Course Management System

Aryan Desai (100707721)25%, Tanzir Hossain (100709113)25%, Tiwaloluwa Ojo (100700622)25%, Kalid Ajibade (100660188)25%

Date: November 29, 2020

Table of Contents

Part 1: Use cases, constraints, and quality attributes

I. Use Case Model

Use case	Description
STUDENT	
UC-1:Class Information	The system should provide the student with the means to see basic information on the Teacher and other classmates such as email, and who is with a class. Private information will stay hidden to other users
UC-2:Course Logs	The system will be able to show the user previously attended courses and enable the student access to contents of the course over the years
UC-3: Subscription Tracker	The system will be able to allow the students to subscribe for courses and exams, and also be able to keep track of the subscribers
UC-4:Collaboration Environment	The student should be able to upload and download various file types from the system, and also share with others all within a sectioned off team environment. This is possible with a grouping system.
UC-5:File Database	The system should enable students to upload and download various file types from the server.
UC-6:Mailbox	The system provides the students a means of communication this enables them send files, group invitations or just ask questions
UC-7:Search Bar	The system will offer a keyword search bar that works with static information and dynamic information
UC-8:Settings	The system should have an interface that enables the user to change various things. The user will be able do different things in regard to their password such as change and rest. Settings will also allow the user to change their

	notification behaviour. And visibility settings where the user will determine what others see on their profile
UC-9: Grade Interface	Gives the user access to see information regarding their grade in each course. It will offer things such as overall average, class average, teacher comments and more. The students grade are protected and only the student and instructor can view them
LECTURERS	
UC-10: Creating a Course	The system will allow lecturers to create a course. Either from scratch or from a previous course/period by duplicating and importing the materials required. They will also be able to establish enrolment policies based on the department, grade and first-come first-serve (fcfs). The lecturers will also be able to limit the number of students in a class and prevent unqualified students from subscribing.
UC-11: Assistant Lecturers	The system will allow the lecturers to register assistant lecturers and assign them to teams
UC-12: Upload material	The lecturers can upload course material for the course and prepare schedules
UC-13: Manage grades and grading policy	Lecturers need to be able to insert, update and calculate final grades within the system in accordance with the courses grading policy. Grading statistics are required.
UC-14: Sending and receiving mail	Lecturers need to send custom mail to multiple recipients (students) at once and have the support of mail merge templates. This feature should be a single click function
UC-15: Manage Students and Student Teams	Lecturers will be granted access to student profiles and plan meetings with them. They can also manage student teams (create, insert, remove, delete, rename) and enter their grades
UC-16: Manage Class	They need to manage static and dynamic course information, as well as post news messages and manage the course archives (visibility).

UC-17: Visibility	The system will grant visibility of grades and course materials to the right actors in the system. (i.e. Lecturers can view all grades, while students can view only theirs)
UC-18: Importability	Importing BOZ roster info in the course roster is needed for the lecturers
SYSTEM MAINTAINER	
UC-19:Creating Back-Ups of System	System will create backups of system in cases where either the system has faults or the system is down temporarily
UC-20:allow for size limitations of uploaded files from lecturers and students	Size limitations for uploaded files will ease the system load as if the files were too large then the system would very quickly run out of size and also files of huge size uploaded at the same time will cause heavy workload
UC-21:Maintain a limit for total space available for each course	System limits for each course allow it so that courses don't have an imbalance of size and encourages for organization of its file uploading system
UC-22:Allowing restoration of partial and complete back ups of specific date	Used in case of the system being in very heavy load, bugs and errors or if the system goes down. The system will be able to iterate to a previous version for temporary usage until the updated iteration comes through
ADMIN	
UC-23:Creating classes for the new school term	Create new classes for the new term and store them in the database
UC-24: Add new lecturer	Add a new lecturer to the specified course and store in database
UC-25:Limit course enrollment	Set a minimum limit to the amount of students that can enroll in a specific course

UC-26: Get student information	Get the students full information from a specified course and display it on screen.
UC-27: Add prerequisites of a course	The system will add a prerequisite to a specified course given their department, so the next time a student adds a course the system will check if the prerequisites have been completed.
UC-28: Calculate grade statistics of school	The system will take the grade statistics from each student, department, year and course and create a grade statistic that represents the entire school.
UC-29: Manually enroll a student	The system will add a student to a specified course, even if it violates the requirements or enrolment policies
UC-30 :Calculate the number of student who have failed this semester	The system will find every user that has passed and remove them from the array to leave behind the failed ones and display them on screen.
UC-31 :Detect Faults in system	The system will detect faults in the system if there are any.

II. Quality Attributes

ID	Quality Attribute	Scenario	Associated use case	Priority
STUDENT				
QA-1	Availability	The system should be watched regularly to make sure it is operating fine, in the case of an issue downtimes will be announced and the maintenance should take 4 hours. Downtime will be set for low traffic times if possible .	(UC-1) (UC-2) (UC-4) (UC-9)	H, H
QA-2	User Friendliness	As the user enters the system they will need there student login that works for all other services, Then they will be greeted with a simple UI that displays courses and important information, the user will also have the option to change the language the want to view, and there will be descriptive text when the user hovers the mouse over something	(UC-1) (UC-4) (UC-5) (UC-8) (UC-9)	H, M

QA-3	Accessibility	The system will be accessible through many devices, and will have accommodations for disabled users, giving them an experience catered to their needs	(UC-1) (UC-8)	H, H
QA-4	Security	All info is encrypted from hackers and those without direct access, students will also have the ability to change information in the case of a mistake or address change.	(UC-1) (UC-2) (UC-5) (UC-6)	H, H
QA-5	Interoperability	This system will enable the user to download their schedule in csv format. This will allow them to export their calendar and use it on a personal calendar platform.	(UC-1) (UC-2) (UC-3)	H, L
LECTURERS				
QA-6	Security	Several aspects of the system are only visible toward the lecturers. The lecturers may set content visibility on course info between the lecturers and students. Course enrolment is only acceptable with students who meet enrolment policy parameters. Only approved lecturers can view their students profiles and write to their grades.	(UC-10) (UC-13) (UC-15) (UC-16) (UC-17)	M, H
SYSTEM MAINTAINER				
QA-7	Scalability	System should be able to handle file handling when under very heavy workload like when whole class is accessing the system	(UC-2)	H, H
QA-8	Maintainability	Need to be able to update components of system in order to fix errors or add onto system functionality	(UC-4), (UC-1)	M, H
QA-9	Interoperability	The system should be able to limit sizes available for when oncoming assignments are being submitted for a class	(UC-3)	L, M
ADMIN				
QA-10	Performance	Several network devices send traps to the management system peak load. 100% of the traps are successfully processed and stored.	(UC-2)	H, H
QA-11	Security	Only the administrator should be allowed to do the following: - update , delete or manage courses - Specify prerequisites - Appoint lecturers to courses - Specify the minimum amount of students in a course	(UC-3) (UC-4) (UC-5) (UC-6)	H, H
QA-12	Maintainability	Needs to detect a fault in the network if it exists	(UC-1)	H, H

III. Constraints

ID	Constraints
STUDENT	
CON-1	Students must meet certain course grade to advance
CON-2	Student should meet certain requirements to be enrolled into the system
CON-3	Students must comply with course enrollment policies.
CON-4	Students and lecturers cannot access secondary university systems
CON-5	Only students can change study information of others
CON-6	Students can only view their own grades
LECTURER	
CON-7	Web accessible.
CON-8	Importing of outside data (rooster)
CON-9	Only lecturers can manage students grades
CON-10	Lecturers can manage and create student teams
SYSTEM MAINTAINER	
CON-11	Partial and Full Events from last 30 days must be stored for emergency use or debugging
CON-12	Files of up to 50 mb must be supported and System must be able to handle uploads and downloads of files
CON-13	Must set have a space limit for each course uploaded
ADMIN	
CON-14	A Minimum Of 50 Simultaneous Users Must Be Supported.
CON-15	Only administrators can update delete or manage courses and appoint lecturers to them
CON-16	Only administrators can specify the minimum amount of students for a course
CON-17	Only administrators can specify prerequisites including the completion of them

CON-18	No students can change information in the secondary university systems.
CON-19	The system must be able to create views for each stakeholder where some of the attributes have been restricted for use.

IV. Concerns

ID	Concern
CRN-1	Effective communication to teacher and students
CRN-2	Stable website
CRN-3	Effective presentation of course materials
CRN-4	Establishing an overall initial system structure that can handle uploading and downloading
CRN-5	Established System can handle heavy load from multiple sources
CRN-6	Established System can be restored to previous iterations for emergency use or debugging
CRN-7	Leverage teams knowledge of OOP principles and patterns, as well as Web Frameworks (Angular.Js, React, Django...).
CRN-8	Allocate work to the members of the development team

Part 2: ADD Process

I. ADD Iteration 1:Reference Architecture and Overall System Structure

A. Step 1: Reviewing Inputs

<table>
<tr><td>Category</td><td>Details</td></tr>
<tr><td>Design purpose</td><td>This is a greenfield system of a mature domain. The design purpose is to create a university management system that upholds stakeholder requirements and upholds abstraction among them.</td></tr>
<tr><td>Primary Functional Requirements</td><td>UC-5,19,20,21,22,31</td></tr>
<tr><td>Quality Attribute Scenarios</td><td>
<table>
<tr><td>Scenario Id</td><td>Importance to Customer</td><td>Difficulty of implementation according to architect</td></tr>
<tr><td>QA-2</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-3</td><td>High</td><td>High</td></tr>
<tr><td>QA-6</td><td>Medium</td><td>High</td></tr>
<tr><td>QA-8</td><td>Medium</td><td>Medium</td></tr>
</table>
</td></tr>
<tr><td>Constraints</td><td>All of the constraints discussed before are included as drivers</td></tr>
<tr><td>Architectural Concerns</td><td>All of the architectural concerns discussed previously are included as drivers</td></tr>
</table>

B. Step 2: Establish iteration goal by selecting drivers

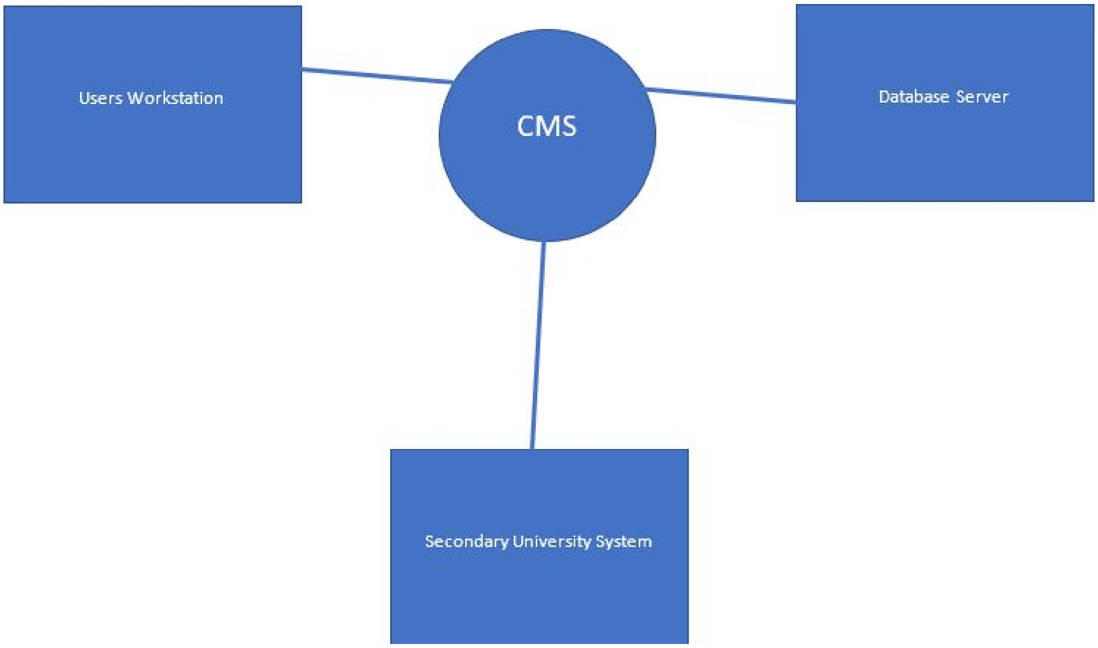
Iteration goal is to achieve CRN-1 (Establishing a overall initial system architecture) which would also be considered the first iteration of greenfield system

Inputs to be considered: all
Architect must be mindful of:

- QA-2, QA-3, QA-6, QA-8, CON- 7, CON-8, CON-12, CON-13, CON-19, CRN-4

C. Step 3: Choose one or more elements of the system to refine

This project would be considered a greenfield development effort, in which the element to refine is the FCAPS system and the refinement phase uses decomposition.



D. Step 4: Choose one or more design concepts that satisfy the selected drivers

Reference Architectures	
Design decisions & location	Rationale
Rich Internet Applications	The Rich Internet Application allows a rich user interface within a browser that can also be simultaneously accessed by multiple users at the same time. It also supports the web capabilities specified in QA-2, while dissolving possible issues with CON-7. This will also be helpful in achieving deployment and updates without changing the users system in QA-1
ALTERNATIVES	
Alternative	Reason for discarding
Web Application	This system does not provide a rich user interface which is needed in QA-2 to provide a simple and useful UI.
Rich Client Application	This is oriented toward developing the application in a web browser isolated on the clients system and would cause a conflict with QA-1.
Mobile Application	A mobile application would only be applicable to mobile client users and the systems need to delve more into a desktop/laptop applications on the web

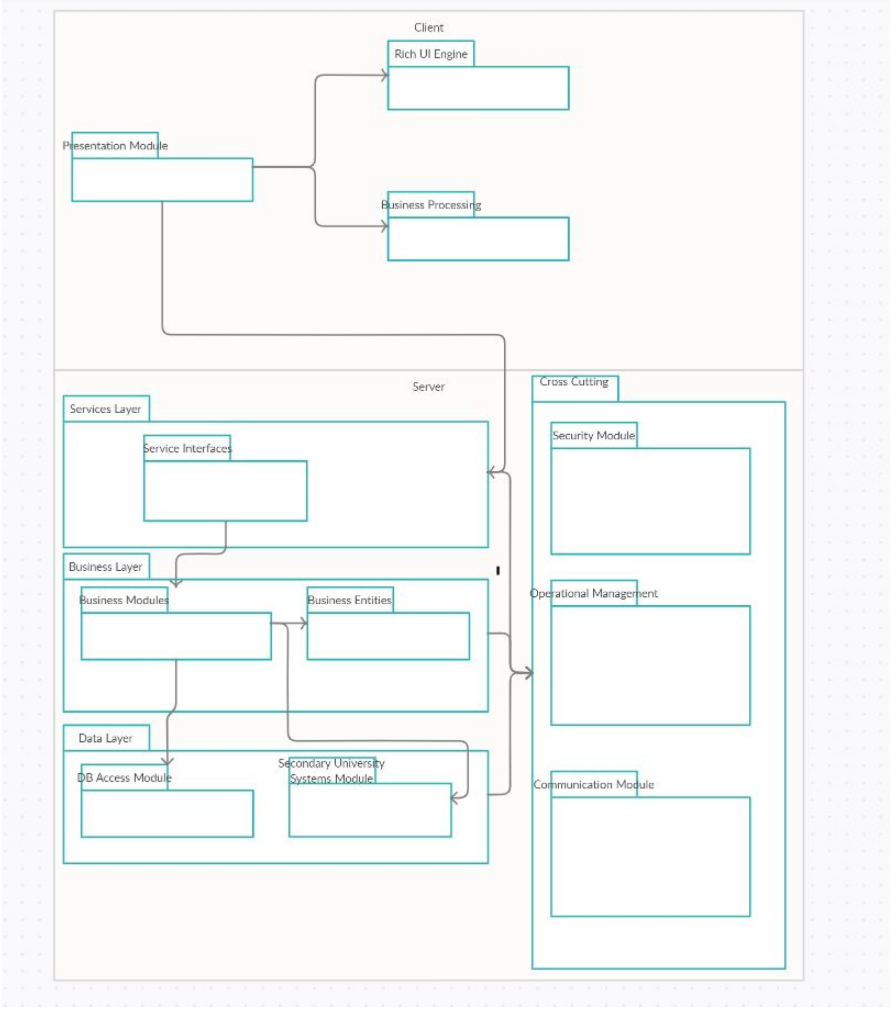
Deployment Patterns	
Four-Tier Deployment	The web server and application server are deployed in different tiers improving security.
User Interface	
Build the user interface of the internet application using javascript, HTML, XML and other technologies	The rich internet application framework supports the use of javascript and XML which supports QA-2's requirements for web capabilities.
Deploy the application using asynchronous Javascript and XML	Access to the application is obtained by going to the specified url in a web browser (CON-7)

E. Step 5: Instantiate arch. elements, allocate responsibilities and define interfaces

Design Decision and Location	Rationale
Add local data into the rich Internet application	Upon the user having to submit or edit a document within the system they would likely require storage to read and write from. Communication and Processing is performed on the client side along with the Rich UI engine. While retrieving and sending data is performed on the data layer
Create a Module dedicated to the secondary university systems server in the data layer	Creating this module in the data layer allows for abstraction of the access to the secondary university systems which supports CON-4 and CON-18.

F. Step 6: Sketch views and revord design decisions

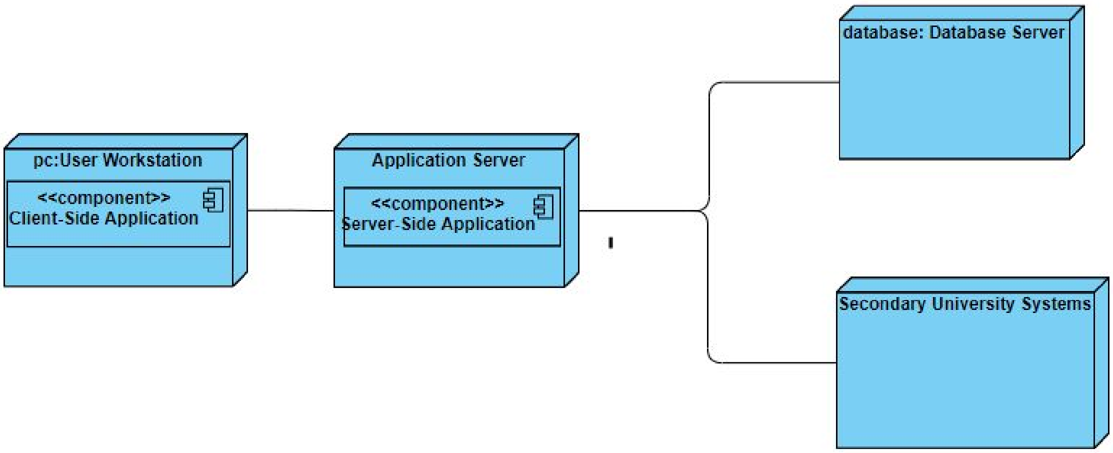
The following diagram shows the sketch of a module view of the reference architectures that were selected



Element	**Responsibility**
Presentation Module	This layer contains the modules that fetch data from the server and send it to the UI engine and business processing
Rich UI Engine	This is the module that uses javascript and XML to produce a rich User interface that is simple and aesthetically appealing
Business Processing	This module contains the logic for processing business information
Services Layer	This layer contains the modules that expose the services that are consumed by the client
Service Interfaces	This module has the functionality to expose services that are consumed by the user
Business Layer	This layer implements business logic operations that are performed on the server side
Business Modules	This module implements business operations that are performed on the server side

Business Entities	These entities make up the domain model
Data Layer	This layer contains modules that communicate with the database and secondary university systems
DB Access Module	This module contains functionalities that communicate with the database
Secondary University Systems Access Module	This module contains functionalities that communicate with the secondary sniversity systems.
Cross Cutting	This layer includes modules with functionality across layers such as security,logging and IO

The following deployment diagram sketches an allocation view that illustrates where the components associated with the modules in the previous diagram will be deployed.



The responsibilities of the elements are displayed here:

Element	**Responsibility**
User Workstation	The users pc which hosts the rich user interface and business processing
Application Server	The server that hosts the logic of the application and communicating with the database server and secondary university systems.
Database Server	The server that hosts the legacy relational database
Secondary University Systems	The external university systems that are monitored by the administrator.

The relationship between the elements is summarized in the following table:

Relationship	**Description**
Between app server and database server	Communication with the database will be done with standard HTTP protocol
Between app server and secondary university systems	Communication will be done with SNMP protocol

G. Step 7: Perform analysis of current design and review iteration

The following table summarize the design process using the Kanban board technique

Not Addressed	Partially Addressed	Completed Addressed	Design decisions made during the iteration
	UC-5		Selected reference architecture establishes the modules that will support this functionality
	UC-19		Selected reference architecture establishes the modules that will support this functionality
	UC-20		Selected reference architecture establishes the modules that will support this functionality
	UC-21		Selected reference architecture establishes the modules that will support this functionality
	UC-22		Selected reference architecture establishes the modules that will support this functionality
	UC-31		Selected reference architecture establishes the modules that will support this functionality
	QA-2		Selected reference architecture establishes the modules that will support this functionality
		QA-3	The selected architecture allows for accessibility through web browsers
	QA-6		Selected reference architecture establishes the modules that will support this functionality
	QA-8		Selected reference architecture establishes the modules that will support this functionality

CON-1-6			No decisions made
		CON-7	Selected reference architecture establishes the modules that will support this functionality
CON-8-11			No decisions made
	CON-12		Selected reference architecture establishes the modules that will support this functionality
CON-13			No decisions made
	CON-14		Selected reference architecture establishes the modules that will support this functionality
CON-15-19			No decisions made
CRN-1			No decisions made
	CRN-2		The design decisions have been established using the architecture modules
CRN-3			No decisions made
	CRN-4,5		Selected reference architecture establishes the modules that will support this functionality
	CRN-6		Utilizing the 4-tier deployment pattern we can rollback changes for emergency use
		CRN-7,8	Since the application is web oriented, it can developed using a variety of web frameworks

II. ADD Iteration 2: Identifying structures to support primary functionality

A. Step 2 : Establish Iteration Goal by Selecting Drivers (Primary Usecases)

The goal of this iteration is to address the general architecture concern of identifying structures to support primary functionality.
Primary use cases: UC-5,19,20,21,22,31

B. Step 3:Choose One or More Elements of the System to Refine

1. In the previous iteration the element that was the entire university system. The refinement process was to decompose the system to different elements (User Workstation, Database Server, and Secondary University System.) In order to

refine the elements of the system again, we would need to refine the Database server, user workstation and secondary university by decomposing it further.

C. Step 4:Choose One or More Design Concepts That Satisfy the Selected Drivers

Design decisions and location	Relation and Assumptions
Create a Domain Model	Before decomposition identify the major entities and their relationships
Identify Domain Objects	Each element of the application needs encapsulation in a self-contained block (i.e. a domain object)
Decompose Domain Objects into components	Domain objects are sets of functionality supported by elements within the layers. (i.e. the components and modules)
Use Web framework such as Django	Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. This framework was selected over the alternative spring framework due to the versatile and efficient nature of the framework.

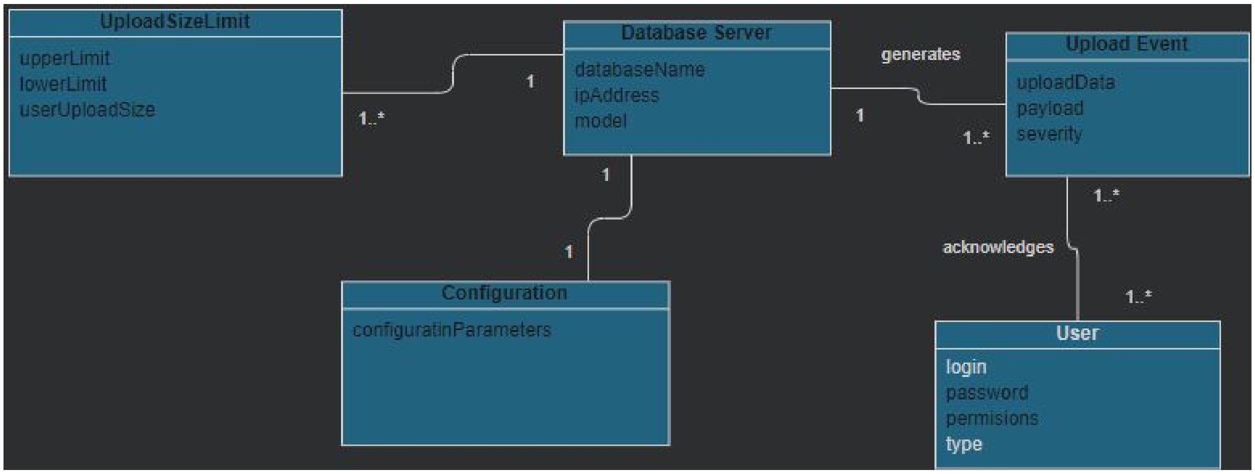
D. Step 5 : Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

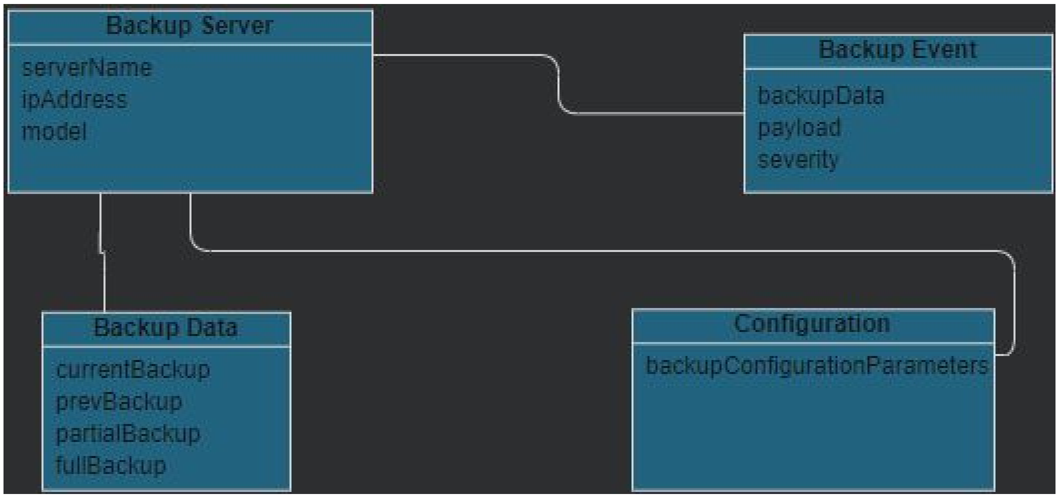
Design decisions and location	Rationale
Create a initial Domain Model	Initial domain model is created to help speed up the phase of design and helps model and identify the entities that identify the main use cases
Map use cases to domain objects	Instantiation of domain objects are created through analysis of the use cases. To address CRN-8
Decompose Domain Objects across layers to create layer specific modules with an explicit interface	-This techniques serves to ensure that the modules that support the functionalities are identified -We do this for the primary use cases so that we can also address CRN-8 in which we will make other team members address the rest of the modules -CRN-9: finding a way to reliably back up the server
Connect component associated with modules using Django	This framework gives the ability to support both the database and backup server

Element	Responsibility
NetworkStatusView	Displays the network representation and updates it when events are received.
RequestManager	Responsible with communication with server side logic
RequestService	Provides a facade that receives requests from the client
UploadController	Contains business logic related to the upload information
DomainEntities	Contains entities from the domain model
BackupEventController	Contains business logic related to the management of backup type events
DataCollectionController	Contains logic to perform data collection and storage of backups
FileDataController	Contains logic related to the upload file storage on the system
BackupDataMapper	Responsible for mapping backup data to a location in the server
EventDataMapper	Responsible for mapping the Event data to a location on the server
BackupServerConnector	Responsible for communication with the backup server.

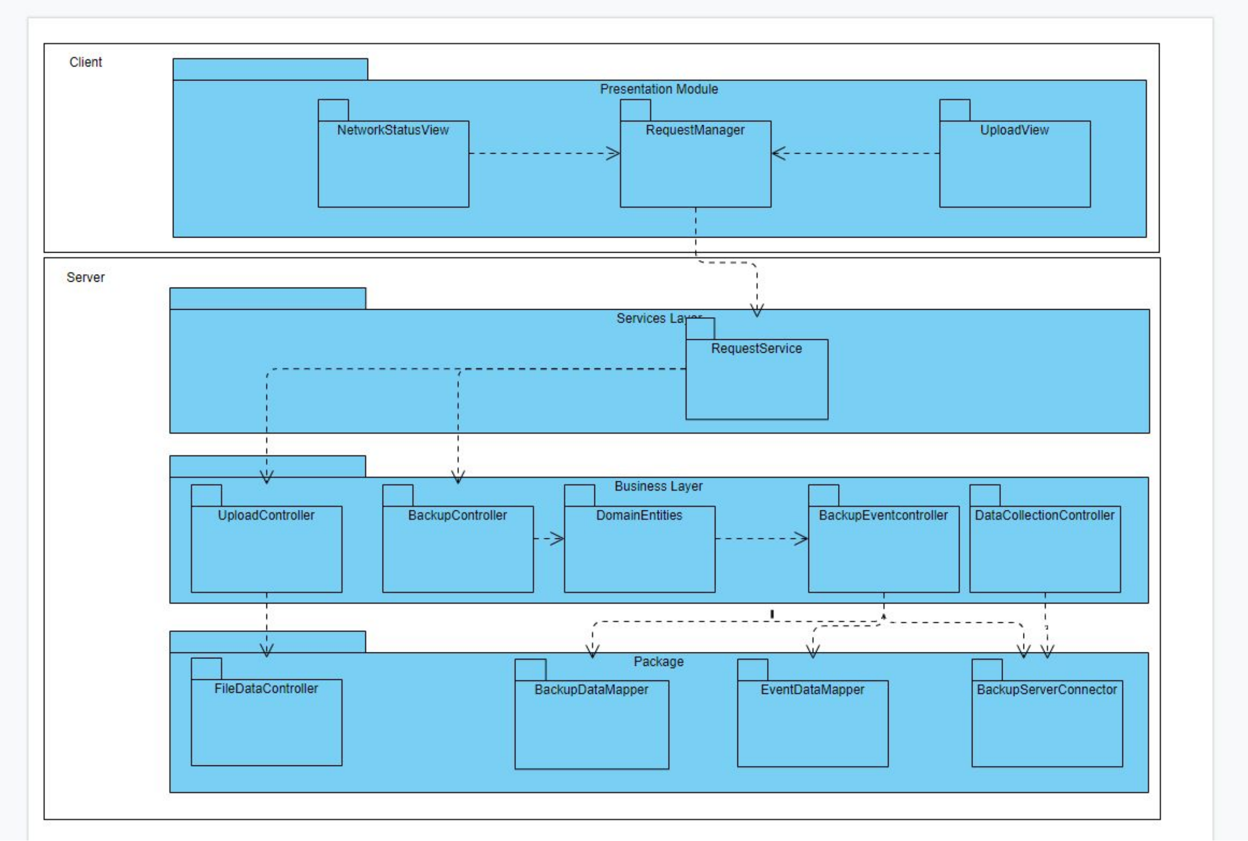
E. Step 6 : Sketch views and record design decisions

Domain Diagrams:

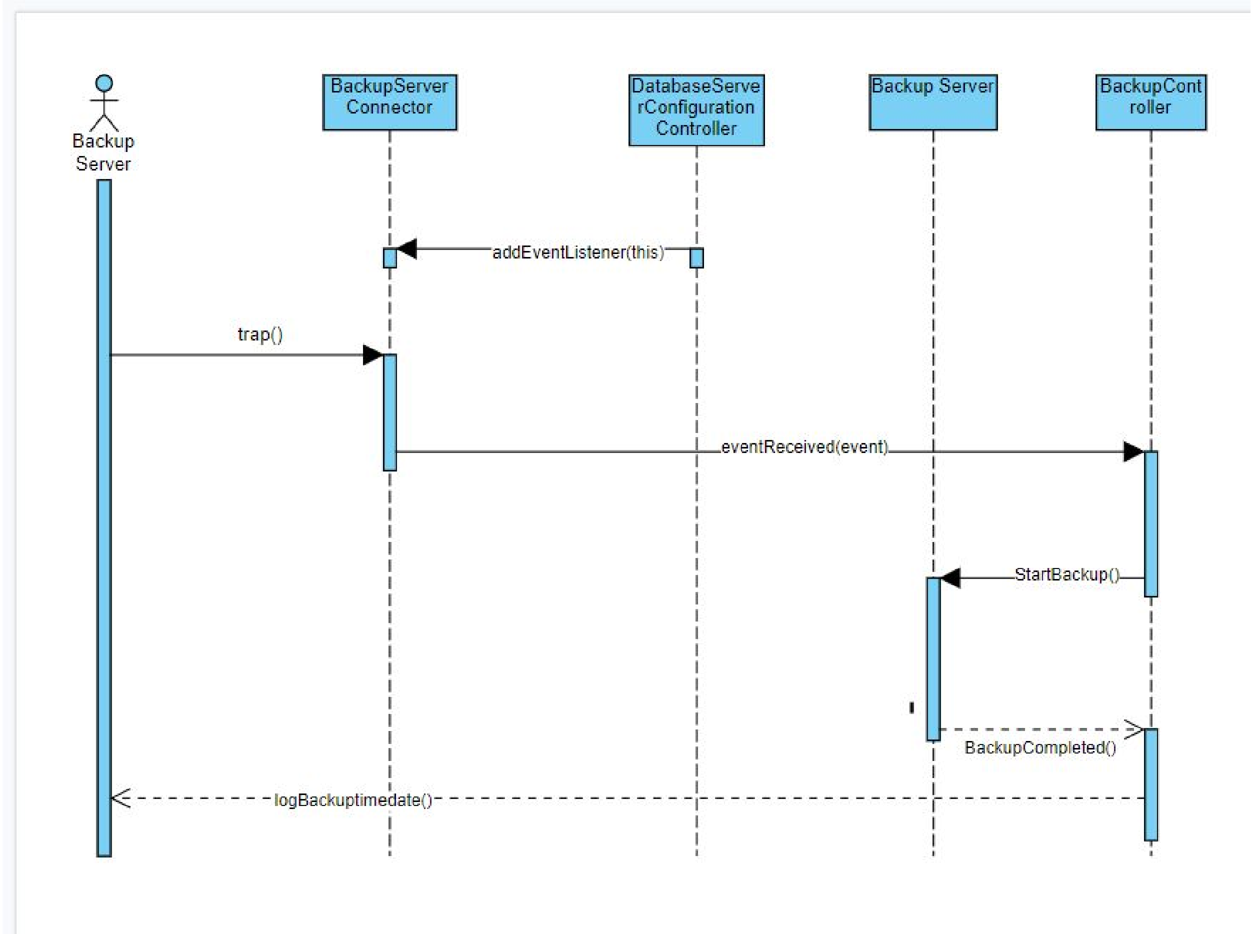




Modules that support primary use cases:

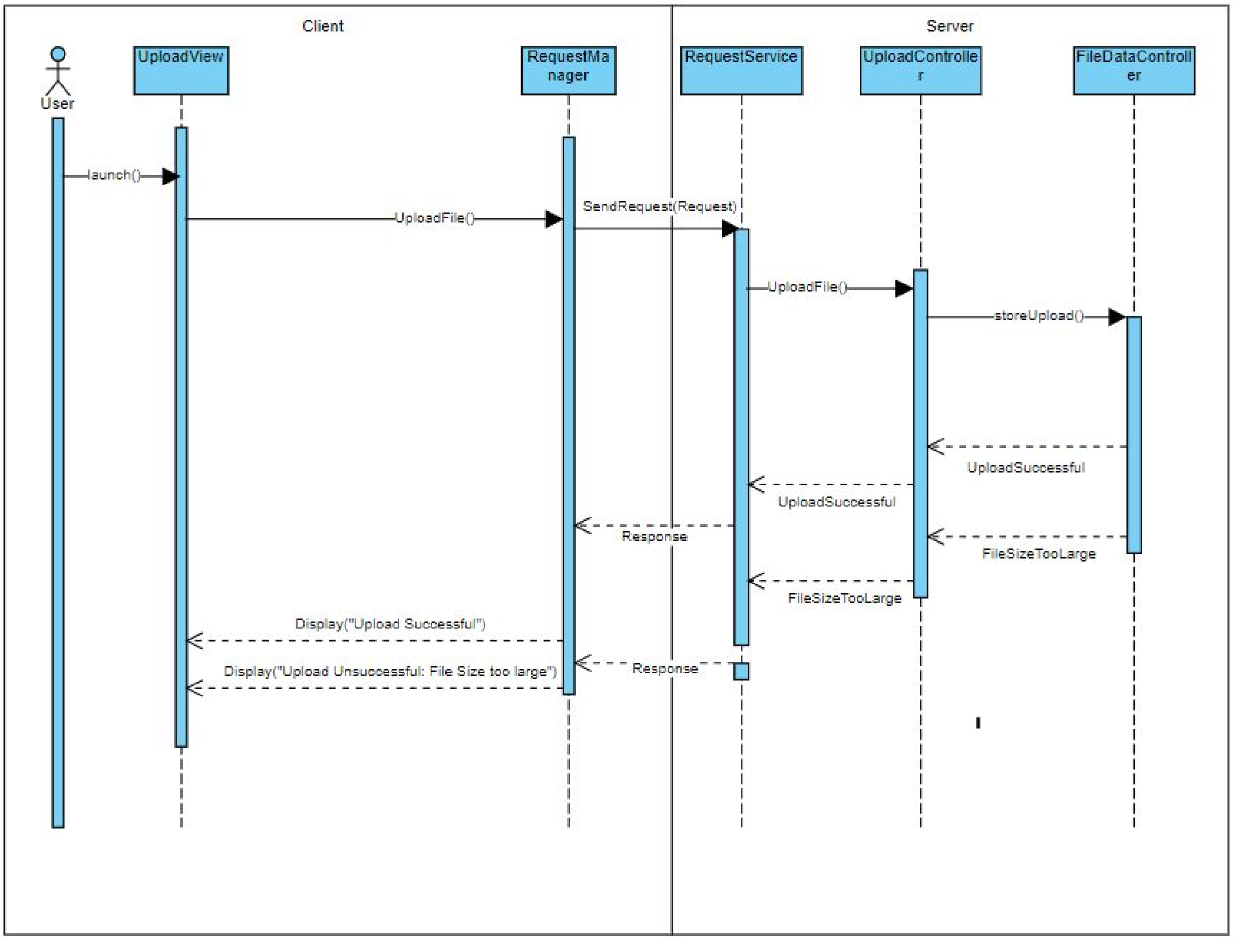


Sequence diagram for UC-19 and UC-313



Method Name	Description
boolean addEventListener(EventListener el)	This method allows components from the business logic to register themselves as listeners to events that received from the database server
Boolean eventReceived(Event evt)	This callback method notifies the clients that a new event has occurred
Boolean StartBackup()	This method starts backing up the system
Boolean backupCompleted()	This method sends a boolean value which indicates weather the backup process has been completed.
String logBackuptimedate()	This method saves the time it took to backup the system and displays it on the screen.

Sequence diagram for UC-20 and UC-21



Method Name0	Description
Boolean initialize()	Opens up the network representation so that users can interact with it
File uploadFile()	Requests server to upload a file
File storeUpload()	Stores the file on the database server

F. Step 7 : Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

Note:check concerns and constraints

Not Addressed	Partially Addressed	Completed Addressed	Design decisions made during the iteration
		UC-5	Modules across layers and main interfaces to support this use case was identified
		UC19-22	Modules across layers and main interfaces to support this use case was identified
		UC 31	Modules across layers and main interfaces to support this use case was identified
	QA-2,QA-4,QA-11		Elements thats support uc-5 has been identified
		QA-3	No relevant decisions made
	QA-6		No relevant decisions made
	QA-8		No relevant decisions made
CON-1-6			No relevant decisions made
CON-8-10			No relevant decisions made
	CON-11		Modules responsible for collecting data has been identified
	CON-12		Modules responsible for collecting data has been identified
	CON-13		Modules responsible for collecting data has been identified
	CON-14		Modules responsible for collecting data has been

			identified
CON-15-19			No relevant decisions made
CRN-1			No relevant decisions made
	CRN-2		No relevant decisions made
CRN-3			No relevant decisions made
	CRN-4		Modules responsible for the upload have been addressed
	CRN-5		Selected reference architecture establishes the modules that will some support this functionality
		CRN-6	Elements that support CRN-6 has been identified with Backup server module
		CRN-8	Modules responsible for collecting data has been identified

III. ADD Iteration 3: Addressing Quality Attribute Scenario Driver (QA-11)

A. Step 2: Establish Iteration Goal by Selecting Drivers

For this iteration, the architect will focus on the QA-11 quality attribute scenario: The Admin wants to access the university system settings to manage courses and appoint lectures to courses. The system is secure and only he is able to gain access to the system settings with his credentials.

B. Step 3: Choose One or More Elements of the System to Refine

For this security scenario, the elements that will be refined are the physical nodes that were identified during the first iteration:

- Business Layer
- Modules
- Entities

C. Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers

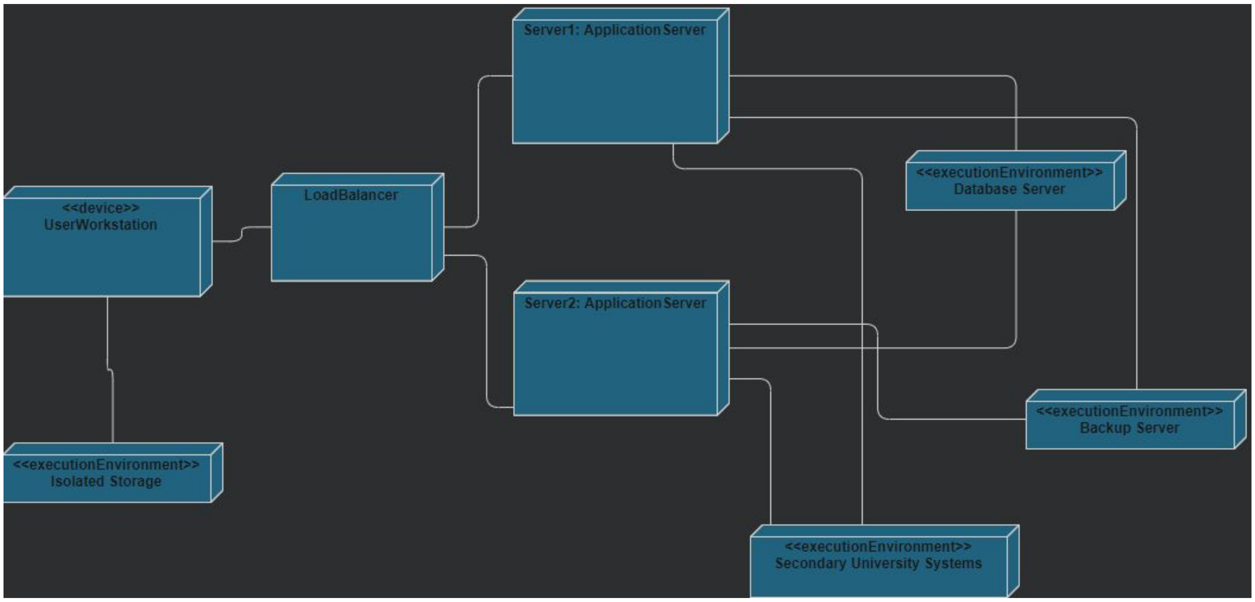
Design Decision	Rationale
Introduce Revoke Access tactic by limiting access to sensitive	By limiting access to sensitive resources even for legitimate users, we can resolve suspected attacks

resources	
Introduce Half-Sync/Half-Async to provide a faster experience for the user	By developing in in a concurrent software we can provide parallel processing for the user without sacrificing runtime efficiency

D. Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

Design Decision	Rationale
Revoke access to sensitive resources on the application server	Revoking access on the application server would prevent resources from being fetched and prevent suspected attacks
Perform concurrent programming on the application framework	Performing concurrent programming on the application framework will distribute load across the application clusters

E. Step 6: Sketch Views and Record Design Decisions



Element	Responsibility
LoadBalancer	Dispatches client responsibility toward the Application Servers

F. Step 7: Perform analysis of current design & review iteration goal & achievement of design purpose

Ignore UC5,19-22,31 QA3, CON 4,5,7,8 (this is a personal note it will be deleted)

Not Addressed	Partially Addressed	Completed Addressed	Design decisions made during the iteration
QA-1/QA-5			No relevant decisions made
QA-2			No relevant decisions made
		QA-6	Leaturer views and content visibility has beens discussed during this iteration certain criteria must be met before the student can be given access to the system
QA-(7-9)			No relevant decisions made
		QA-11	Admin privileges have been explicitly defined with our system and have been further protected during this iteration
CON-1			No relevant decisions made
CON-(2, 3)			No relevant decisions made
	CON-(9-10)		No relevant decisions made
CON-(11-13)			No relevant decisions made
CON-14			No relevant decisions made
		CON-(15-19)	Admin privileges have been explicitly defined with our system and have been further protected during this iteration
CRN-1			No relevant decisions made
	CRN-2		Within this iteration by implementing the revoke access tactic, protecting the system form potential threats
CRN-3			No relevant decisions made
		CRN-5	The load balancer shares work amongst members allowing for a high workload