apply Holds |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully able to Apply/Clear holds on a student. |
| **Main Success Scenario** | 1. Admin search for the student via student id. 2. Admin Select the student from search and apply or remove holds from student’s account. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-22 |
| **Use Case Name** | Create/Update Master Schedule |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully able to Create/Update Master Schedule. |
| **Main Success Scenario** | 1. Admin Select “Create/Update” Master Schedule. 2. Admin add/update the course information to schedule. 3. Save changes. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-19.50 |
| **Use Case Name** | Change Grades |
| **Primary Actor** | Professor |
| **Pre-Conditions** | Professor Must be logged in. |
| **Post-Conditions** | Professor Cannot Change Grades after certain time |
| **Main Success Scenario** | 1. Professor Selects the class he/she teaches 2. Select the upload grades 3. Select the student from the list 4. Input the grades for the student 5. Submit the grades. 6. If the time to change grade is expired than professor get message cannot change grade. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-23 |
| **Use Case Name** | Create/Update Academic calendar |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully able to create/update academic calendar. |
| **Main Success Scenario** | 1. Admin Select “New Term” 2. Admin Selects “Academic Calendar” 3. Inputs data to the list 4. Submit changes. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-24 |
| **Use Case Name** | Reset User Password |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully resets the password of the user |
| **Main Success Scenario** | 1. Admin Access user list. 2. Search for user 3. Select the user and reset password |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-25 |
| **Use Case Name** | Add User |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully add the user |
| **Main Success Scenario** | 1. Admin Selects “Add User” to add new user to the system 2. Admin Enter the information of user and define its role. 3. Click create button and save the changes. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-26 |
| **Use Case Name** | Delete User |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully delete the user |
| **Main Success Scenario** | 1. Admin selects all user 2. Search the desire user 3. Hit the “delete user” button 4. Save the changes. |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-27 |
| **Use Case Name** | Add Course Catalog |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully add the course to catalog. |
| **Main Success Scenario** | 1. Admin selects catalog 2. Select “Add Course” |
| **Frequency of Use** | On Need Basis. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-28 |
| **Use Case Name** | Remove Course Catalog |
| **Primary Actor** | Admin |
| **Pre-Conditions** | Admin Must be logged in. |
| **Post-Conditions** | Admin successfully remove the course to catalog. |
| **Main Success Scenario** | 1. Admin selects catalog 2. Select the course from list 3. Hit remove button 4. Save changes. |
| **Frequency of Use** | On Need Basis. |

## **Technologies to be use**

As there are many technologies can be used to develop the system based on web app like ASP.NET, Python, PHP and also there are different technologies can be combined to create a web app. Here we are going to develop NY web-based Registration system into php scripting language. Further information is given below.

**Front-End Technology**: For every web the front needs to be in HTML, CSS and JS with Jquery.

**Back-End Technology:** our functional language will be in PHP and it have many frameworks also including core.

**Database:** We are going to use the database MySQL.

## **Feasibility:**

### **Time Constraints**

For each part of the implementation, we will have time limits where each task is

completed

The database implementation and population will take no more than 3

days as we already would have the data.

The back-end programming as the most complex task will take at maximum 3 weeks

Implementation with the front end should take us no more than a week

For the testing part we will spend another week testing and fixing issues

### **Task Breakdown:**

Our tasks will break down in the following

* Implement the landing page with a personal domain:

Implement the existing landing page with a personalized domain

Make sure the set up works for any kind of browser

* Push the data and the database

Push the user data

Test if the data works

* Back end and front-end programming:

Php back-end programming for managing the data

Front end data display data to each user

* Testing

Test the system in different browsers

Test failure cases

Test in different Operating Systems

* Fixing issues

Fix any issue encountered on the back end

Move to fixing the front end