Course work submitted

by

Justice Aluya

SID: 1017166

for CMM503 – WEB SYSTEM DEVELOPMENT

at The Robert Gordon University Aberdeen

# Description

This is a web project, proposed to facilitate the assignment of student to project groups, taking into account minimum and maximum group sizes, as well as degree cohort.

The system will automatically assign students into groups with criteria;

Administrators can train the system to make better choice of peers.

Student will log-in to system to work with their group, they can request group swap with another student upon agreement and the authorization of an administrator.

Peers will be able to upload research materials as well as other files to group billboard or to individual peers.

The system will be developed in PHP providing both front-end interface for student and back-end interface for Administrators; There will be server side codes that will handle interaction with database, web and file servers to provide dynamic content to system users.

The entire system shall be hosted on the Microsoft Azure cloud platform.

# Constraints

The system shall be designed to conform to PHP standards, and compatible with the Microsoft Azure platform. This will require me to design and implement database to conform with SQL Server requirements.

A login system will be developed for both administrators and students to log-in to the system, students will be able to view areas to which they have been granted access by virtue of their session information.

Student will be able to view all groups but no access to individual group projects and members activities except for the group to which they belong.

Students will be able to upload and download files to/from their group and peers.

Administrators will be able to download and upload files to all groups as well as be able to delete files on a censorship duty.

Table of Contents

[Description 2](#__RefHeading___Toc503_1103359429)

[Constraints 2](#__RefHeading___Toc505_1103359429)

[Functional Requirements 4](#__RefHeading___Toc507_1103359429)

[Login with Default Administrator Account 4](#__RefHeading___Toc509_1103359429)

[Creating Administrator Accounts 4](#__RefHeading___Toc511_1103359429)

[Creating Projects 4](#__RefHeading___Toc513_1103359429)

[Creating Student Accounts 4](#__RefHeading___Toc515_1103359429)

[Students Initial Sign-on 5](#__RefHeading___Toc517_1103359429)

[Students sign-in 5](#__RefHeading___Toc519_1103359429)

[creating project groups 5](#__RefHeading___Toc521_1103359429)

[Automatic Group assignment 5](#__RefHeading___Toc523_1103359429)

[Non-Functional Requirements 5](#__RefHeading___Toc525_1103359429)

[Interface (HCI) 5](#__RefHeading___Toc527_1103359429)

[Extrnded HCI 6](#__RefHeading___Toc529_1103359429)

# Functional Requirements

### Login with Default Administrator Account

The system must provide mechanism to:

1. Accept user name: Administrator
2. Accept Password : Admin
3. request user to change password to a secure password (8 or more characters long with a combination of Letters and Numbers)
4. provide functionality to Create other administrator accounts
5. provide functionality to create projects
6. provide functionality to create project groups
7. provide functionality to create student accounts
8. provide functionality to assign students to groups Automatically
9. provide functionality to assign swap students group

### Creating Administrator Accounts

The system could provide mechanism to

1. Accept user name
2. Accept password (Strong)
3. accept Name
4. Accept Surname
5. Accept email address
6. accept mobile number

### Creating Projects

The system must provide mechanism to:

1. accept project name
2. accept project description
3. accept project type

### Creating Student Accounts

The system must provide mechanism to:

1. A. provide functionality to upload class list containing details in 1 – 6 below
2. Accept Student Name
3. Accept student Surname
4. Accept student programme Major
5. Accept student programme Minor
6. Accept DOB
7. Assign student ID

### Students Initial Sign-on

The system must provide mechanism to:

1. Accept student’s ID
2. Accept student’s DOB
3. Accept students New password
4. Confirm Student’s new password

### Students sign-in

The system must provide mechanism to:

1. Accept student’s ID
2. accept student password
3. provide students with groups list
4. provide students with advanced options
5. allow students to request group swap
6. provide student with group bill board
7. allow students to leave group comments
8. allow students to leave comments for peers
9. provide functionality to upload files with comments (upload possible without comments)
10. provide functionality to read and download comment files and uploads

### creating project groups

The system must provide mechanism to:

1. accept group name
2. accept group min size
3. accept group max size

### Automatic Group assignment

The system must provide mechanism to:

1. count students
2. dive count by the highest of minimum group size
3. if studentsCount modulo Max(minGroupSize) exist, then divide the outcome by group count
4. assign |studentsCount modulo Max(minGroupSize)/groupCount| + studentsCount/Max(minGroupSize)

# Non-Functional Requirements

### Interface (HCI)

The system could:

1. Fully/partially adopt the Nielsen’s Heuristics
2. provide adaptive output (re-arranging or minimizing) on the basis of end device maximum output line width and minimum output line numbers
3. Audible error notification
4. Adopt the MVC Framework
5. Multi-Factor Authentication

### Extended HCI

The system will not provide:

1. Bar code authentication/authorization
2. smart card authentication/authorization
3. additional cloud storage