Phase 2

Prepared for: Maher Hakim, Professor
Prepared by: Simpletech

February 21, 2013

Table of Contents

Executive Summary 1

Use Cases 2

- Diagrams 2
- A-Level 3
- B-level 10
- C-level 15

Test Cases 18

- A-Level 19
- B-Level 22
- C-Level 25

Non-functional requirements 26

User Experience and Site Layout 30

- Prototypes 30
- Site map 43
- Design Issues 44

Technical Architecture & Design 46

- Description 46
- ERD 47
- Data Model 48
- Data dictionary 49

Project Plan 51

- Phase 3 52
- Phase 4 54
- Phase 5 56

Project Progress Report 58

- Time Accounting 58
- Feasibility Update 59
- QA manager's Report 60
- Project Metrics 61
- Risk Management 62
- Problem Status 64

Executive Summary

Our team, Simpletech, is developing an application for to change the cross registration process in Education City. The application would allow students to find courses and apply for them online. It would also allow university registrars to view these applications and approve or decline them. This application will make the cross registration process faster and more efficient, as the current process takes a lot of time in paper work and physically going to the campuses to apply for courses.

During this phase, our team met regularly to discuss the use cases of the application and how to test them. During these meetings, we came up with the use case and classified them according to importance to the system. We decided on the most important non functional requirements of the application according to our stakeholders and users of the application. The team moved forward to design the site layout, as we developed prototypes for how the site would look like and a site map to show the organization of the site. We had to make sure that the site does not have too many levels, to minimize the number of clicks a user has to make.

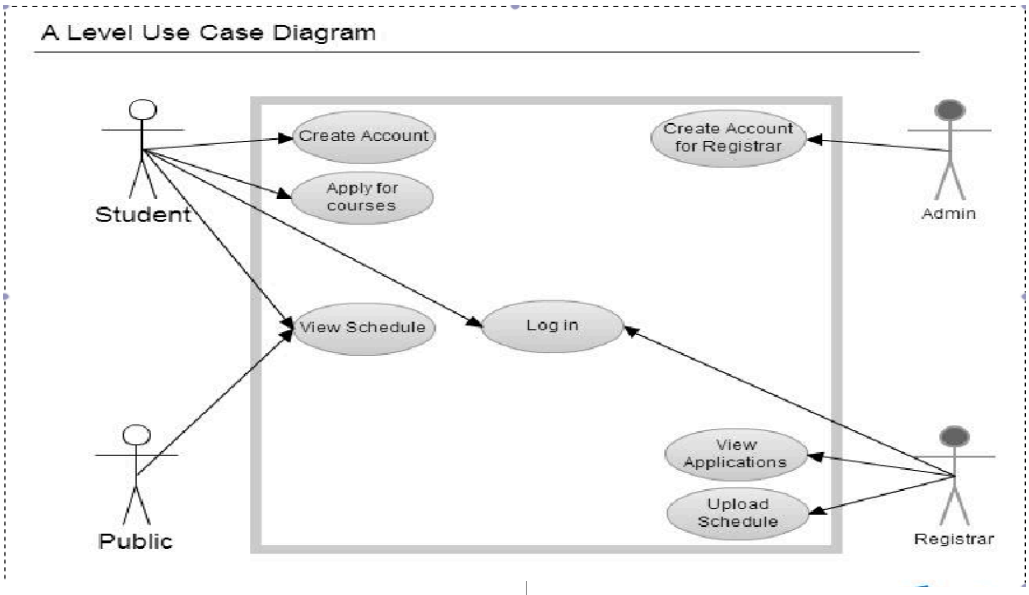
Simpletech progressed to designing the technical architecture and design of the application, which consisted of a description of the tools and technologies that are going to be used to develop the application, an Entity Relationship diagram, a data model and a data dictionary. The technical architecture will help to guide the team during the developing phases. We will be using Ruby on Rails to develop the application that has the following entities: University, Student, Application, Registrar and Schedule.

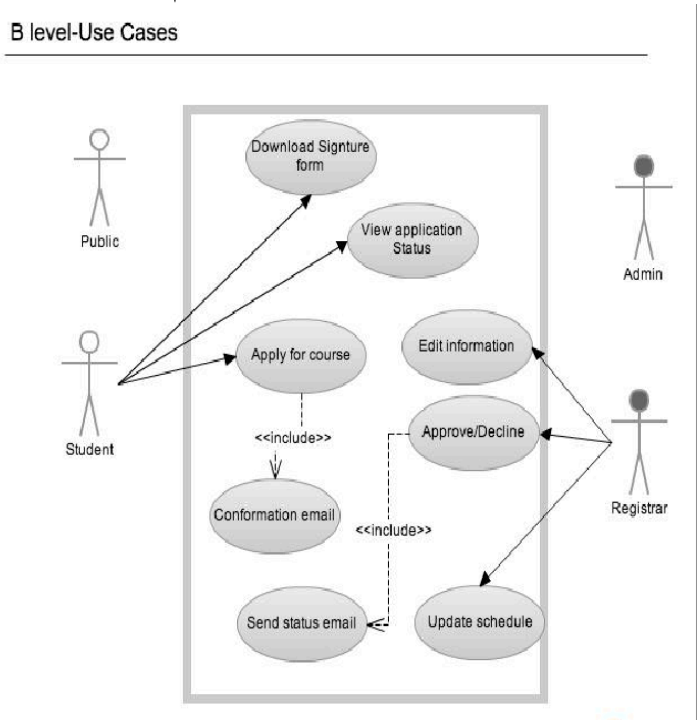
The next step was to figure out what are we going to do next. The answer came through with the project plan for the next three phases of implementation, which we divided according to the use cases of the application. The project plan includes the division of tasks, deadlines and the estimated number of hours each member will spend in development.

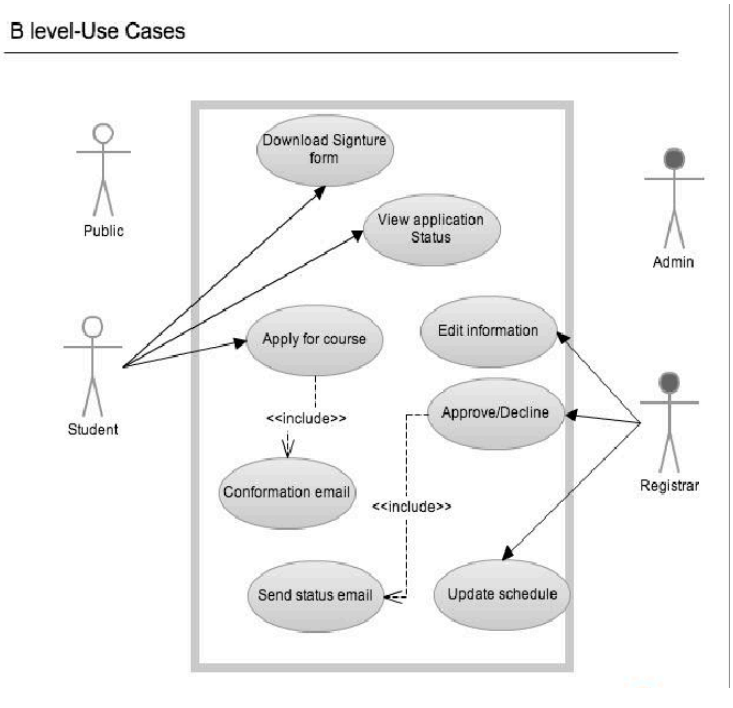
Lastly, the team had to update the previous documentation we had. The update included the project feasibility, project metrics, risk analysis and problem analysis. These updates provide a feel of where the team stands in the project and will further impact our work. Our Quality Assurance manager provided a report about the quality of the work and how is it progressing, in order to ensure better quality of our future work.

Use Cases

Diagrams







Actor List		
Actor Name	Level	Description
Admin	Average	This actor is the developer, designer and system administrator. This actor will update the system, make sure everything is working smoothly and supervise the system.
Registrar	Complex	This actor is a registrar, will have access to the site, will interact with the system and check applications of Education City students that would like to register for a certain course.
Student	Complex	This actor is the student who will use the website to look for a course or register for it
Public	Complex	This actor can be a student, a professor or anyone who just want to check the website or schedules with no intention to register for anything.

A-Level

Name	Create an account
Description	This use case allows students to create an account.
Actors	Student
Preconditions	Student is connected to the internet. Student is on the “Home” page.
Basic flow of events	1. The actor clicks “Sign up” button 2. The system redirects the actor to “Sign up” page 3. The actor fills up the details on “Sign up” form (first name*, last name*, email*, password*, university*, student id*, major*, expected graduation year*, phone number*) and presses “Sign up” 4. The system then validates the fields, first in terms of availability then checks for minimum/maximum length or the type of input or the format of input(using regex) 5. Actor is redirected to “Log in” page and a success message is shown.

Triggers	The actor clicks “Sign up” button.
Alternative sequence	Student clicks the cancel button and is redirected to the “home” page
Error sequence	Invalid input by the actor wont allow the creation.
Postconditions	The actor is successfully registered in the system. Redirection to the Student “Home” page.

Name	Create account for Registrar
Description	This use case allows admins to create an account for university registrars.
Actors	Admin.
Preconditions	Admin is connected to the internet. Admin is on the “Home” page.
Basic flow of events	1. The actor clicks on the “Registrar button” 2. System redirects actor to the registrar page. 3. The actor clicks on “Create” button. 4. The actor is redirected to the “Create page” 5. The actor fills up the details on “Sign up” form (first name*, last name*, email*, password*, university*, phone number*) and presses “Sign up” 6. The system then validates the fields, first in terms of availability then checks for minimum/ maximum length or the type of input or the format of input(using regex) 7. Actor is redirected to “Log in” page and a success message is shown.

Triggers	The actor clicks “Create” button on the Registrar page.
Alternative sequence	Student clicks the cancel button and is redirected to the “home” page.
Error sequence	Invalid input by the actor wont allow the creation.
Postconditions	Registrar is successfully registered.

Name	Log in
Description	This use case allows actors to log in to the application
Actors	Registrar, Student
Preconditions	Actor is connected to the internet. Actor is on the “Home” page.
Basic flow of events	1. The actor clicks “Sign in” button. 2. The actor chooses the type of account. 3. The actor fills up the details on “Sign in” page (email*, password*) and presses “Sign in” 4. The system then does a backend process and authenticates the actor’s information provided and stored in the system. 5. If everything is correct the backend sends an ok signal to the system to log in the Actor 6. Actor is redirected to “Home” page and a success message is shown.
Triggers	The actor clicks on the Sign in button.
Alternative sequence	Actor clicks cancel / another link on the page.

Error sequence	Invalid email and password will redirect the actor to the sign in page.
Postconditions	Actor is successfully signed in and have access to different features.

Name	Upload schedule
Description	This use case allows university registrars to upload a schedule for a university.
Actors	Registrar
Preconditions	Registrar is connected to the internet. Registrar logged in.
Basic flow of events	1. The actor clicks “Schedule” button on the “Home” page. 2. The system redirects the actor to “Schedule” page. 3. The actor click on “Add” button. 4. The actor fills up the details on the schedule “Add” page (university name*, semester*). 5. The actor clicks on “choose file” and selects a file from hard drive. 6. The actor clicks on “Save” button. 7. The schedule will be saved. 8. The actor will be redirected to “Schedule page” and a success message will be shown.
Triggers	The actor clicks on the schedule button.
Alternative sequence	Registrar clicks cancel and redirected to the schedule page.
Error sequence	Invalid file.

Postconditions	New schedule uploaded. Database is updated with the new schedule.

Name	View schedule
Description	This use case allows Students to view course schedules for all universities
Actors	Students
Preconditions	Student is connected to the internet. Student should be on the “Home” page
Basic flow of events	1. The actor clicks “Schedule” button. 2. The system redirects the actor to “Schedule” page. 3. The actor clicks on the link of a specific university of whom he/she wants to view the schedule of. 4. The system then redirects to the schedule page of that university. 5. The actor can now download a PDF format file that has the schedule of that university.
Triggers	The actor clicks on the schedule button.
Alternative sequence	Student clicks cancel and redirected to the schedule page.
Error sequence	Invalid file. File corrupted
Postconditions	New schedule uploaded. Database is updated with the new schedule.

Name	Apply for courses.
Description	This use case allows students to apply for courses.
Actors	Student

Preconditions	Student is connected to the internet. Student logged in.
Basic flow of events	1. The actor clicks “Applications” button on the “Home” page. 2. The system redirects the actor to “Applications” page. 3. The actor click on “Apply” button on “Application” page. 4. The system redirects to “Application Form” page. 5. The actor fills up the details on the “Apply” form(course number*, host university*) 6. The actor clicks on “choose file” to upload passport copy and selects a file from hard drive. 7. The actor clicks on “choose file” to upload registration form and selects a file from hard drive. 8. The actor clicks on “Submit” button. 9. The application will be saved on to the system. 10. The actor will be redirected to “Apply” page and a success message will be shown.
Triggers	The actor clicks on the “Apply” button
Alternative sequence	Actor clicks cancel and redirected to the application page.
Error sequence	Actor inputs invalid course information. Actor chooses Invalid file.
Postconditions	The application is sent to the registrar of the specific university.

Name	View applications.
Description	This use case allows Registrars to view the applications from students.

Actors	Registrar.
Preconditions	Registrar is connected to the internet. Registrar is logged in.
Basic flow of events	1. The actor clicks “Applications” button on the “Home” page 2. The system redirects the actor to “Application” page. 3. The actor can now see applications from his/her specific university. 4. The actor can choose an application and click on it. 5. The actor will now be redirect to that application view page.
Triggers	The actor clicks on the “Applications” button
Alternative sequence	Actor clicks cancel and redirected to the applications page.
Error sequence	Actor can’t view the applications.
Postconditions	Registrar can view the applications.

B-level

Name	Approve/decline applications
Description	This use case allows Registrars to approve or decline applications sent by students.
Actors	Registrar.
Preconditions	Registrar is connected to the internet. Registrar is logged in.
Basic flow of events	1. The actor clicks “Applications” button on the “Home” page. 2. The system redirects the actor to “Application” page. 3. The actor can now see applications from his/her specific university. 4. The actor can choose an application and click on it. 5. The actor will now be redirect to that application view page. 6. The actor clicks on edit button. 7. The actor will be redirected to “Edit” page of that application. 8. The actor can choose a status “Approve, Decline or Pending” from a drop down box.. 9. Click on save. 10.The actor will be redirected to that application view page and a success message will be shown.
Triggers	The actor clicks on the “Applications” button
Alternative sequence	Actor clicks cancel and redirected to the applications page.
Error sequence	Status of application is not changed.
Postconditions	Application status is changed.

Name	View application status.
Description	This use case allows students to view the status of their applications.
Actors	Student

Preconditions	Student is connected to the internet. Student is logged in.
Basic flow of events	1. The actor clicks “Applications” button on the “Home” page. 2. The system redirects the actor to “Application” page. 3. The actor can now see applications that he/she created. 4. The actor can choose an application and click on it. 5. The actor will now be redirect to that application view page. 6. The actor will be able to view the status.
Triggers	The actor clicks on the “Applications” button
Alternative sequence	Actor clicks cancel and redirected to the applications page.
Error sequence	Actor can’t view the applications or statuses.
Postconditions	Actor can see application list with their statuses.

Name	Edit information.
Description	This use case allows actors change their information on the system.
Actors	Student, Registrar
Preconditions	Actor is connected to the internet. Actor is logged in.

Basic flow of events	1. The actor clicks on his/her name on "Home". 2. The actor is redirect to his/her profile view page. 3. The actor clicks on "Edit" button. 4. The actor will now be redirect to the "Edit" page. 5. The actor can now change anything he/she wants. 6. After the actor done, he/she clicks "Save" button. 7. Actor is redirected to profile view page.
Triggers	The actor clicks on his/her name on the "Home" page.
Alternative sequence	Actor clicks cancel and redirected to the profile page.
Error sequence	Information not saved.
Postconditions	Information changed and saved to the system.

Name	Conformation email
Description	This use case allows students to receive emails to conform the application is submitted.
Actors	Student
Preconditions	Actor is connected to the internet. Actor is registered in the system.
Basic flow of events	1. The actor submits the application by filling all details and uploading files. 2. The actor receives a confirmation email for application success.
Triggers	The actor applies for a course.
Alternative sequence	N/A

Error sequence	Email is not sent to the student.
Postconditions	Email received and student knows that application has been sent.

Name	Status email
Description	This use case allows students to receive emails when the status of the application has changed.
Actors	Student
Preconditions	Actor is connected to the internet. Actor is registered in the system.
Basic flow of events	1. The registrar check student's application. 2. The registrar changes the status of application to either "Approved or Declined" 3. The actor receives an email telling about the change in status of application.
Triggers	The actor applies for a course.
Alternative sequence	N/A
Error sequence	Email is not sent to the student.
Postconditions	Email received and student knows that status has been changed.

Name	Updating schedule
Description	This use case allows university registrars to upload a new schedule for courses.
Actors	Registrar

Preconditions	Registrar is connected to the internet. Registrar logged in.
Basic flow of events	1. The registrar clicks upload new schedule button. 2. Registrar chooses a file. 3. Registrar clicks upload. 4. New schedule is posted on the system.
Triggers	Registrar chooses a file and clicks upload button.
Alternative sequence	Registrar clicks cancel.
Error sequence	Invalid file.
Postconditions	New schedule uploaded. Database is updated with the new schedule.

Name	Signature form
Description	This use case allows students to download the signature forms.
Actors	Student
Preconditions	Student is connected to the internet. Student is logged in.
Triggers	Student clicks download button.
Basic flow of events	1. Students clicks the download form button. 2. Form is fetched from the database. 3. Form is downloaded into the students computer.
Alternative sequence	Student clicks back/cancel download.
Error sequence	File not found. Download fails.
Postconditions	Download completed.

C-level

Name	Contact page
Description	This use case allows the view of the contact page for the university registrars.
Actors	Public
Preconditions	Internet connection.
Triggers	Contact page link clicked.
Basic flow of events	1. Actor clicks on the link. 2. Actor is redirected to the contact page on the application.
Alternative sequence	Stop page load or click another page.
Error sequence	Page not found.
Postconditions	Information is viewed.

Name	Delete account
Description	This use case allows a student to delete his account
Actors	Student
Preconditions	Internet connection. Student is logged in.
Triggers	Student clicks delete account button.
Basic flow of events	1. Student clicks the delete account button. 2. System displays a conformation message. 3. Students conforms request. 4. Account is deleted. 5. Redirection to the main application page.
Alternative sequence	Student cancels the deletion.

Error sequence	Student not successfully deleted.
Postconditions	Student no longer can apply for courses and sign into the system.

Name	About application
Description	This use case allows the public to read and learn about the application.
Actors	Public, student, registrar
Preconditions	Internet connection.
Triggers	Public clicks the about application link
Basic flow of events	1. Actor clicks on the link. 2. Actor is redirected to the page.
Alternative sequence	Stop page load or click another page.
Error sequence	Page not found.
Postconditions	Information is viewed.

Name	Privacy page.
Description	This use case allows the public to read and learn about privacy of the application.
Actors	Public, student, registrar
Preconditions	Internet connection.
Triggers	Public clicks the privacy policy link
Basic flow of events	1. Actor clicks on the link. 2. Actor is redirected to the page.

Alternative sequence	Stop page load or click another page.
Error sequence	Page not found.
Postconditions	Information is viewed.

Name	About AcrossEC
Description	This use case allows the students to read and learn about the application.
Actors	Student.
Preconditions	Internet connection. Student is logged in.
Triggers	Student clicks the about us link.
Basic flow of events	1. Student clicks on about us link. 2. Student is redirected to the about us page.
Alternative sequence	Stop page load or click another page.
Error sequence	Page not found.
Postconditions	Information is viewed.

Test Cases

As we will be doing the tests for this website, we will test our website during the implementation phase where Cucumber and Rspec will be the testing tools. They will facilitate our development and help us identify errors and bugs throughout the implementation and the coding process.

The application will be tested mainly on two operating systems, Mac OSX and Windows. On both of these operation systems, the application would be tested on different web browsers; Firefox, Safari, Google Chrome and Internet Explorer. These test will be conducted in order to ensure that the application is presentable on different browsers.

In order to keep track of all the bugs we encounter during the testing phase, all bugs will be recorded in our Google documents folders. The developer/tester will capture a screen shot of the bug along with the location of bug, time and date, and enter the status of the bug. This will allow us to know ahead of time if we are still facing bugs in the development of the application. The table that bugs will be recored in looks as follows:

Location of bug	Date & time	Screen Shot	Status

During the design phases, we will brainstorm and come up with a list of tasks to be given to a number of students and registrars to do on the implemented website. We will have the process of doing these tasks recorded using screen recorder tool. We will also ask the users to think aloud to record everything they will think of and say and will know what are them main aspects of the website that would be causing confusion.

As a team, we will be designing a survey that users will have to fill after they finish doing the tasks given to them. The questions of the survey will be designed to get as much feedback as possible from the users to have a clearer idea of how the website should be improved.

A-Level

Student creates an Account			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for invalid input: Test for invalid First name Test for invalid Last name Test for invalid Email Test for invalid Number Test for invalid University Name Test for invalid University ID Test for invalid Major Test for invalid Class number Test for successful Sign up	7awra Abdallah, nabehih@gmail.com zghalib@hotmail.com *blank 337509222 22346545 3375e493 Qatar University Carnegy Mellon University, 638909m Nursing 2022 Valid personal information	Error Flash message Successful Sign up!	**Tester** **Result**

Admin creates an Account for Registrar			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for invalid input: Test for invalid First name Test for invalid Last name Test for invalid Email	Bisnca V1n Zundert bianca7z@qtar.cmu.edu b!ancaz@qatar.cmu.edu	Error Flash message	**Tester** **Result**

Admin creates an Account for Registrar			
Test for invalid Number Test for invalid University Name Test for successful Sign up	337509222 22346545 3375e493 Georgetown University *blank Valid information	Successful sign up.	

Registrar logs in			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for invalid input: Test for invalid Username Test for invalid Password Successful Sign in	bianva! bianv@n *blank bianca12# bianca1234 Valid username and password entered	Error Flash message Successful sign in. Directed to home page.	**Tester** **Result**

Registrar upload schedule			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for invalid input: Test for invalid file type	Schedule_fall_TAMUQ_2013.docx Schedule_fall_TAMUQ_2013.png	Error Flash message	**Tester** **Result**

Students apply for courses			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for invalid input: Test for invalid course number Test for invalid Passport copy file Test for invalid Signature form file Test for successful applying process.	37-33a (For CMUQ) passport_nawalbehih.png passport_nawalbehih.docx signatureform_nawalbehih.png signatureform_nawalbehih.docx Insert valid course number and valid files.	Error Flash message Application submitted	**Tester** **Result**

Registrars view applications			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for valid view process: Valid log in Valid view of applicants	Valid username and password Click on “View applicants”	Successful log in View list of applications waiting for approve or decline.	**Tester** **Result**

Students can upload files			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for valid view process: Valid log in Valid view of applicants	Valid username and password Click on “View applicants”	Successful log in View list of applications waiting for approve or decline.	**Tester** **Result**

B-Level

Registrars approve/decline applications			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for taking decision process: Valid log in Valid view of applicants Click on approve/decline button	Valid username and password Click on “View applicants” Application is accepted or declined.	Successful log in View list of applications waiting for approve or decline. User is added to the list of approved applicants or to the declined applicants.	**Tester** **Result**

Students view decision			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for valid view of the application status: Valid log in Click on “Application Status”	Valid username and password Valid access to the Application status page.	Successful log in Application status appears on the page “Application accepted or Application declined”	**Tester** **Result**

Student edit information			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test if the user logged in	User clicks edit button	User is taken to edit page Changes are successfully saved	**Tester**
Test for saved name	'Nawal'	The user name changed from Nawel to Nawal	**Result**
Test for saved email	hmabdull@qatar.cmu.edu	The email field change from Hawra.92@gmail.com to hmabdull@qatar.cmu.edu	
Test for saved student_no	24092339	Student_no change	
Test for saved mobile	77118331	Student mobile number change	
Test for saved major	Information systems	Student major change	
Test for saved year_of_graduation	2015	Student year_of_graduation change Changes can't be save Error message from program	
Test for invalid input: Test for invalid name	In student no. having na-wa11202	Can't save the new information, data type is not correct	
Test for invalid name	Nas22a		
Test for invalid email	Jsk!sa.d		
Test for invalid student_no	Cmu1221-3		
Test for invalid mobile	661mm123		
Test for invalid major	space/Blank		
Test for invalid year_of_graduation	May2011		

Registrars can update schedule			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Test for registrar account	Registrar can view update schedule button	Registrar click on update schedule button	
			Tester
Registrar upload new schedule	File.pdf	Program message "The new schedule has successfully uploaded	
			Result
Registrar upload new schedule	File.pdf	Program message " Sorry the file couldn't be recognize please upload the schedule again, Thank you."	

Student downloads Signature Form			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Student can view application	Student click on application	Application page appear	
		The name of the file appear in the field	**Tester**
Student click on upload button for the signature form field	File.pdf		
		Program message "Sorry the system does not support files other than PDF, please upload the file again, Thank you"	**Result**
Student click on upload button for the signature form field	File.pdf		

View Contact Page			
Purpose	**Test Data**	**Expected Results**	**Test Date**
User can view contact page	User click on "contact us" in the main navigation bar of the website	User click on "contact us" in the main navigation bar of the website	
			Tester
			Result

C-Level

Student delete an account			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Student view delete button of an account User delete account	User go to account/profile User click delete account button	User can see delete account button Program message "The account has been successfully deleted"	
			Tester
			Result

Public read about Application			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Viewer read about application	Viewer click "About Application" in the global navigation bar	About application page appear	
			Tester
			Result

Public read about privacy page			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Viewer read Privacy Policy	Viewer click "Privacy policy" in the footer	Privacy Policy page appear	
			Tester
			Result

Student Can read About Us			
Purpose	**Test Data**	**Expected Results**	**Test Date**
Viewer read About us	Viewer click "About us in the footer	About us page appear	
			Tester
			Result

Non-functional requirements

Our team has come up with a list of Non-functional Requirements that we think they are important for our team and for the stakeholders that will be using the site. It is very important for the website to meet all the non-functional requirements in order to have a successful project that will be useful and will server the Education City community.
Some of the constraints that we might face throughout the implementation of the website are:

Overload of usage of the site, especially during registration week where many students will be using the website to guarantee a spot in the class they want to register for. Having all education City students that want to cross register to use the website at the same can be a problem as the website might crash of the overload of usage. We will make sure that such problem won't occur once the website is launched.

Overload of database can be another issue as it can make the website slower than usual because of the amount of the application stored in the database. During the testing stage we will focus on this point and make sure it will not affect the website.

Design can be another constraints where the designers of Across EC website may not be able to fully put themselves in the the user's place and look at the website from their prospective. This constraint will be addressed once our team reaches the user testing stage. The students and registrars will be asked to use the website and give feedback about how comfortable where they using the website and wether of not the website was clear and easy to navigate through.

These Non-functional Requirements are:

Non-functional Requirement	**Description**
Functionality	The web-based system is created to serve a good purpose; To help students cross register without doing the paper based process. In order for the system to be functional is has to allow students register for the course they want in any university they would like to choose.
Performance	The web-based system must load in reasonable amount of time as it allow students and registrars to register for courses, view applications, approve and decline them.
Security	The web-based system must resist unauthorized attempts. It should only allow Education City students as the website has private information like the information students include and the passport copy they upload for finishing the registration process.

Non-functional Requirement	Description
Availability	The website should be working properly, active, and available for all Education City students and registrars all the time especially during the registration week.
Usability	The website should be easy to use for all the users. It should be flexible and clear for the students to help them navigate through the website and finish their registration. The website should also be clear enough for the registrars to help them use it to view applications to approve and decline them.
Modifiability	The web-based system must accommodate changes like change in policy or change technology used.
Portability	The website must get along with being deployed through Heroku and other deployment environments. It should also be able to be used through different browsers: Safari, Chrome..
Testability	As the implementation team, we must be able to know what would go wrong with the site: We could find out about test errors using Rspec, Cucumber, and User testing.
Cost and Schedule	As the development team, we must estimate the life time of the web-system and know the cost of it.
Marketability	The website is a good competitor against the paper based system. It is faster, more efficient and greener.
Appropriateness for Organization	Our team must be familiar with the software structure and new the business process. We should know how is it going to be suitable for users.
Correctness	All system requirements have to be met.
Sensitivity	The team must make measurements and tests regarding the affect of the number of applications on the website.
Calibrability	The website developed must be flexible and adjustable to changes.

The main non functional requirements that we will be enforcing are: Performance, Availability, Usability, Testability, Portability and Sensitivity. The test scenarios for these attributes are as follows:

Performance:

Source	Students
Stimulus	Register for a course
Artifact	Web application
Environment	Normal conditions
Response	Application has been submitted
Measure	Application is submitted to the right registrar in 2 seconds

Availability

Source	Registrars
Stimulus	View Applications
Artifact	Web application
Environment	Extreme conditions
Response	Applications have been viewed by the registrar
Measure	System down for 30 seconds

Usability

Source	Students
Stimulus	Students testing the newly developed website
Artifact	Web application
Environment	Normal conditions
Response	Apply for a course
Measure	It takes 7 minutes for student to apply for a course.

Testability

Source	Students
Stimulus	View application status
Artifact	Web application
Environment	Normal conditions
Response	Error
Measure	It takes up to 30 minutes to fix the error.

Portability

Source	Admin
Stimulus	Deploying the website into Heroku link
Artifact	Web application
Environment	Normal conditions
Response	Website deployed successfully
Measure	It takes 8 minutes for deployment.

Sensitivity

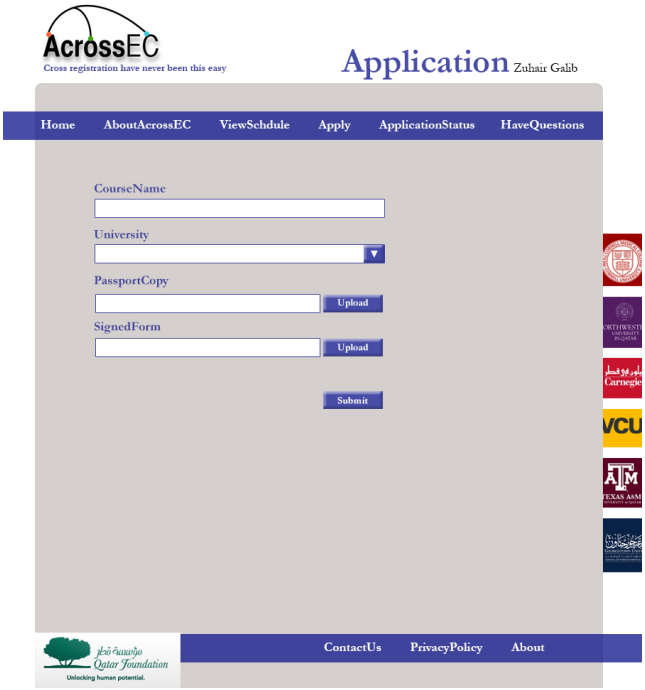
Source	Students
Stimulus	Apply for courses
Artifact	Web application
Environment	Normal conditions
Response	Website works properly.
Measure	Number of applications at the same time. Up to 75 students.

User Experience and Site Layout

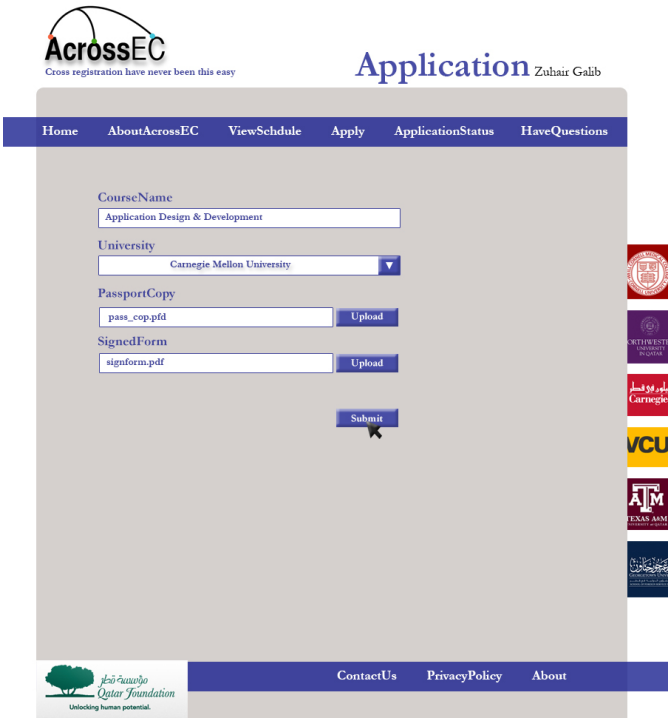
Prototypes

Use Case 1: Student register for a course

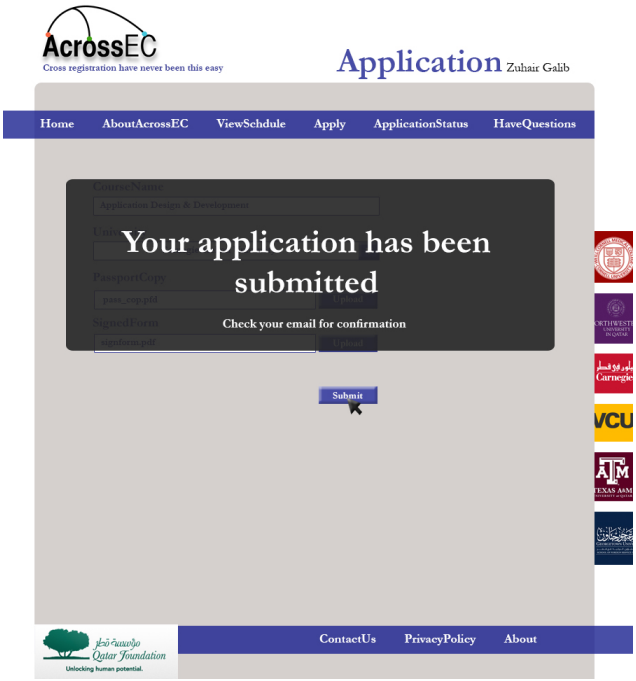
1. The application page for a student who want to apply for a course.



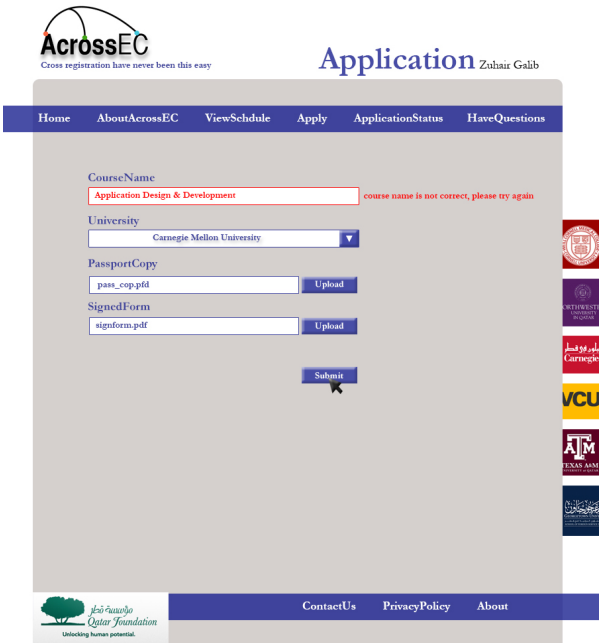
2. The student fill the application, upload the passport copy, upload the signed form and hit the submit button



3. If all the fields are filled with the correct information and the right format, a flash message will appear confirming successful submission and a confirmation email will be sent

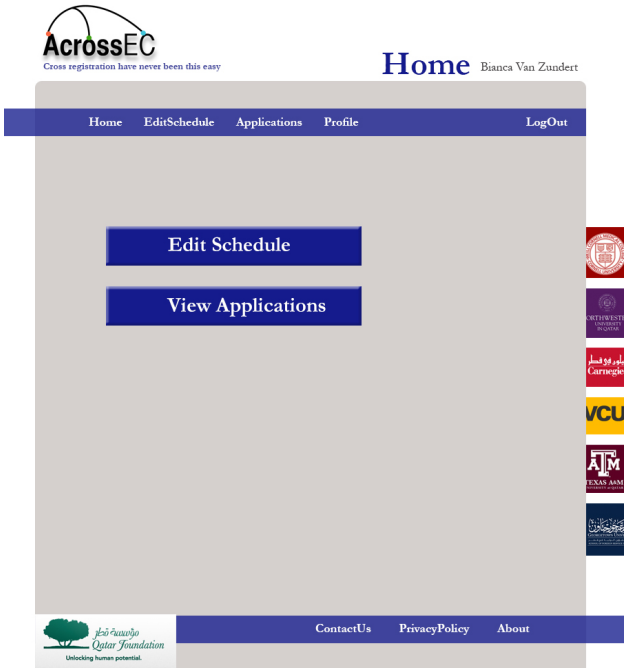


4. In case the student has a wrong input and hit submit, instead of the flash message, the student will be asked to re enter the data with the right format. A red box will appear around the wrong data field, and a message asking for the right data

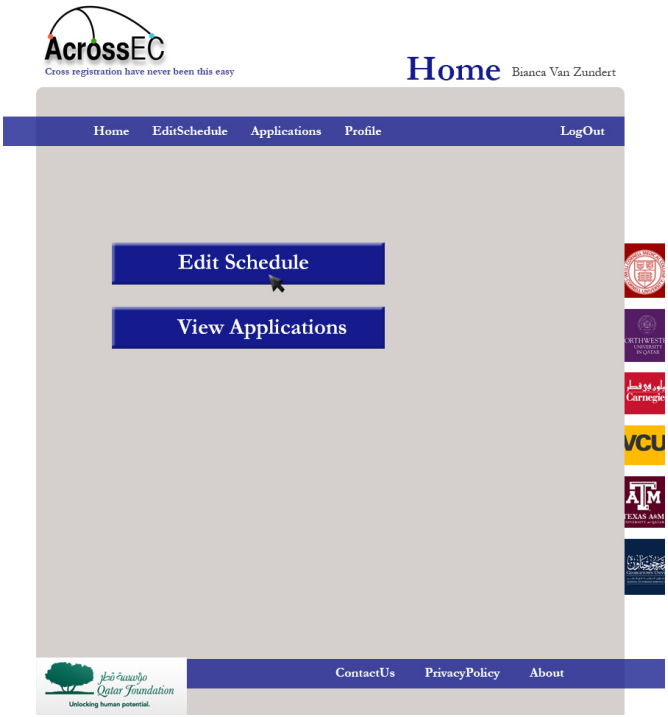


Use Case 2: Registrar Uploading schedule

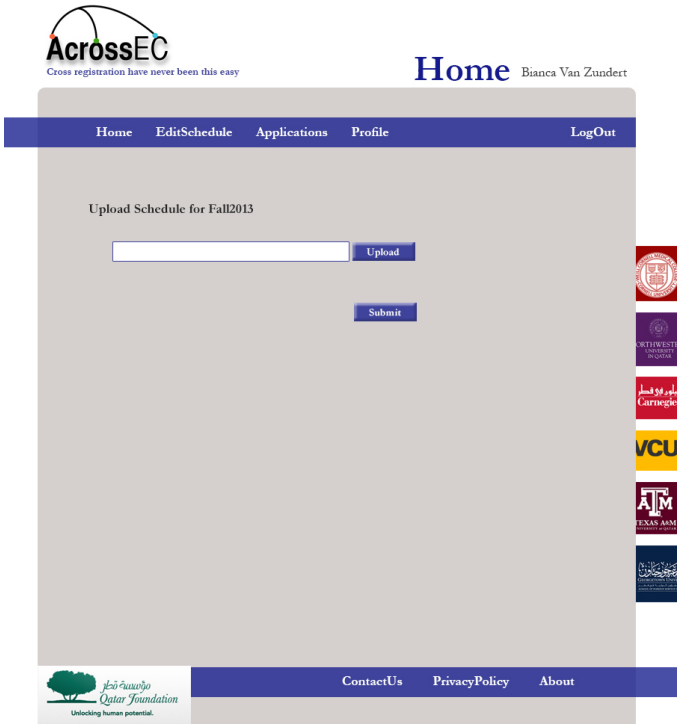
1. The home page for a university registrar, he/she have the choice to upload/edit a schedule or view applications



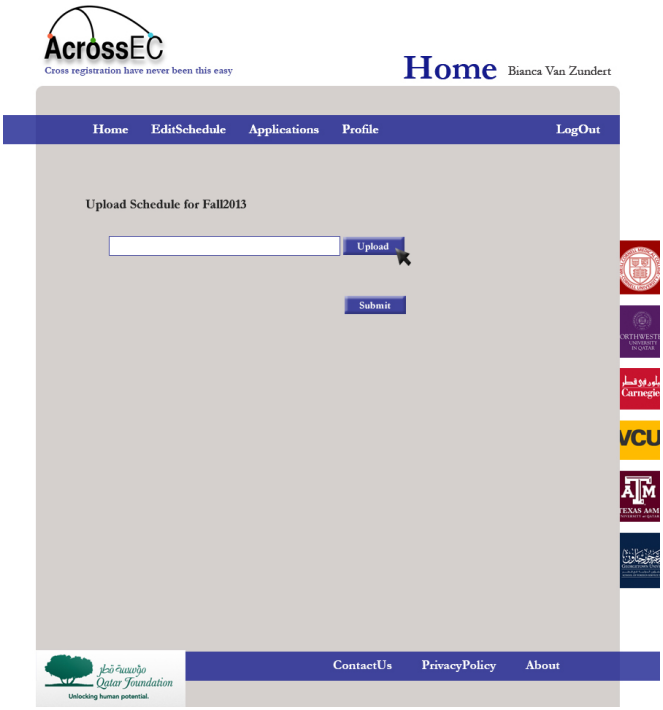
2. The registrar choose to edit a schedule



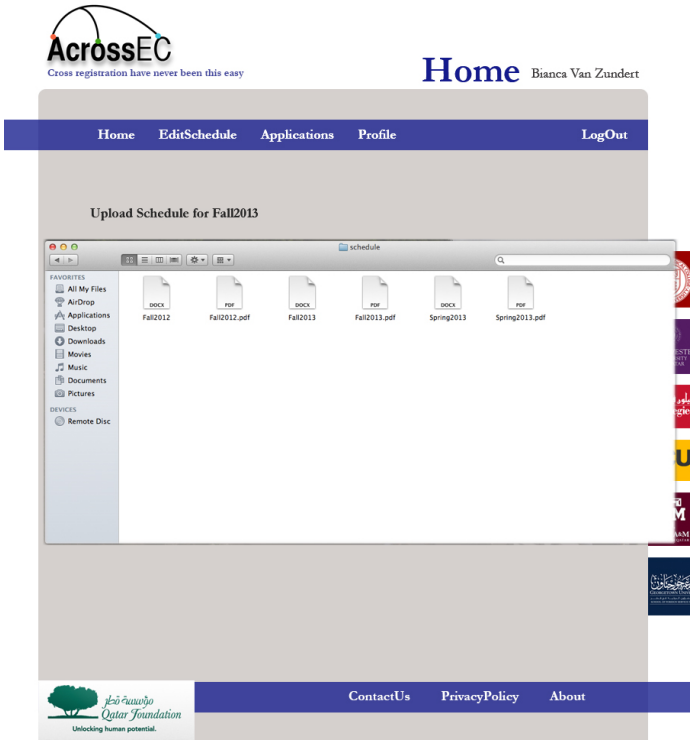
3. An upload page for the schedule shows



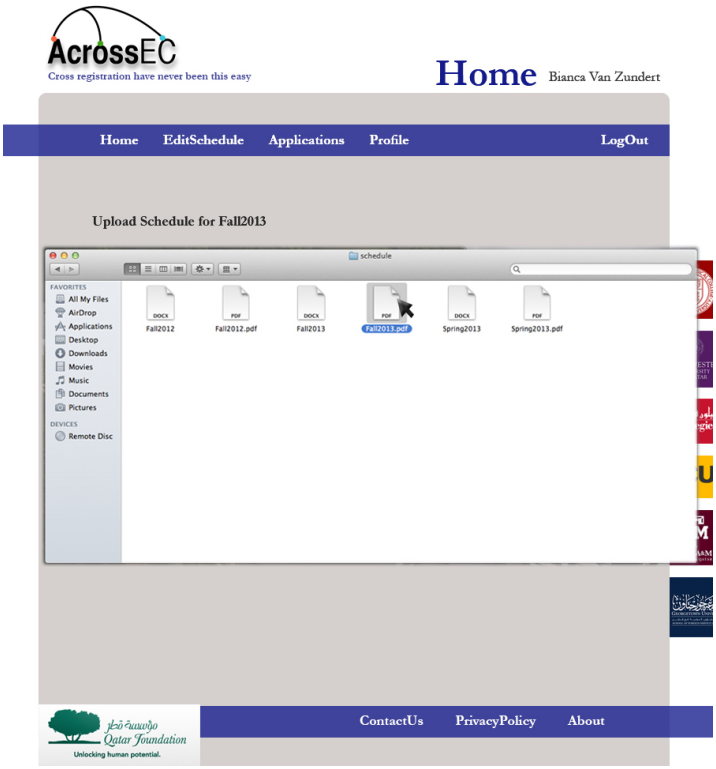
4. The registrar click on the upload button



5. A browse window pop, to let the registrar choose a document from his files

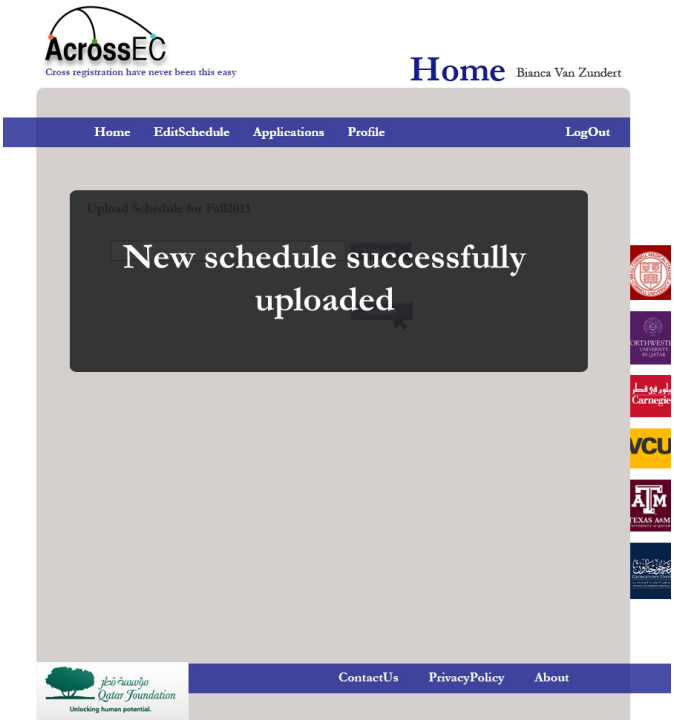


6. The Registrar chooses a pdf file for next Fall

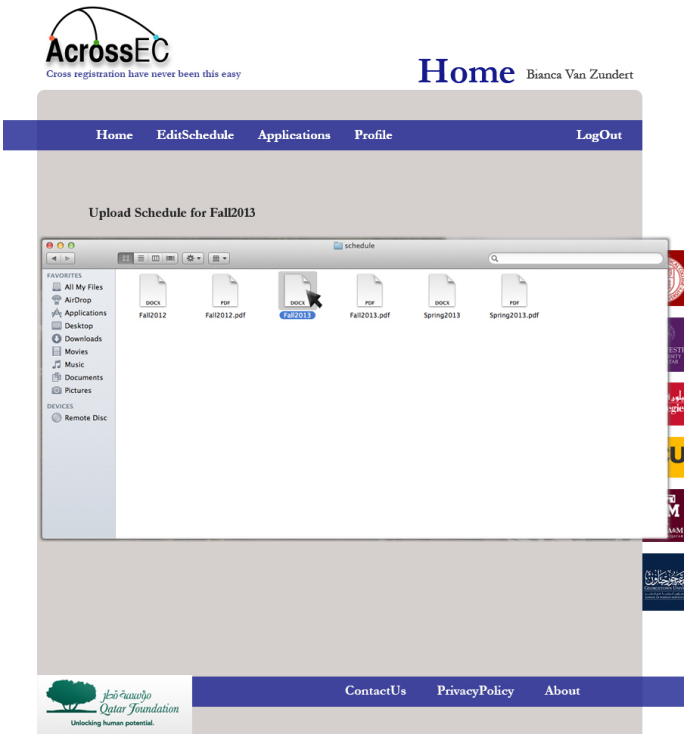


7. Registrar click the submit button

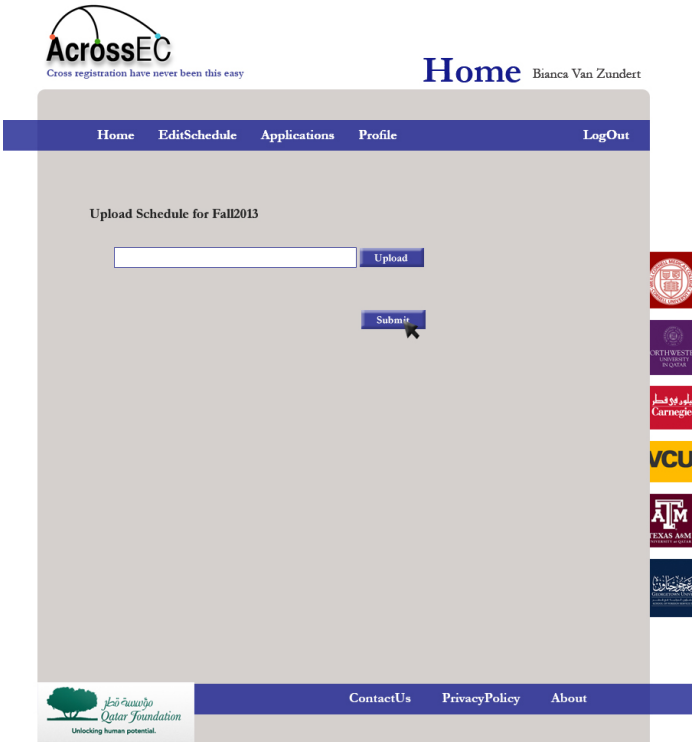
8. A flash message appears stating a successful submission.



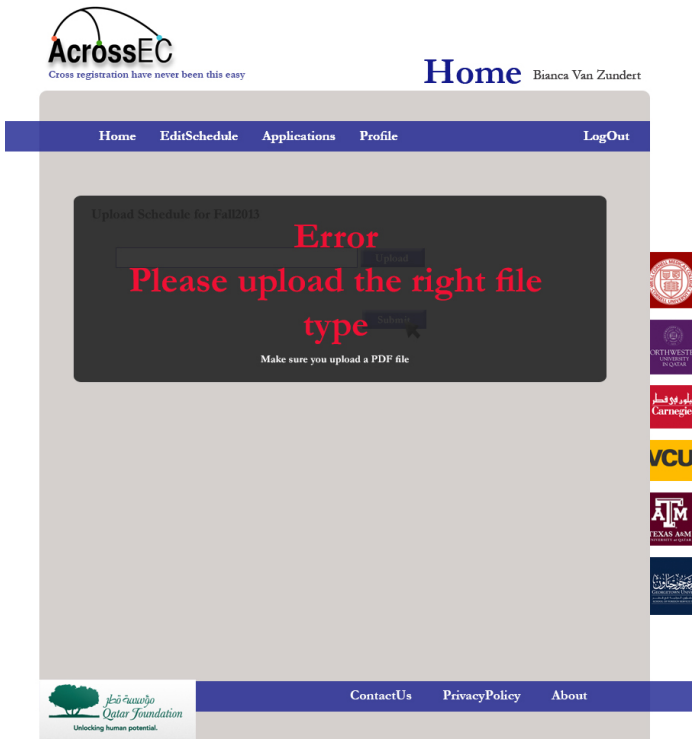
9. If the registrar choose .docx file extension (wrong format)



10. Regsitrar click the submit button

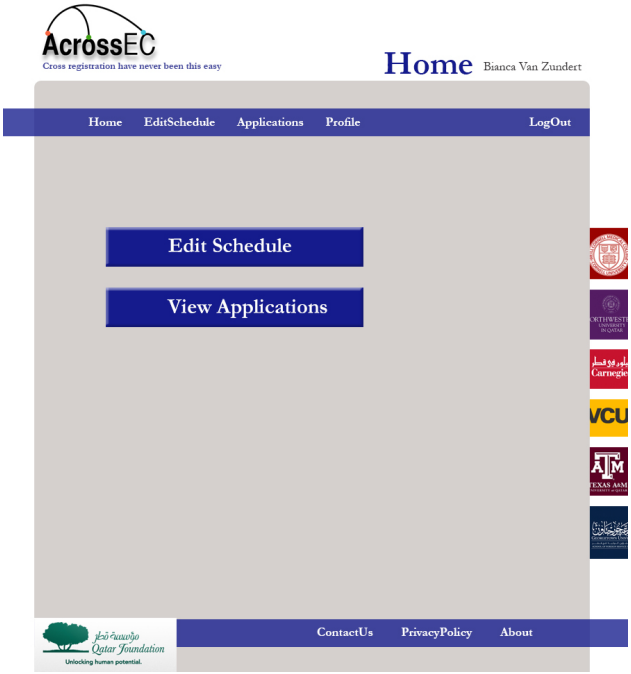


11. A flash message pops telling the user there is an error, the file in not in the right format, upload again and make sure it's a pdf file

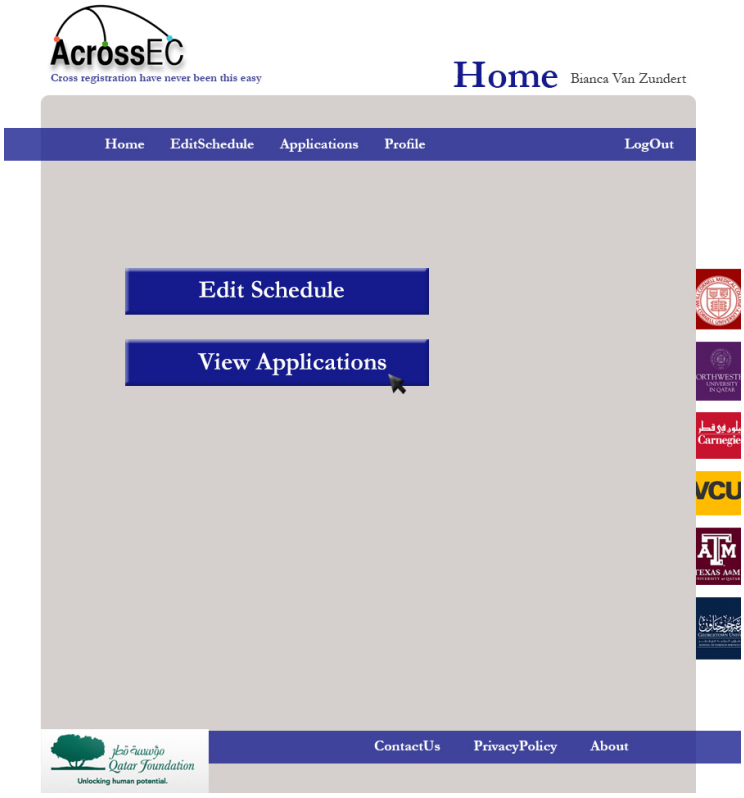


Use Case 3: Registrar approve/decile applications

1. The registrar view home page



2. Registrar clicks on view applications

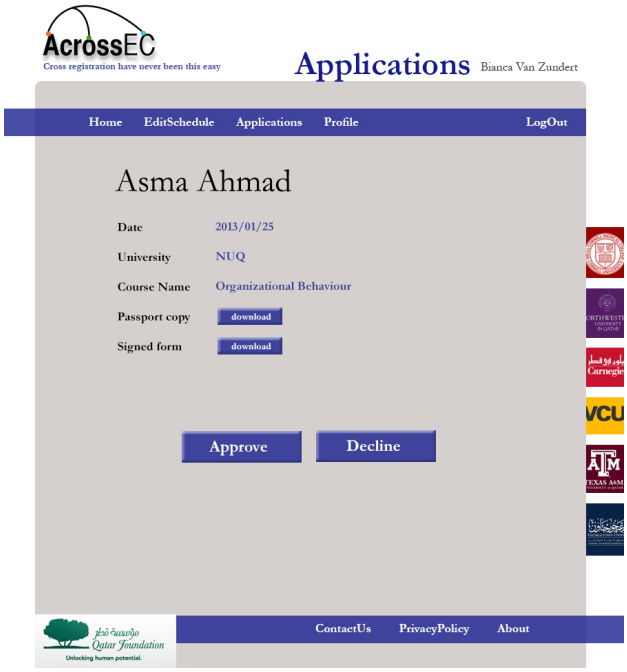


3. A new page appears with three buttons, explaining the status of the application, if it was pending, approved or declined. The Registrar chooses pending applications

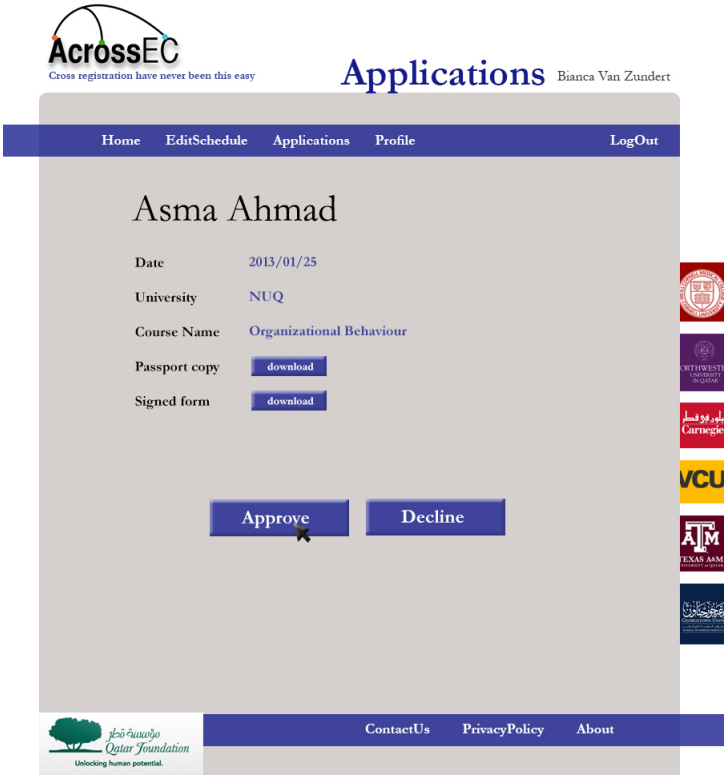
4. A list of all student applications appear, showing the date of the submission, student name, student university, and a view application button

5. The registrar clicks on view application for the second student

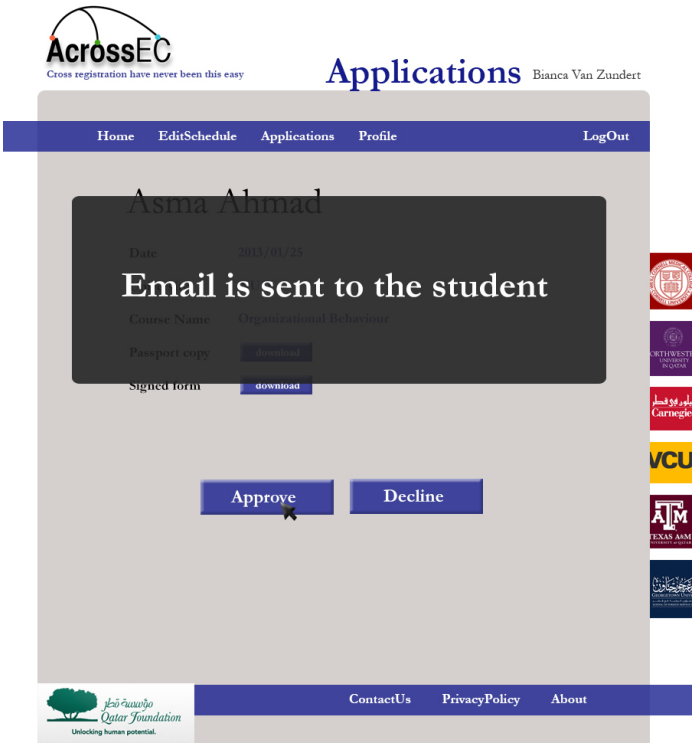
6. A new page appear having the name of the student, information about the course the student is applying for, download button for the passport copy and the signed form, and an approve/ decline buttons



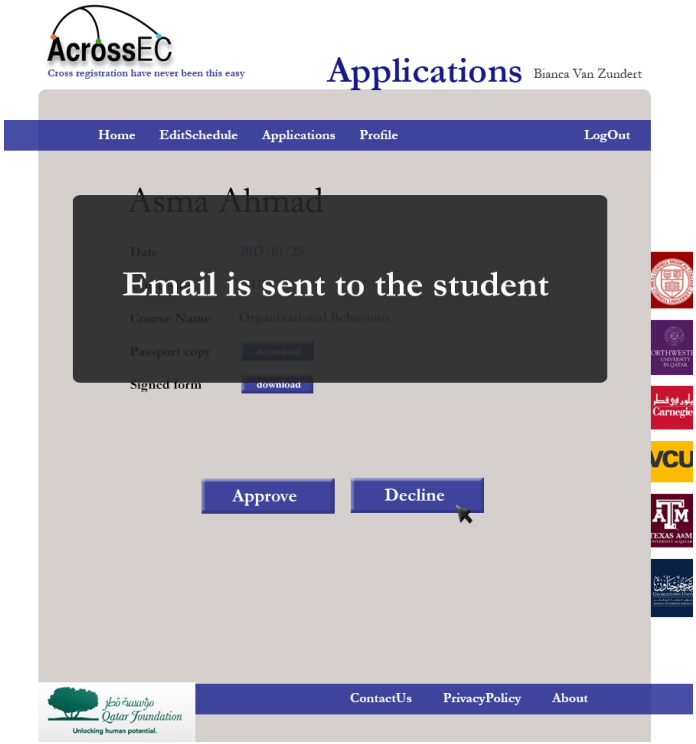
7. The registrar clicks on Approve button



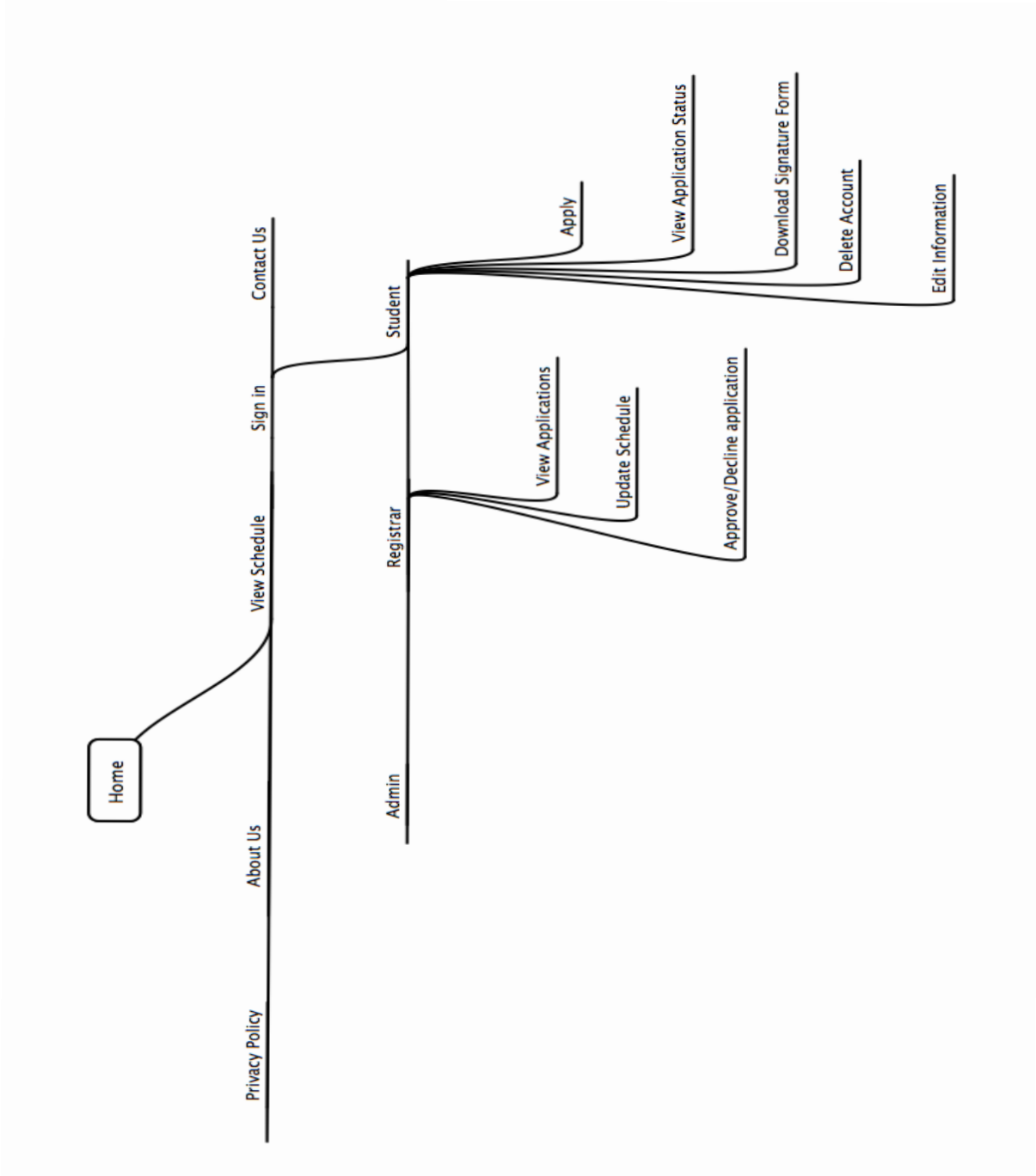
8. A flash message appears, stating that an email is sent to the student



9. In case the registrar clicks on decline button, a flash message will also appear stating that an email is sent to the students



Site map


Home
Privacy Policy
About Us
View Schedule
Sign in
Contact Us
Admin
Registrar
Student
View Applications
Update Schedule
Approve/Decline application
Apply
View Application Status
Download Signature Form
Delete Account
Edit Information

Design Issues

Design related Issues:

- In the prototypes, the design is under development. We are aiming to make our design versatile enough to support all the functionalities with keeping the user requirements at its core. We aim to make the design to support future upgrades.

- In the student account page design, you can see the student logged in, but there is no logout button, this issue needs to be fixed in future designs. There is a logout button for registrars and we will be adding one for the students in the next design.

- At the bottom of the page we added Qatar Foundation's logo, because all the universities enrolled in the system are part of Qatar Foundation (under the roof of QF).

- To personalize the website for users, the registrar is able to see the logo of his/her university when logged in. Considering the fact that it will add extra information on the page and make it crowded, which led to its removal.

- AcrossEC's current logo represents the name of the project, and dots that are on top of the letters representing the universities, and they are connected through the system.

- The color chosen initially was orange, which wasn't aesthetically pleasing. We were looking for something that is professional, simple, elegant, and subtle. Therefore we chose grey and dark blue.

- In order to show the involvement of different universities, we included links to all the universities websites. This will also help the prospect students to acquire about these universities.

- We placed the webpage title on the top right corner of the page in a bigger font to help navigate users.

- The name of the person is shown next to the webpage title, to make the website more personalized, and make sure that the person is logged in.

System-design related Issues:

- We developed the prototypes for 3 use-cases to provide a look and feel of the product.

- We have a pop out confirmation appearing as a flash message rather than a new page, as it will be more efficient and convenient.

- For specific options, we will put drop down boxes in the application for accuracy and to avoid spelling mistakes. It will also save time filling in the forms.

- Every file to be uploaded on the system should be in PDF format, since it presents the document in a nice readable format and is stable. We are concerned with the large size of files in terms of the capability of the web server.

- To complete the application, student needs to acquire the course number from the schedule available on the website and put it in the application form. This is to allow students to register for courses even if the latest schedule is not up yet due to any delay.

- While viewing the application, the registrar would be referring back to the schedule in order to know for what course the student has applied using the reference of course number.

- We are making the assumption that every user will be knowledgeable about how to convert files to PDF format, as people use it in their daily lives.

- Initially, we planned to integrate a database with all the courses as individual objects within the system, rather than PDF. This is to enhance the search capabilities to helps students easily find their desired courses. But due to the limited time frame, we set it as one of our future plans.

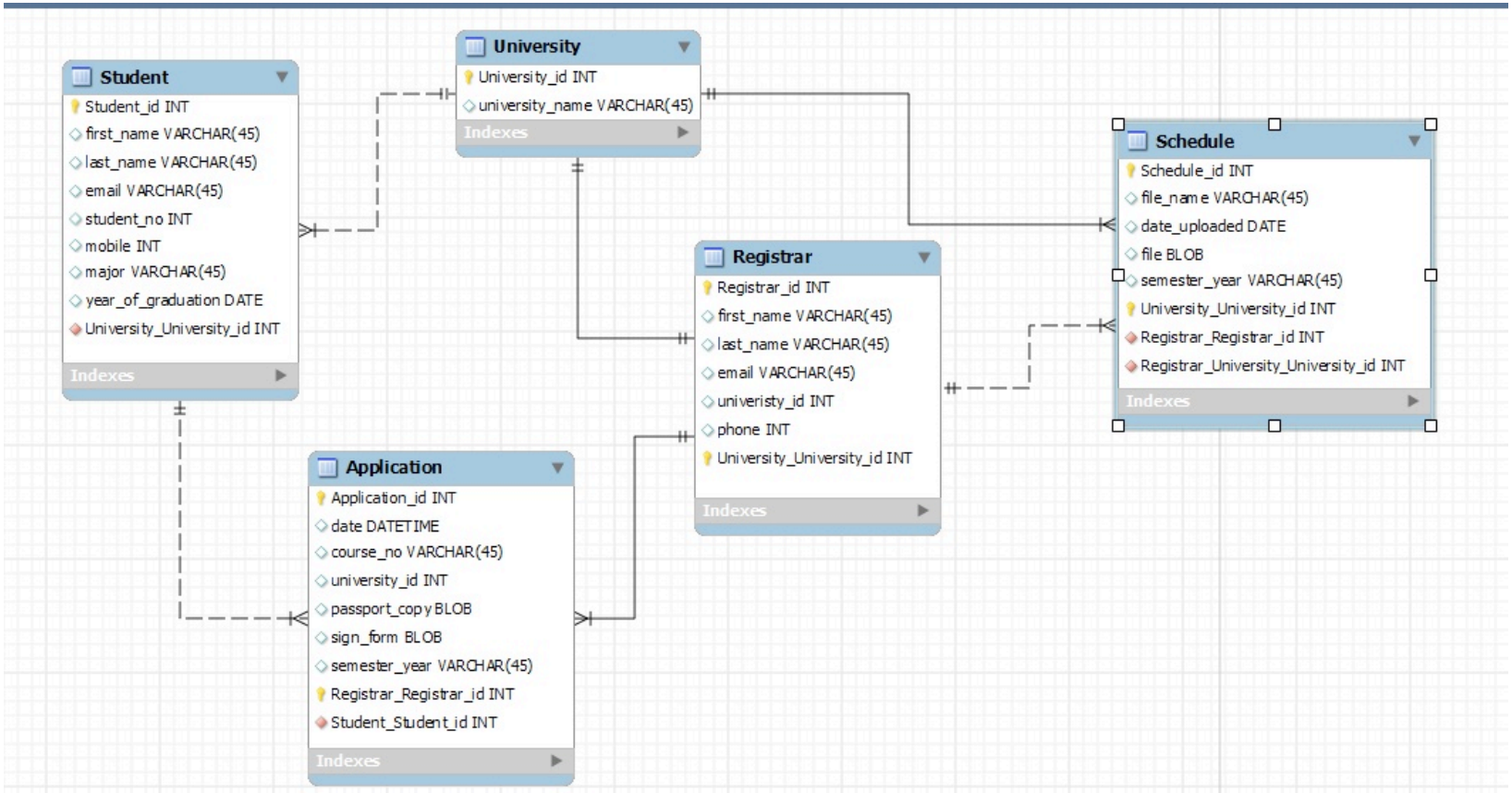
- One of our aims is to make the website look presentable on all browsers, which include Firefox, Google Chrome, Safari and Internet Explorer.

Technical Architecture & Design

Description

Main technologies	Ruby (version 1.9.3) & Rails (version 3.2.11)
Gems will be used for	Testing (rspec & cucumber), Email function (Mail gem), file upload function (Paperclip gem) and website hosting(Heroku gem)
Operating system	Ubuntu (Linux) and Mac OS
Browsers for website testing	Internet Explorer, Mozilla Firefox, Safari and Chrome
APIs	Ruby (http://www.ruby-doc.org/core-1.9.3/) Rails (http://api.rubyonrails.org/)
Website hosting platform	Heroku.com

ERD


Student
Student_id INT
first_name VARCHAR(45)
last_name VARCHAR(45)
email VARCHAR(45)
student_no INT
mobile INT
major VARCHAR(45)
year_of_graduation DATE
University_University_id INT
Indexes
University
University_id INT
university_name VARCHAR(45)
Indexes
Schedule
Schedule_id INT
file_name VARCHAR(45)
date_uploaded DATE
file BLOB
semester_year VARCHAR(45)
University_University_id INT
Registrar_Registrar_id INT
Registrar_University_University_id INT
Indexes
Registrar
Registrar_id INT
first_name VARCHAR(45)
last_name VARCHAR(45)
email VARCHAR(45)
univeristy_id INT
phone INT
University_University_id INT
Indexes
Application
Application_id INT
date DATETIME
course_no VARCHAR(45)
university_id INT
passport_copy BLOB
sign_form BLOB
semester_year VARCHAR(45)
Registrar_Registrar_id INT
Student_Student_id INT
Indexes

Data Model

University	
Id	name

Student								
Id	first_name	last_name	email	student_no	mobile_no	major	year_of_graduation	university_id

Registrar					
Id	first_name	last_name	email	phone_no	university_id

Schedule				
Id	file	semester	registrar_id	university_id

Application								
Id	date	course_no	passport_copy	form	semester	registrar_id	student_id	university_id

Bold: Primary key

Italic: Foreign key

Data dictionary

Attribute name	Attribute type	Attributes restrictions	Attribute default	Attribute description
Student				
Student_id	int	NN		Student id is the identifying number for all students registered in the system
first_name	Varchar(45)			The student first name
last_name	Varchar(45)			The student last name
email	Varchar(45)			Student email address (to send confirmation and acceptance notification)
student_no	int			The student number in his enrolled university
mobile	int			Student mobile number
major	Varchar(45)			Student major in his university
year_of_graduation	date			Expected year of graduation
Application				
Application_id	int	NN		The application ID is a unique number that identifies each application
date	datetime			The date the application is filled on
course_no	Varchar(45)			The course number the student want to apply for
university_id	int			University id for the university the student want to cross register for
passport_copy	blob			A copy of the student passport
sign_form	blob			A copy of the signed form
semester_year	Varchar(45)			Which semester and year the student is applying. E.g.: Spring 2013
Registrar				
Registar_id	int	NN		The registrar unique id
first_name	Varchar(45)			Registrar first name
last_name	Varchar(45)			Registrar last name
email	Varchar(45)			Registrar email address
university_id	int			University Id that the registrar belongs to
phone	Int			Registrar phone number (For contact purposes)
Schedule				
Schedule_id	int	NN		Schedule Id number
file_name	Varchar(45)			The file name
date_uploaded	Date			Date the file uploaded

file	Blob			The schedule file
semester_year	Varchar(45)			The schedule uploaded for which year semester
University				
University_id	int	NN		University unique id number
university_name	Varchar(45)			The university name

Project Plan

To have a better idea of our project and how the work will be divided to finish it by the third sprint, we have distributed the time we have on the stages, use cases, and the tasks that needs to get done. The distribution of the work during the implementation, the design and the testing phase was based on our skills, strengths, and weaknesses.

As we have mutually agreed, the distribution of roles and work is listed below in this table.

Zuhair Ghalib	Qality Assurance Manager, The programmer, and The Tester.
Nawal Behih	Scrum Master, The programmer, The Tester, and the Architect.
Hawra Abdallah	Cheif of design, Product Owner, The Database Administrator and the UI Designer.
Talal Shoeib	Technical Lead, The Programmer, and The Tester.

Based on the work that needs to be done in the next three sprints, We have been able to make an estimation for the number of hours that needs to be spent on each task and use case. We have agreed that we will focus on the A-level use cases. We will implement all the A-level use cases in the first two sprints then move to the B-level and C-level use cases. The web development will be done through Ruby on rails mainly. CSS styling will be also part of each sprint that will be handled by chief of design.
The team members have made their best in brainstorming and estimating number the time that needs to be spent on each task and each stage. We haven't mentioned the estimation of the time as the team might have overestimated or under estimated time that needs to be spent on each task.

Below is a detailed estimation of the tasks and the time will be spent getting them done:

Phase 3

Task 1: Set deadlines for the coming stages.			
Sub Tasks	**Description**	**Team Member handling it**	**Number of hours**
Setting final deadlines for Sprint 1	Discuss as a group the other courses, deadlines, suitable and preferred deadlines and deciding after mutual agreement on the deadlines for the sub tasks.	4 hours	Group

Task 2: Implementation of the Use cases(A-level 1 to 6)			
Sub Task	**Description**	**Team Member handling it**	**Number of hours**
Coding	Student Model(Validation, Scope, Rspec&Cucumber..)	Nawal	24 hours
	Registrar Model(Validation, Scope, Rspec&Cucumber..)	Zuhair	24 hours
	Application Model(Validation, Scope, Rspec&Cucumber..)	Nawal	24 hours
	Schedule Model (Validation, Scope, Rspec&Cucumber...)	Zuhair	24 hours
	University Model(Validation, Scope, Rspec&Cucumber..)	Talal	24 hours
	Authentication and Authorization	Talal	24 hours
Setting up Database	Research and meeting with professor Daniel Phelps for questions	Group	8 hours
	Connecting the Database to the website implemented.	Hawra	9 hours
Design	Sketches	Nawal and Hawra	8 hours (4 hours each)
	Converting Prototypes to HTML and CSS	Hawra	20 hours

Task 2: Implementation of the Use cases(A-level 1 to 6)			
User Testing	Brainstorming and making a list of tasks for the users to do as they use the implemented website	Group	12 hours (3 hours each)
	Design a survey for the users to ask them questions about the website and get feedback from them.	Hawra	5 hours
	12 students and 4 registrars will be asked to do the User testing.	Group	15 minutes for each user testing 4 hours spent by each one of the team member. Total of 16 hours.

Task 3: Project Management		
Sub Tasks	**Team Member handling it**	**Number of hours**
Progress Report	Zuhair	3 hours
Feasibility Update	Talal	1 hour
QA Manager's Report	Zuhair	2 hours
Time Accounting	Nawal	3 hours
Project Metrics	Talal	1 hour
Risk Management	Zuhair	1 hours
Problem Status	Nawal	1 hours

Phase 4

Task 1: Set deadlines for the coming stages.			
Sub Tasks	**Description**	**Team Member handling it**	**Number of hours**
Setting final deadlines for Sprint 2	Discuss as a group the coming deadlines, suitable and preferred deadlines and deciding after mutual agreement on the final deadlines for that sprint.	Group	4 hours (1 hour each)

Task 2: Implementation of the Use cases(A-level 7&8, B-level 1 to 5)			
Sub Task	**Description**	**Team Member handling it**	**Number of hours**
Coding	code viewing list of applications	Nawal	10 hours
	Code Students uploading files	Nawal	10 hours
	Code sending an email once an application is submitted	Talal	10 hours
	Code sending an email once an application is approved or declined.	Nawal	10 hours
	Code Student view Application status page.	Talal	8 hours
	Code the page students can edit information.	Zuhair	4 hours
	Code uploading file for registrars.	Zuhair	8 hours
Testing the sensitivity of the website.	As the process of registration and viewing applications will be done, we need to make sure that the website can still work efficiently no matter how many applications are submitted.	Nawal, Talal, and Zuhair	12 hours (4 hours each)
Design	Updating HTML and CSS	Hawra	4 hours
	Adding HTML and CSS to the use cases and features implemented.	Hawra	14 hours
User Testing	Brainstorming and making a second list of tasks for the users to do as they use the implemented website	Group	8 hours

Task 2: Implementation of the Use cases(A-level 7&8, B-level 1 to 5)			
	Design a second survey to for the users to ask them questions about the website and get feedback from them	Hawra	5 hours
	4 students and 2 registrars will be asked to do the User testing. Number of users testing the website is to be discussed further once we get to this phase.	Group	15 minutes for each user testing 4 hours spent by each one of the team member. Total of 16 hours.

Task 3: Project Management		
Sub Tasks	**Team Member handling it**	**Number of hours**
Progress Report	Zuhair	3 hours
Feasibility Update	Talal	1 hour
QA Manager's Report	Zuhair	2 hours
Time Accounting	Hawra	3 hours
Project Metrics	Talal	1 hour
Risk Management	Nawal	1 hours
Problem Status	Nawal	1 hours

Phase 5

Task 1: Set deadlines for the coming stages.			
Sub Tasks	**Description**	**Team Member handling it**	**Number of hours**
Setting final deadlines for Sprint 3	As the assignments and finals will be around the corner, we will sit and figure a good timeline for the last phase where each one of us will be comfortably working on the parts assigned to them without any pressure or rush.	Group	4 hours (1 hour each)

Task 2: Implementation of the Use cases(B-level 6 to 9, C-level 1 to 4)			
Sub Task	**Description**	**Team Member handling it**	**Number of hours**
Coding	Code updating schedules for registrars	Nawal	6 hours
	Code Downloading Signature forms for Students	Nawal	8 hours
	Code Contact page	Zuhair	3 hours
	Code for submitting and application and approving or declining it.	Zuhair	6 hours
	Code for deleting an account by student	Nawal	2 hours
	Code the page to read about the website for public and users.	Talal	2 hours
	Code the page to read about privacy for public and users.	Talal	2 hours
	Code the page About Us for more information for the public and users.	Talal	2 hours
Design	Updating HTML and CSS	Hawra	4 hours
	Adding HTML and CSS to the use cases and features implemented.	Hawra	20 hours

Task 2: Implementation of the Use cases(B-level 6 to 9, C-level 1 to 4)			
User Testing	Brainstorming and making a third list of tasks for the users to do as they use the implemented website	Group	8 hours
	Design a third survey to for the users to ask them questions about the web-site and get feedback from them	Hawra	5 hours
	4 students and 2 registrars will be asked to do the User testing.	Group	15 minutes for each user testing 4 hours spent by each one of the team member. Total of 16 hours.

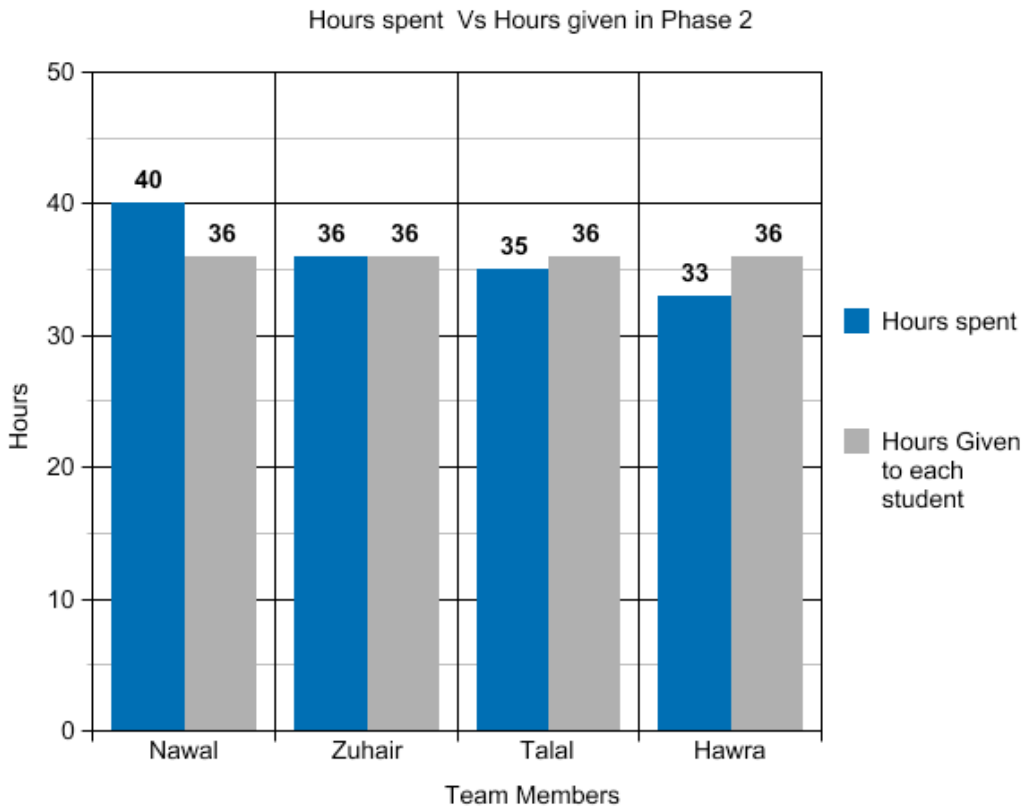
Task 3: Project Management		
Sub Tasks	**Team Member handling it**	**Number of hours**
Progress Report	Zuhair	3 hours
Feasibility Update	Talal	1 hour
QA Manager's Report	Zuhair	2 hours
Time Accounting	Hawra	3 hours
Project Metrics	Talal	1 hour
Risk Management	Nawal	1 hours
Problem Status	Nawal	1 hours

Project Progress Report

Time Accounting

Time accounting was one of the important parts of Project Plan. As we are moving to the implementation phases, we are having changes in the tasks and work that needs to be done. We will be moving from documentation to implementation.

Below are number of hours spent during this phase for each team member compared to the number of hours given for each one during this phase.



During this phase, we have focused on group meetings where we didn't move to documentation until we saw that each one of us was comfortable with the system, features and use cases as well as the parts assigned to them after the meetings and discussions.

Feasibility Update

In phase 1, we badly suffered due to the different point of views among the team about the application. This problem occurred due to lack of collaboration. In order to make a change, we increased the number of meeting and hours. This made us quite effective in work and due to time spent together; we have established an understanding that we are able to reach upon agreements quickly.

We stated and predicted that there might be issues in design factor due to only one design expert among team. It has been a problem, not severe but it took substantial time to finalize our logo and website prototypes. We put significant effort considering the level of criticalness and consulted with Professor Alexander Cheek. Even with the existence of these constraints the effort speaks for itself.

We are still concerned about the hosting of website. We want to ensure the website is up and running when it's needed. For that to happen we need a secure cutting-edge service that would defend the website from service of denial attacks. In addition to hosting website on Heroku, we will be discussing with Professor Daniel Philips on how we can host our website on multi servers and different platforms.

In terms of the technical aspect, we are planning impressively with ease for the sprints ahead. Since we have the expertise of Ruby on Rails, our solutions will quite innovative and efficiently tackle the problem. In few time we are ready with steps to execute for the development phase.

To account for different requirements of different universities, we are developing such a solution in order to cope up with every one's need. Our Use cases, functionalities and designs for every component of website are well thought keeping in mind the externalities.

In short since phase 1, we have improvements in collaboration and thought process which is fostering innovation and problem solving at its best. We believe to achieve significantly.

QA manager's Report

The team's methodology of ensuring the quality of the work is by continuous revision. We approach that by meeting to discuss what needs to be done, followed by individual documentation of different parts, reviewing the completed work as a team, and finally reviewing the final product by me, the quality assurance manager.

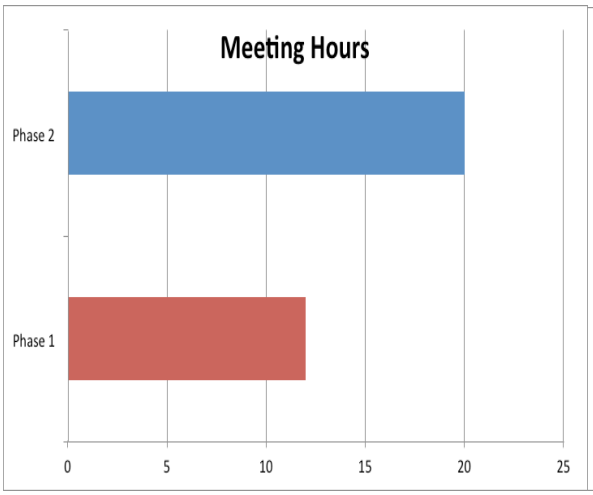
We are encountering some deficiencies that are affecting the quality of work. First, the work getting done on individual bases is not consistence. Second, the work sometimes has bad quality. Third, the work done individually is submitted late, which gives me and the group less time to revise them and ensure the quality.

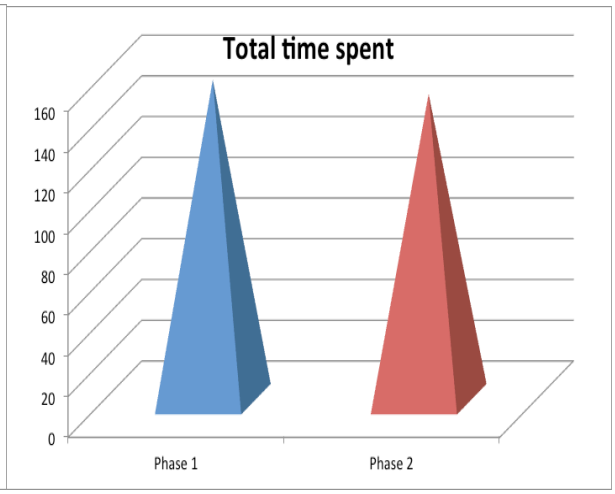
These deficiencies will be remediated through our constant revision to the work. The consistency issue will be remediated by ensuring that we agree on how parts will be done and pay closer attention to the parts. The quality issue will be remediated by making sure that all team members write to their best ability and bad quality work will be returned to be corrected. Lastly, the lateness issue will be remediated by making the deadline more serious and continuously reminding members that work should be done with high quality. In order to ensure better quality, I will dedicate extra time to revise work multiple times and improve the quality.

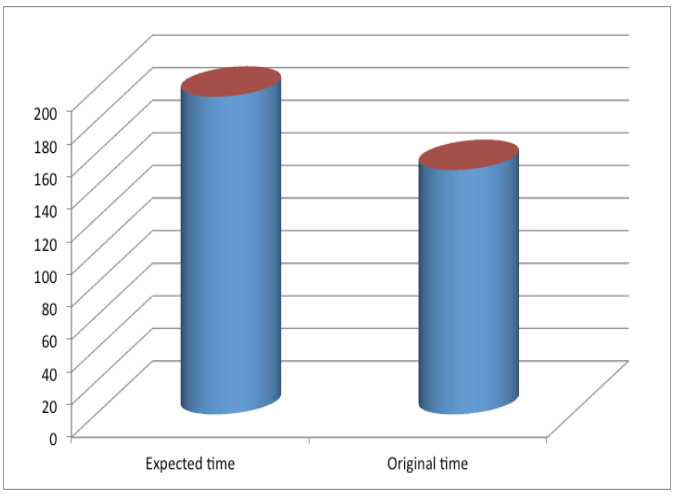
Project Metrics

The team spent 157 hours in phase 1 and 150 in phase 2. This decrease is because of the fact that we as a team have gotten along well increasing the effectiveness in time versus quality aspect. In less time we are delivering more quality work. To achieve this effectiveness, we increased our meeting times. We have been meeting every day of the week to go through each and everything together as a team. That's why we have significant increase in meeting hours as compared to phase 1. Even though we had to fix many wrong things from phase 1 but we managed to get quality work done in time.

We are progressing smoothly and are confident with our planning for the next sprints. We have divided the work for next sprints based on everybody's expertise and schedule. We expected to spend more time in phase 2 but we as a team have shown sheer commitment and worked hard to achieve more in less time.







Risk Management

Risk	Description	Likelihood of occurrence	Magnitude of potential loss	Symptoms of early warning	Abatement strategy
Schedule flaws	With software development it might be hard to schedule because of the intangible nature of the software.	Medium	High	-Bugs occur in development. -The formation of new plans during development.	Set self-deadlines before the scheduled time for submission.
User acceptance	As we are changing the registration process, the university registrars and students might resist the change.	Low	High	-Getting negative feedback about changing the process.	Making the application user friendly by conducting user test.
Development delay	Similar to schedule flaws, the development of the application could be delayed based on some problems that occur.	Medium	High	-Bugs and errors in development. -Not getting the scheduled team submissions on time.	Setting earlier deadlines for the team to finish development before having to submit the final product would eliminate the problem, as it would provide more time to finish.
Security	Security issue associated with people having information online and uploading a passport copy to the site.	Low	Medium	-Users refusing to upload passport copies. -Users constantly asking about the security of the site.	To insure users that the site is secured and that we took into consideration the risk involved before developing the site

Update management	How the system will be updated with new schedules and how the system will update the number of seats of courses.	Medium	Medium	-Application being slow. - Application crashing and not uploading schedules.	Testing would be the optimal method to avoid this risk and to insure that the system is correctly updated.
Team skill set	The team might not have the required skills to complete the application.	Low	High	-Getting too many errors at an early stage. -Having to ask and learn a new programming language from the start.	Choosing a development tool that all the group members are familiar with would minimize the occurrence of this risk.
Time allocation	Team members set enough time to work on the project.	Medium	High	Assigned work is not getting done. Not attending usual meetings.	Early symptoms occurred and we are trying to fix them by setting fixed daily meeting to check the work progress and have a long meeting weekly to sit and work as a group.
Quality of work	The quality of work produced is not sufficient	Medium	Medium	Consistence poor quality of work produced.	The symptoms have occurred, we are currently applying multiple revisions to ensure best quality.

Problem Status

Problem	Description	Resolution
Communication	Group members don't respond to emails/whatsapp group fast enough.	Constant reminders about the importance of communication in teamwork. In addition to encouraging friendly communication. This issue is still open, as members don't always respond to communication.
Commitment	Commitment of members to finish the work and to be ready to review the work before giving the final product.	Having daily meetings to ensure progress and to emphasize the importance on getting the work done on time.
Sharing of work	Work being shared as soon as it is done in order to provide time for the QAM to review and give feedback.	The daily meetings helped to resolve part of the problem. We have added constant reminders to get work done. This issue is still open.
Meeting space	It takes time to find a quite place to conduct our meetings and start working.	This issue is resolved, as we got a fixed office space to meet and work in.