Software Architecture Project

Name	Student ID
Yin Zhou	100314426
Dylan Fernando	100553363

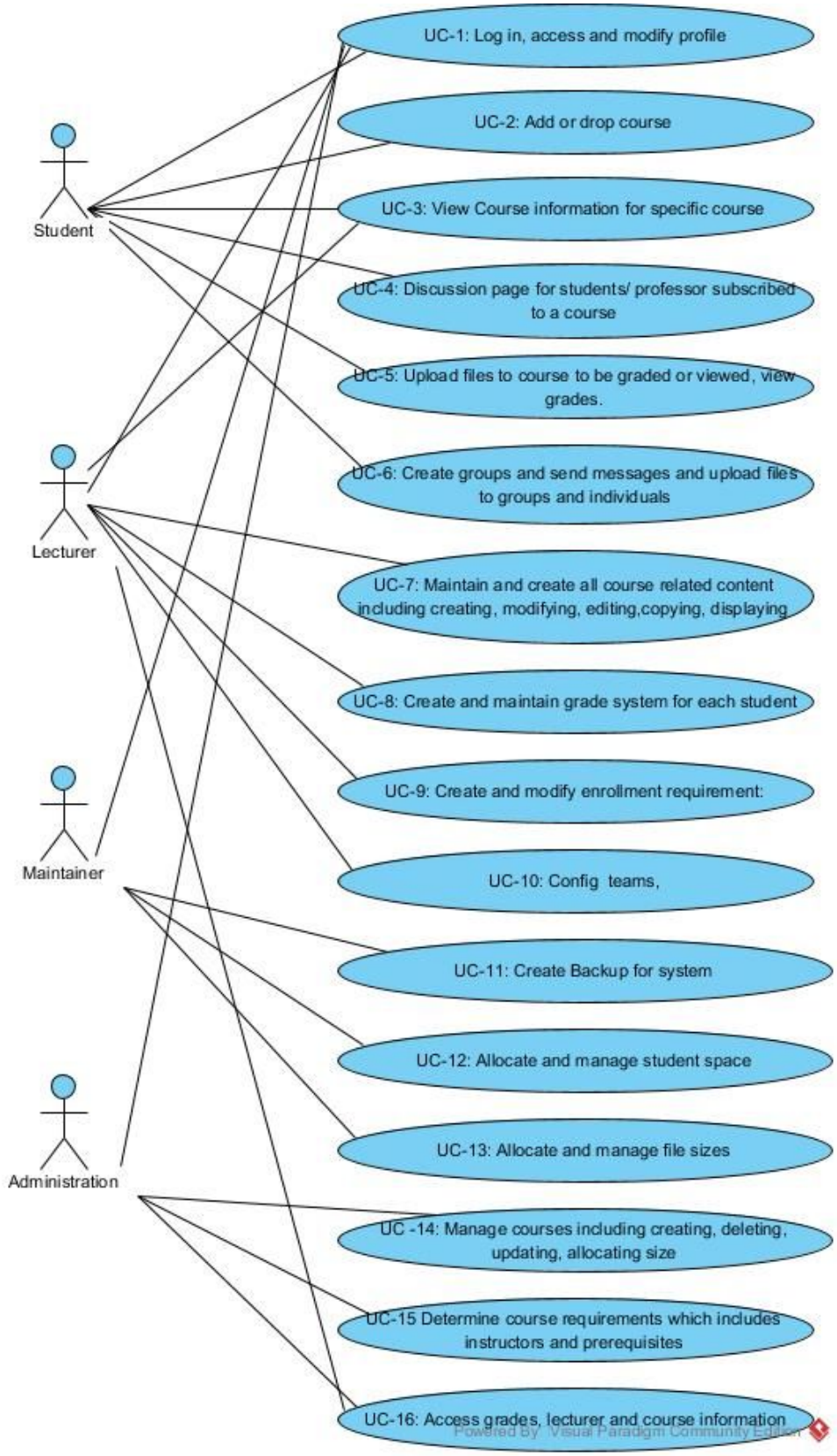
A lab report submitted in fulfillment of the lecture of Software Architecture in the Faculty of Engineering and Applied Science.

Professor: Dr. Ramiro Liscano

Submitted to the University of Ontario Institute of Technology
October 2018

Business Case

Use Case Model



Use Case

Use Case	Description
UC-1	Log in, access and modify profile
UC-2	Add or drop course
UC-3	View Course information for specific cours
UC-4	Discussion page for students/ professor subscribed to a course
UC-5	Upload files to course to be graded or viewed, view grades.
UC-6	Create groups and send messages and upload files to groups and individuals
UC-7	Maintain and create all course related content including creating, modifying, editing,copying, displaying
UC-8	Create and maintain grade system for each student,
UC-9	Create and modify enrollment requirement:
UC-10	Config teams
UC-11	Create Backup for system
UC-12	Allocate and manage student space
UC-13	Allocate and manage file sizes
UC -14	Manage courses including creating, deleting, updating, allocating size.
UC-15	Determine course requirements which includes instructors and prerequisites
UC-16	Access grades, lecturer and course information

Quality Attribute Scenario

ID	QAS	Scenario	Associated UC
QA-1	Security	When user login into the system, system should determine its user privilege and only allowed pre-determined access and should work 100% of the time	All
QA-2	Availability	The system should work 24/7 without any error. During the expected downtime, maximum 4 hours/month.	All
QA-3	Availability + performance	A message should be send to all users 48 hours prior to downtime and the downtime period should be at off-peak hours	All
QA-4	Useability	The system should have an simple UI to navigate and able to get to any content with maximum three clicks	All
QA-5	Performance	The system should able translate and display two language	All
QA-6	Interoperability	The data from the system can be extracted and printed should work 100% of time	All
QA-7	Interoperability	The system should be able to import	UC-7

		roster information into the course router	
QA-8	Maintainability	The system shall be easily maintained	U-11,U-12
QA-9	Testability	The system shall be easily tested	U12
QA-10	Scalability	The system shall be scalable	U-12, U-13
QA-11	Interoperability	The system shall be interoperable with secondary university system	U-12, UC-14
QA-11	Extensibility	The system shall allow administrator makes exception to enrollment	UC-9

Constraints

ID	Constraint
CON-1	The system must be accessed from a web browser (Chrome, IE, Safari) in various platforms: Windows, IOS, Linux/Unix, Tablet, Phones
CON2	Multiple simultaneous connection is required, size > 300
CON-3	High bandwidth is required to allow fast enough download and upload of files and course materials
CON-4	Server with large capacity is required to store all data since day 1
CON-5	A large relational database is required to maintain student progression and enrollment requirement

Architectural Concerns

ID	Concern
CRN-1	Planning architecture of a large relational database with many complexities.
CRN-2	Organizing information without nesting two tables into one.
CRN-3	Strong knowledge of sql to query the views and input into the database, as well as implementation in a variety of languages such as php or asp.net.

The Design Process

ADD Step 1: Review Inputs
We will first review the inputs and determine which requirements will be drivers.

Category	Details
Design Purpose	This system will be used for schools of any size in order for that institution to perform many tasks such as delegate, manage students courses and grades.
Primary Functional Requirements	UC-1 Necessary for all parties in order for the system to work fully. UC-2 Students are the main user of the system and their main goal is to apply to a course. UC-14 Creates most of the required data for

	the system.

Quality Attribute Scenario

ScenarioID	Importance to Customer	Difficulty of implementation According to Architecture
QA-1	High	Low
QA-2	High	High
QA-3	High	Low
QA-4	High	High
QA-5	Low	Low
QA-6	High	Medium
QA-7	High	Low
QA-8	Medium	Medium
QA-9	Low	Low
QA-10	Low	High
QA-11	Medium	Low
QA-12	Medium	Low

Iteration 1: Establishing a System Structure

In this iteration we will mainly be establishing all the concern that are related to the system. After reviewing the Quality attributes some striking attributes that stand out are the following.

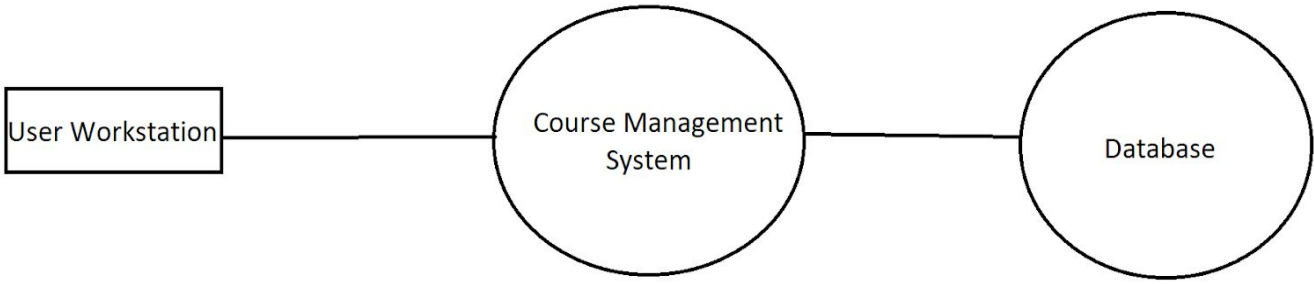
Step 2 Establish Iteration Goal by Selecting Drivers

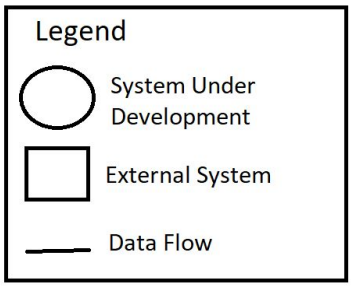
- QA-2 Availability
- QA-4 Useability
- QA-10 Scalability
- CON-2 Multiple simultaneous connection is required, size > 300
- CON-5 A large relational database is required to maintain student progression

and enrollment requirement

- CRN-1 Organizing information without nesting two tables into one.

Context Diagram





Step 4 Choosing one or more Elements to refine

The two elements needed to be refined will be the course management system and the database since they are the only two systems under development.

Design Decisions and Location	**Rationale**
Build the server using Amazon Web Services Cloud Architecture.	One of our concerns CRN-2 states that multiple simultaneous connection is required, size > 300, in order to do this we will need a powerful web hosting service that has a good reputation with handling large amounts of traffic. AWS is a highly scalable type of cloud web hosting architecture which deals with quality attributes that we deemed

	important. QA-10 has high difficulty of implementation and deals with scalability.Cloud architecture will ensure that the web hosting will not be a detriment to the architects since it is a highly scalable cloud service that can handle high traffic for large and small applications. AWS has robust security measures which is a Quality attribute that is of high importance to the customers. This web hosting application is very reliable and is used by many large companies which helps deal with Quality attribute 2 (the system should be running 24/7) which was deemed a high priority and importance for both the architects and users. Cloud web architecture uses data splitting measure to make sure that users cannot have their data shared. This system does not require a lot of computation which is one of cloud architectures main benefits however with this type of architecture the user can choose as much and as little computing resources as they like.
Structure the Database using Model View Controller architecture.	MVC seems the most appropriate and most used for the implementation of the database. The views which would contain course information instructor information etc. would be part of the top layer of the architecture. The middle layer would consist of the controller that can transform the data and is a gateway for changing the way the users see the data from the view. The inner layer is the model which is where the data is stored in the database. This architecture is the most appropriate because of its high maintainability and testability. The architecture can isolate layers so that they arent affected by testing and maintenance. This is one of the most important points of the system. It must be running most of the time with as little downtime as possible, and it must be

	scalable (QA-4 and QA-10).
Bulid the user interface using standard languages such as HTML CSS Javascript	These languages are the easiest to pick up, they are sufficient for the job. The system does not need any overcomplicated fancy UI.The main focus of the UI is to be simple and easy to use. These languages meet this requirement and most developers are familiar with them.

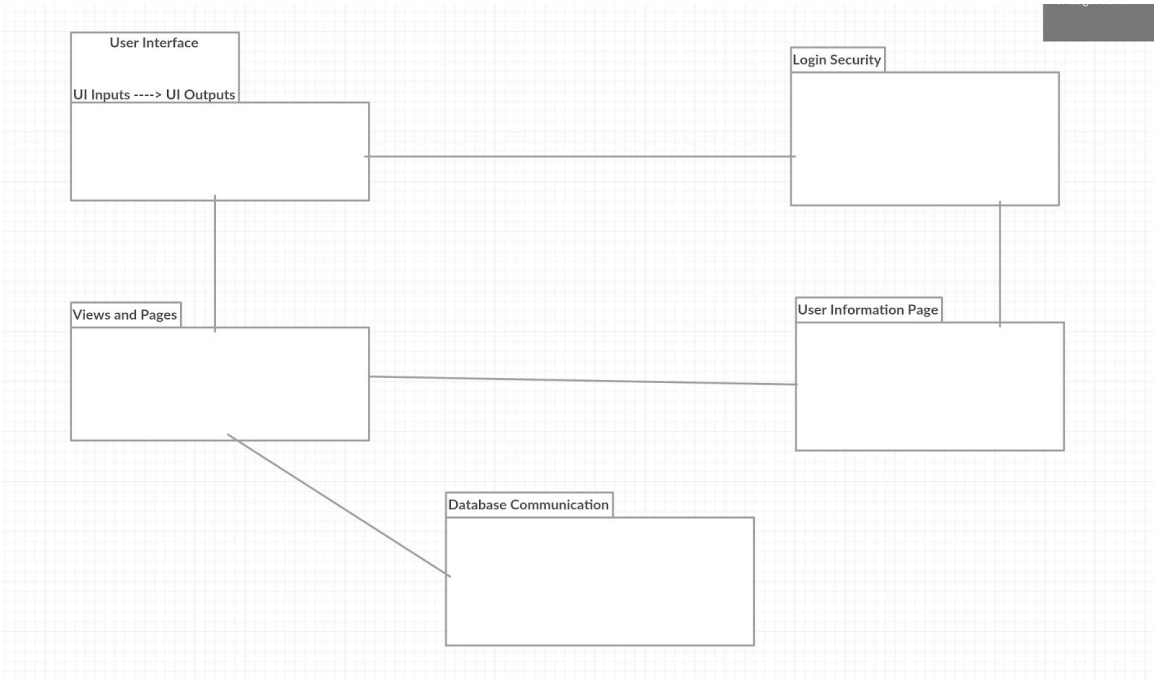
Step 5 Instantiate Architectural Elements, Allocate Responsibilities, and define interfaces

Below we will instantiate the two elements in the system.

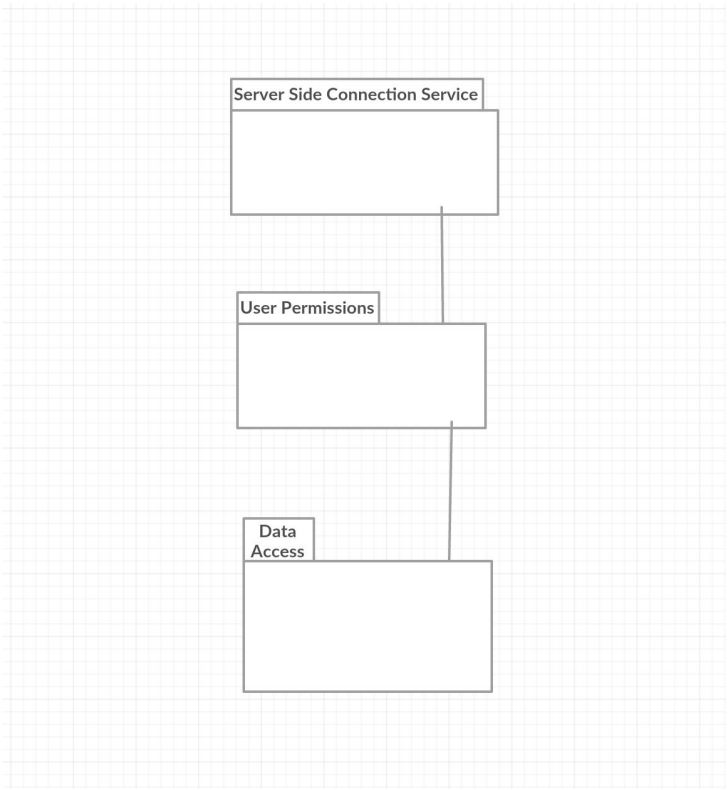
Design Decisions and Location	Rationale
Create cloud server without UI with basic UI pages	The UI will be simple but there is still not enough planning to decide what the UI will have. We know the bare essentials such as a login screen, homepage, and course pages. However we do not have any details so for this iteration these will be the only parts instantiated.
Create data types and tables for the model layer of the database architecture	We have a rough idea on what data will be used in the tables of our database as well as the primary keys.We will instantiate the database by filling some of those tables with our key data types and primary keys.

Step 6 Sketch Views

Client Side



Server Side



Element	Responsibility
User Interface	This layer deals with user input and output so the user can communicate with the web server
Login Security	This layer deals with security and identification of the user so the web client can determine privileges
User Information Page	This module deals with the personal information of users
Views and Pages	This module deals with the various views the user will access in order to get information
Server Communication	This layer deals is called from the view and is needed to communicate and request information from the database
Server Side Connection Service	This module receives the connection from the server communication layer.
User Permission	This module identifies what access the user has to the information stored in the database
Data Access	This layer is the location of stored information and will return to the previous modules and layers it was called from.