SOFE3650 Fall 2020
Software Design and Architecture
2020-11-29

Prepared for:
Professor Kalid Ahmed

Prepared by:
Michelle cheng 100696572 - 33% contribution
Evans Mosomi 100719552 - 33% contribution
Martins Babajide 100709716 - 33% contribution

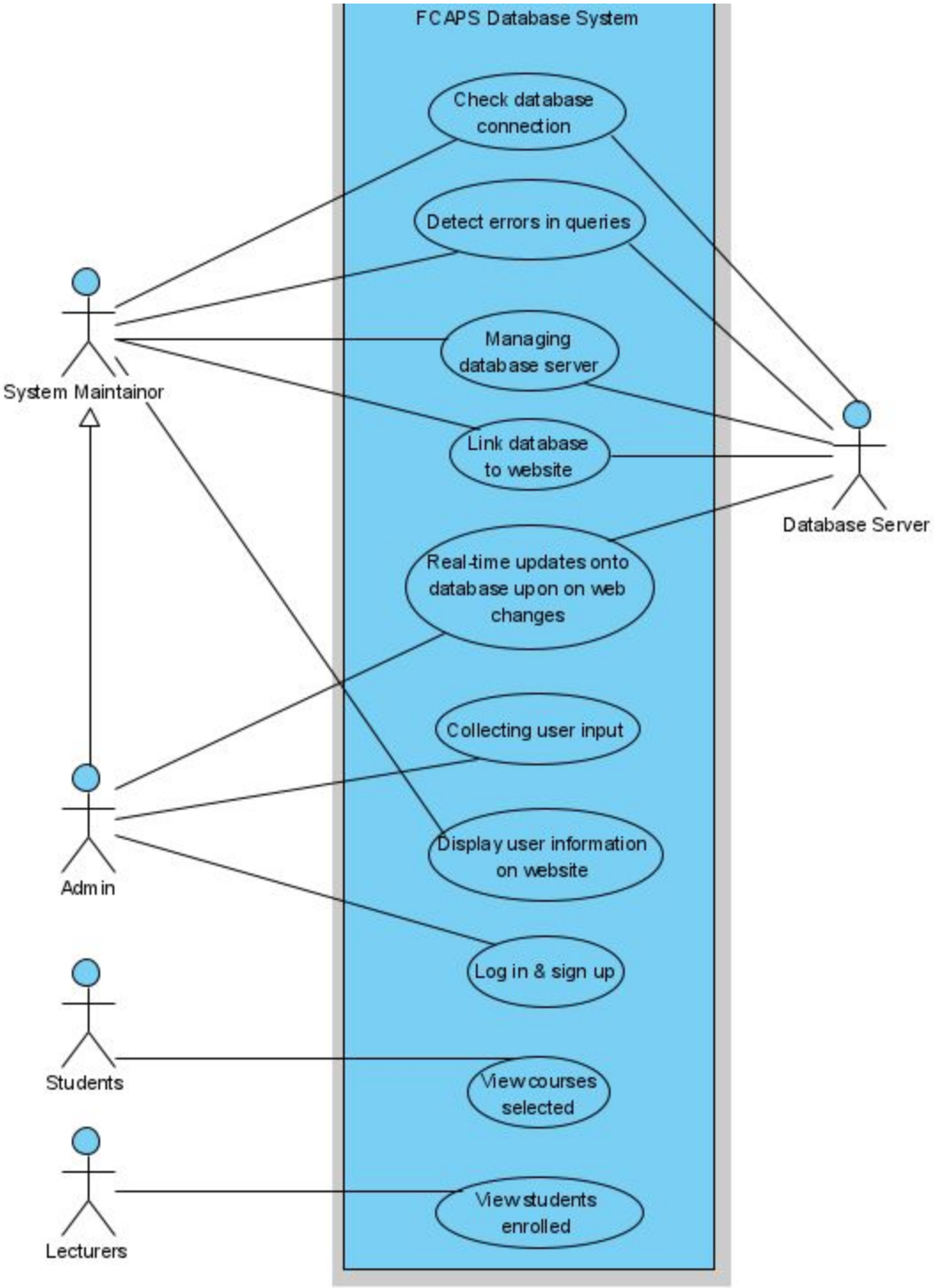
Objectives

The objective of this project is to demonstrate the method of steps in the design of software architecture for a set of requirements given by the course instructor. The design approach is to take the Attribute Driven Design (ADD) presented in the textbook and apply to the case given. The ADD process takes requirements as inputs such as functional, quality and constraints. It expects containers such as responsibilities, interactions and information flow to be the output of the process. This ultimately provides the architecture design and description of the system. It is 7 steps in total and consists of 2-3 iterations as seen later in this report.

Deliverable 1- Use Cases, Constraints, and Quality Attributes

Use Cases

UC 1	Check database connection	System Maintainer checks database connection in the system. Resolves issues that occur due to connectivity.
UC 2	Detect errors in queries	System maintainers detect errors in queries and ensure the database is fetched from the database correctly.
UC 3	Manage database server	System maintainer manages the database server and ensures efficient use of the database.
UC 4	Link database to website	System maintainers link the database to the website to allow for user requests to be made.
UC 5	Real-time update to database from web page	Admin ensures real-time updates from staff and lecturers are made into the database as web requests are made.
UC 6	Collect User input	Admin collects user input from students and lecturers.
UC 7	Display user information on website	System maintainer displays user information on the website as needed.
UC 8	Login and sign up	Admin manages the login and signup request as made by students and lecturer.
UC 9	View courses Selected	Students are able to view courses selected.
UC 10	View students enrolled	Lecturers can view the students enrolled in their courses.


FCAPS Database System
Check database connection
Detect errors in queries
Managing database server
Link database to website
Real-time updates onto database upon on web changes
Collecting user input
Display user information on website
Log in & sign up
View courses selected
View students enrolled
System Maintainor
Admin
Students
Lecturers
Database Server

Quality Attributes

ID	Quality Attribute	Scenario
QA-1	Availability	The system should be available 24/7 unless during backup hours: Sunday 12AM-5AM
QA-2	Usability	Lectures can add/change grades. All users can receive notifications when changes are made
QA-3	Interoperability	Website and database can interact upon every update to provide accurate and timely information. Users can send messages between themselves in the system.
QA-4	Security	Database system should be secure and Students can only view their own personal records and information and access to other student information should be denied.
QA-5	Maintainability	Lecturers and administrators should be able to maintain course assignment and information as needed. System maintainer also works to ensure that backup is created if needed.
QA-6	Scalability	Information for the database should be able to be adjusted as needed by the system.

Constraints

ID	Constraints
CON-1	Back-up maintenance can only be done between 12AM-5AM Sunday
CON-2	The website can be accessed through a WAMP server alone and on the website via localhost
CON-3	Students cannot access other student's records
CON-4	Only admins have access to pass/fail students and add/drop courses outside registration

Deliverable 2-Design of the CMS

ADD Iteration 1

Step 1: Review Inputs

Category	**Details**		
Design Purpose	This is a course management system from a mature domain. The purpose is to create a design for the construction of the system.		
Primary Functional Requirements	UC 1: As it directly supports the system being functional UC 2: As it directly supports the system being functional UC 5: As it directly supports the system being usable.		
Quality Attribute Scenarios	**Scenario ID**	**Importance to Customer**	**Difficulty of implementation**
	QA-1	High	Medium
	QA-2	High	Medium
	QA-2	Medium	High
	QA-4	High	Medium
	QA-5	Low	Medium
	QA-6	Low	Medium
	QA-1, QA-2. QA-3 and QA-4 are drivers.		
Constraints	All constraints CON-1, CON-2, CON-3 and CON-4 are constraints.		
Architectural Concerns	All of the architectural concerns are discussed.		

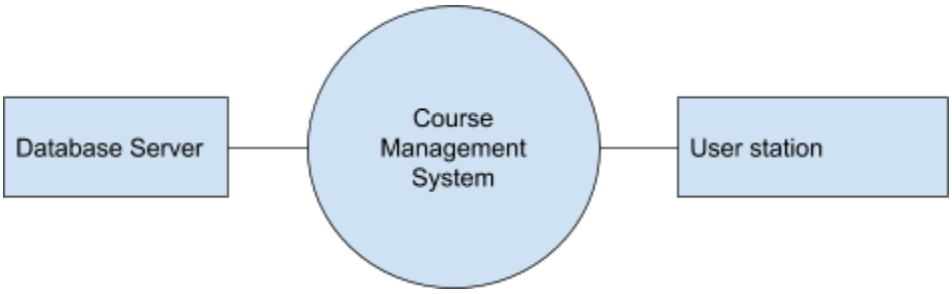
Step 2: Establish Iteration Goal by Selecting Drivers

- QA-1: Availability
- QA-2: Usability
- QA-3: Interoperability
- QA-4: Security
- CON-1: Back-up maintenance can only be done between 12AM-5AM Sunday
- CON-2: The website can be accessed through a WAMP server alone and on the website via localhost
- CON-3: Students cannot access other student's records

- CON-4:Only admins have access to pass/fail students and add/drop courses outside registration

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

This is a greenfield development effort, so in this case the element to define is the entire Course management System. Refinement is performed through decomposition.



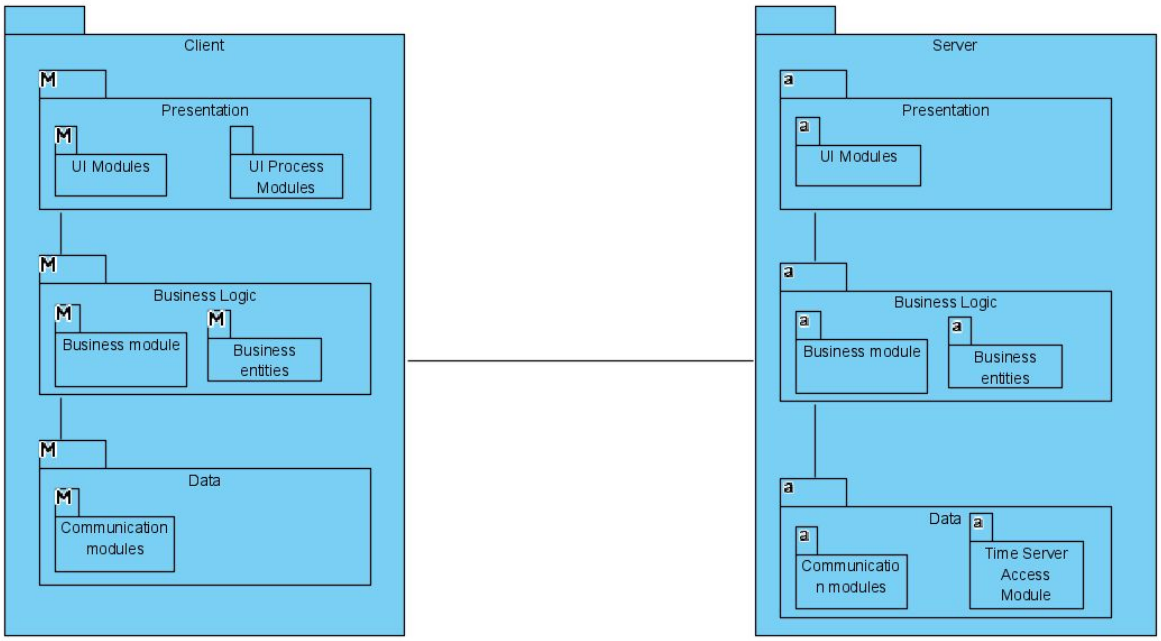
Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers

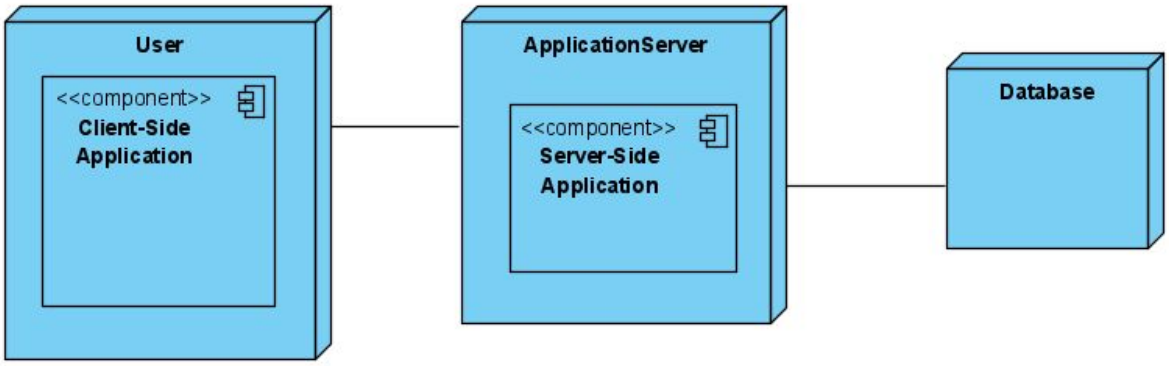
Design Decisions and Location	**Rationale**
Rich Internet Application	Supports development of applications with rich user interfaces in web browsers. Client side processing improves system performance and allows for efficiently to be kept even if complexity increases. Easy to update application which contributes to QA-2, QA-3, QA-4 and CON-4.
Mobile Application	This type of architecture is geared towards development on a handheld device. This would contribute towards QA-1, QA-2 and allow continuous access to the system by users. Therefore is included in the design architecture of the system.
Discarded Alternatives	**Rationale**
Service Application	Do not allow for user interface therefore does not fulfill the needs of this system. Services are oftentimes consumed by other applications and therefore is discarded.
Web Application	Development of applications that are accessed through a web browser. For a course management system this would not be efficient when it comes to updating data for the system or tracking of various data. Therefore it is discarded.
Rich Client Application	Supports development of applications on user's PC. For this system, it would not be ideal due to the complexity level. Can also cause complications to QA-4 when combining with other services.

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

Design Decision and Location	**Rationale**
Kept data storage on Server side	The Course Management System contains personal information about course attendants and this information needs to be kept secure and separate from the user interface systems.
Reduce heavy graphics for system	Save on run time and performance by reducing heavy graphics within application.
Local data storage	In the event of the servers breaking down, a secured database needs to keep all data in the system. This should be kept private at all times unless needed to be accessed due to a server issue
Graphical User Interface correspondence	The User interface should allow for all the front end needs of the system to be met. Full functioning regarding courses and other information related to it.

Step 6: Sketch Views and Record Design Decisions





Element	Responsibility
Browser (CS)	Client side has browser to access system
Data (CS)	Communication with the server to host browser and fetch results from database
Cross-cutting (CS)	Communication between layers improves security and logging between elements.
UI Modules (CS)	Allow for user to make request on user interface
UI process modules (SS)	Manages flow of use cases and how flow of application will be
Business logic (SS)	Process data from application and process on server side.
Business entities (SS)	Associated with the business logic.
Business Utilities (SS)	Manage and maintain related to data storage and retrieval.
DB access module (SS)	Allows business entities to access relational databases.
Communication Module(SS)	Manages communication between server and client to ensure all requests are made.

Responsibilities of elements

Element	Responsibility
Database server	Server that runs request and holds database
Application server	Host server side of application and the web pages associated
Web server	Manages interaction of user and the application
Mobile User Interface	Client side of the application with mobile business and data layers
PC User Interface	Client side of application

Additional relationships between elements

Relationship	**Description**
Web/Application server	HTTPS Protocol
Mobile/PC with Web server	PHP-Localhost
Database/Application server	MySQL

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

Not Addressed	**Partially Addressed**	**Completely Addressed**	**Design Decisions Made During the Iteration**
UC 1			No decisions made.
UC 2			No decisions made.
	UC 5		System accessible 24/7 by users and maintainers to allow for consistent real-time updates.
		QA -1	System accessible 24/7 even if server breaks down, backup database is to be stored.
	QA -2		Multiple user interfaces both web and mobile allows for easy usability by users.
QA-3			No decisions made.
		QA -4	Personal inter formation is to be stored on the server side only. Users are restricted according to rank in the system.
CON-1			No decisions made.
CON-2			No decisions made.
		CON-3	Security protocol, information restricted accordingly depending on user position in the system.
	CON-4		Security protocol for the system.

ADD Iteration 2

Step 1 : **Identifying structures to support primary functionality;** Decision making about units of implementation which affects team formation, interfaces and factors which affect distribution of development tasks come into place.

Step 2 : **Establishing Iteration Goal by Selecting Drivers:** The goal of this iteration is to address the general architecture concern of identifying structures to support primary functionality. Besides CRN-3, we would consider the following system's primary use cases:-
UC-1:Managing database connection
UC-2:Detect errors in queries
UC-3:Manage Database server
UC-6:Collect user input
UC-7:Display user information on website.

Step 3: **Choose one or more elements of the system to refine:** In this iteration, we place more emphasis on refining elements that are part of the different layers in the reference architecture systems mentioned in iteration 1 :(**Rich Internet Application + Mobile Application + Service Application).** The support of functionality of this university database system requires the use of these components to combine together.

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

Design decisions and locations	**Rationale and assumptions**
Create a Domain model for the application	Before starting a functional decomposition, it is necessary to create an initial domain, along with their relationships. A domain model is needed to be created for easy readability and maintenance.
Identify Domain Objects that map to functional requirements.	Each distinct functional element of the application needs to be encapsulated in a domain object.i.e The better way is not to consider domain objects but decompose layers into the modules directly but requirements would fail to be considered
Decompose Domain objects into general and specialized Components	Domain objects represent complete sets of functionality, but the functionality is supported by finer-grained elements located within the layers. We refer to the components as modules.

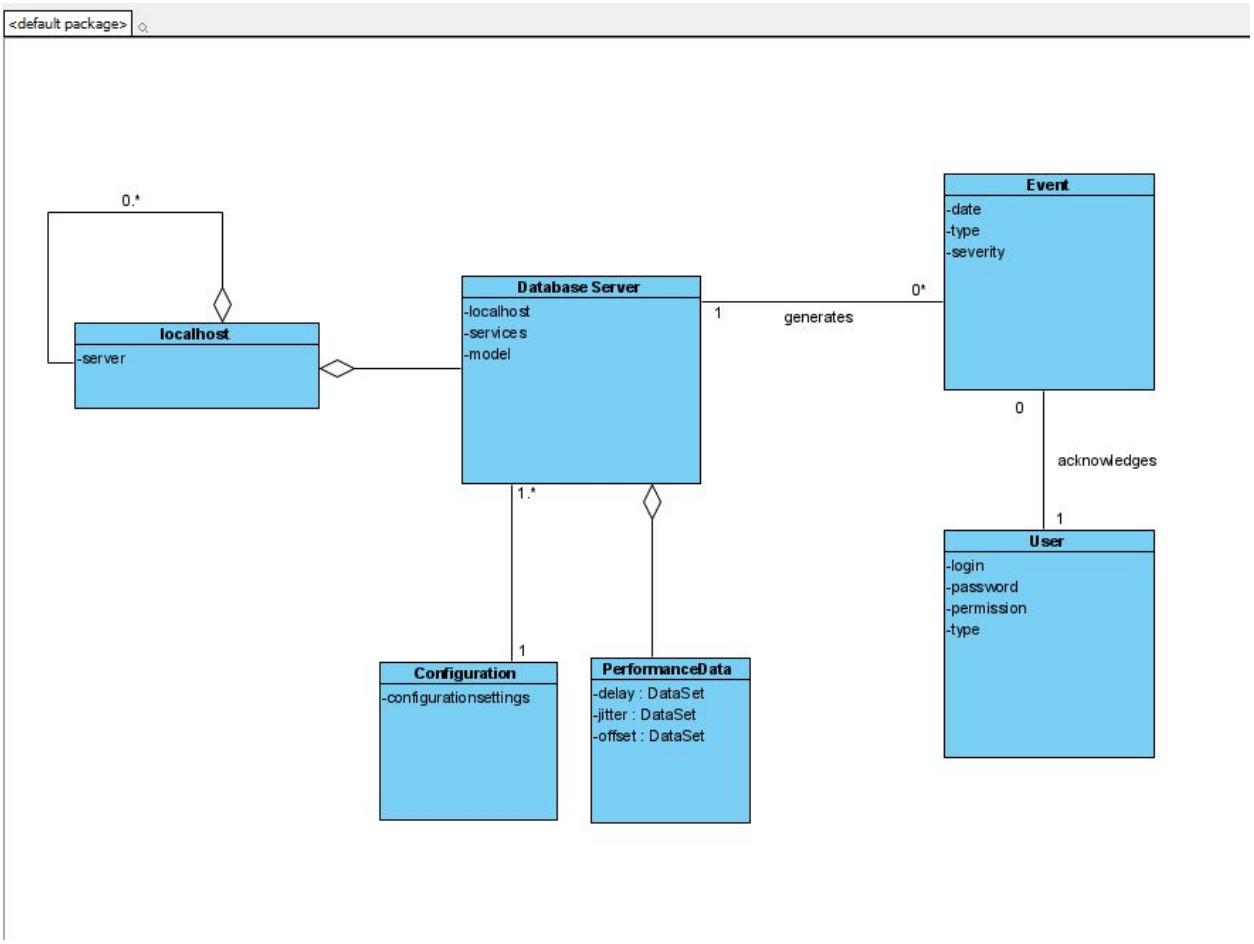
Use spring framework and hibernate	The architecture based on RIA to give the user an easy to use interface and the Json,php and database workbench used to provide accurate information when there is a request sent in by the user to the website.Spring is widely used to support enterprise application development.while, the hibernate is an object which integrates well with spring.

Step 5: **Instantiate Architectural Elements, Allocate Responsibilities and Define interfaces.**

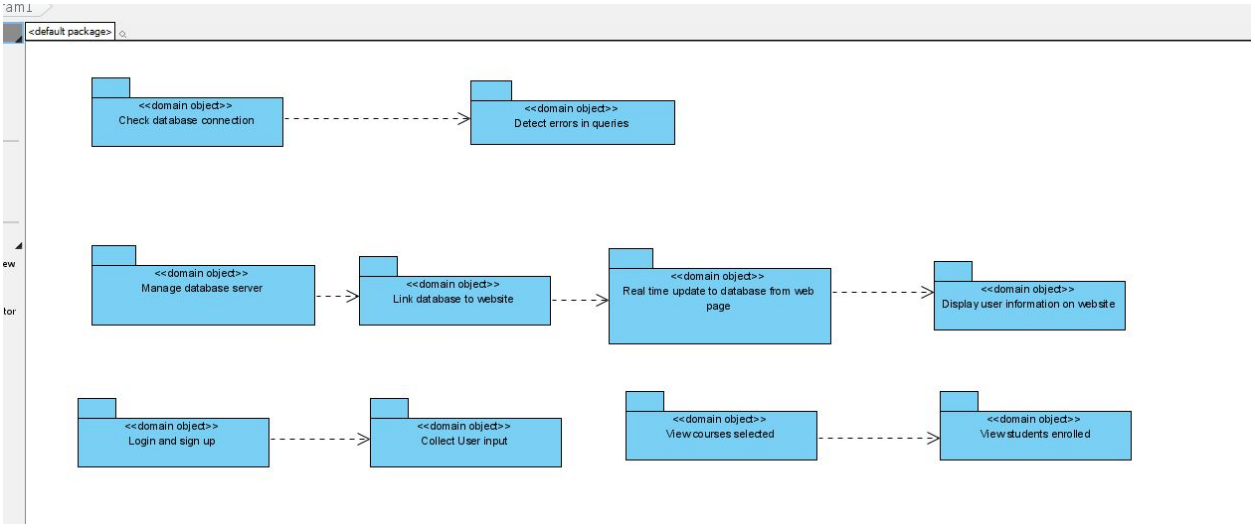
Design decisions and location	**Rationale**
Create only an initial domain model	The entities that participate in the primary use cases need to be identified and modeled but only an initial domain is created.
Map the system use cases to domain objects	Identical identification of domain objects can be made by the analysis of systems use cases.
Decompose the domain objects across the layers to identify layer specific modules with an explicit interface	This technique ensures that the modules that support all the functionalities are identified.
Connect components associated with modules using spring	This framework uses an inversion of control approach that allows different aspects to be supported and modules unit tested.
Associate frameworks with a model in the data layer	ORM mapping is encapsulated in the modules that are contained in the data layer. The hibernate framework which was previously selected is associated with these modules.

Step 6 : **Sketch Views and Record Design Decisions.**

View 1: Initial domain model for the system.

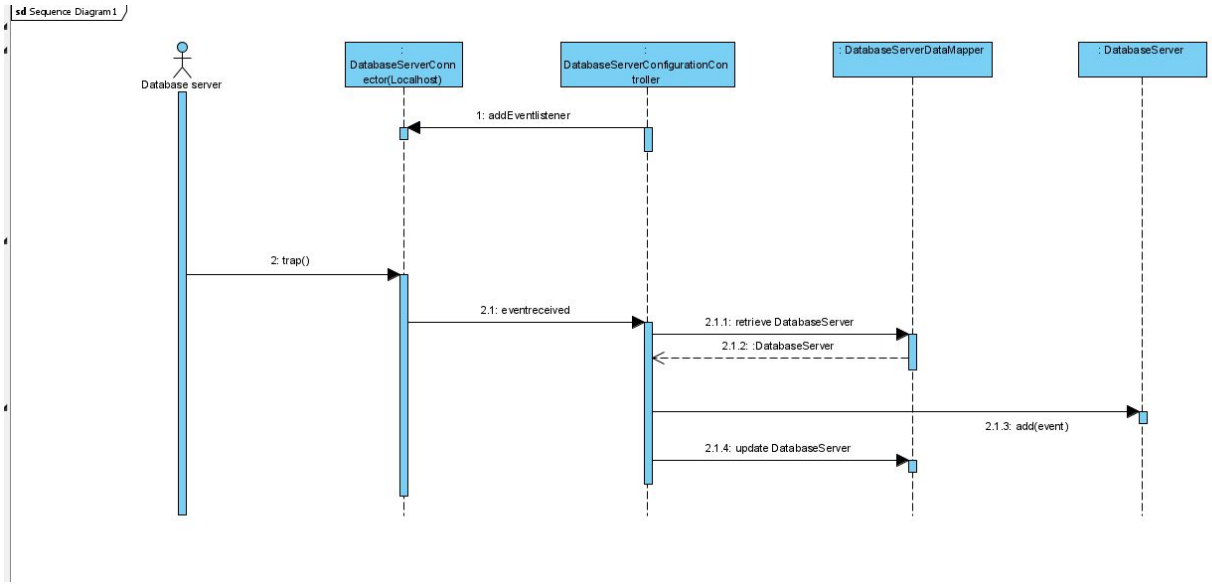


View 2:
Shows the domain objects associated with use cases.



View 3: Sequence diagrams for driving use cases

Sequence for UC-1 and UC-7



Step 7 :

Perform Analysis of current design and review iteration goal and achievement of design purpose

Not addressed	**Partially addressed**	**Completely Addressed**	**Design decision made during the iteration**
		UC-1	Modules across the layers and preliminary interfaces to support this use case have been identified
		UC-2	Modules across the layers and preliminary interfaces to support this use case have been

			identified
		UC-6	Modules across the layers and preliminary interfaces to support this use case have been identified
		UC-7	Modules across the layers and preliminary interfaces to support this use case have been identified
	QA-1		The elements that support the associated use case (UC-1) have been identified
	QA-2		The elements that support the associated use case (UC-5) have been identified
	QA-3		No relevant decision made
	QA-4		The elements that support the associated use case (UC-3) have been identified
	QA-5		No relevant decision made
	QA-6		No relevant decision made
	CON-1		Modules responsible for backing up the system for maintenance have been identified
	CON-2		Modules responsible for database

			connection through servers have been identified.
CON-3			No relevant decision made
CON-4			No relevant decision made
	CRN-2		Additional technologies were identified and selected considering teams knowledge.
		CRN-3	The modules associated with the use cases have been identified and work matrik have also been created.
	CRN-4		The architectural concern of unit testing modules have been partially solved through the use of control approach to connect components associated with modules

ADD Iteration 3

Step 1 : Addressing Key Quality Attribute Scenario Driver (QA-1)

This iteration is solely focused on the Quality Attribute - Availability considering it is the highest prioritized Quality Attribute. Iteration builds on the fundamental structural decisions from Iteration I & 2.

Step 2: Establishing Iteration Goal by Selecting Drivers

In this iteration, we narrow down to QA-1, Availability: The course management system ensures that the system is up and available for use to users on a 24/7 basis and Sundays 12AM-5AM is backup time.

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

For the availability quality scenario, the element to be refined is:

- Database Server

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

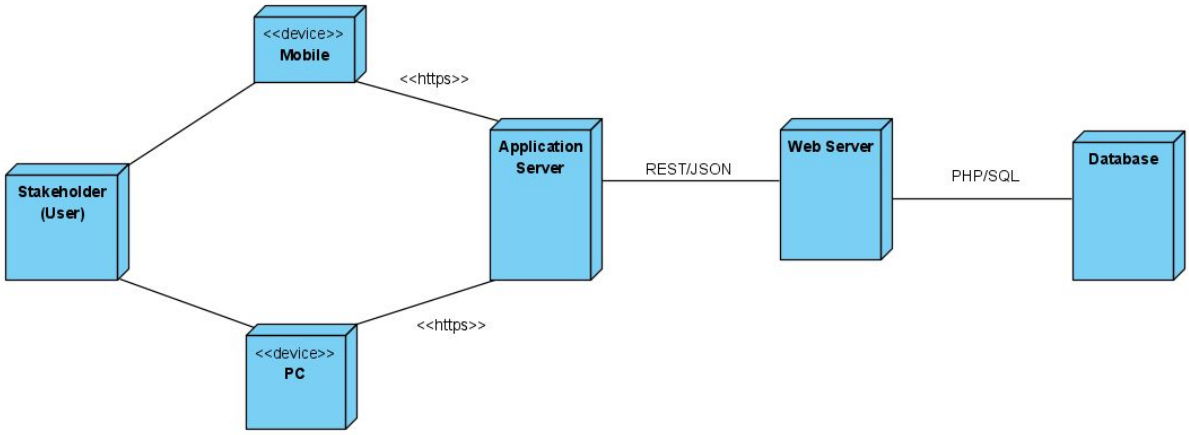
Design Decisions & Locations	Rationale & Assumptions
Introducing the **Back up Zone** where the website to database interactions will be limited/ dormant. The users can make changes or submissions as needed, but changes will reflect after 5AM Sunday morning; right after backup is complete	By introducing a back up, then information will be available for a longer period of time even if a user made a mistake and deleted files from the CMS, and they don't have a copy saved locally.
Introducing the **Back Up storage unit.** Backups from previous versions of the database are kept for a maximum of 4 weeks while new ones overwrite the oldest back-up copy	Information accidentally/intentionally deleted from about 4 weeks ago can be retrieved from the storage unit and recovered to the user and fetched depending on the user's request upon verification of credentials

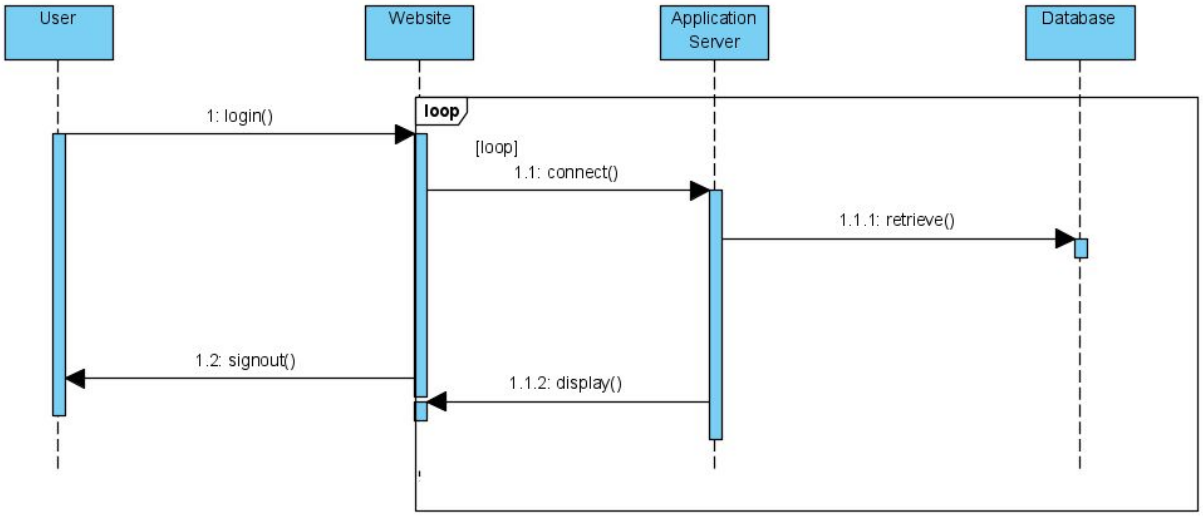
Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

Design Decisions & Location	**Rationale**
Deploy backup notification	This tactic is used to notify users that systems functionality might not be fully available during this time but they can still use the website
Implement course dashboard	This can help each user view the courses that they are enrolled in specifically. The lecturers too can see the specific courses they are teaching for the particular semester and eve plan for the upcoming semester

Step 6: Sketch Views and Record Design Decisions

Elements not previously listed will be included in the refined deployment diagram below to depict the backup system in the system to ensure system availability 24/7 as specified by QA-1





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

Drivers that were completely addressed were not included in this table.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	QA-1		Modules across the system have no time limit and thus the system can run seamlessly 24/7
	QA-2		No relevant decisions made
	QA-3		Back end programming to ensure that website and database have seamless interaction during updates
	QA-4		Elements such as passwords have been introduced in the homepage: Data integrity rules set, so users can't access unauthorized information
	QA-5		System maintainer performs routine checks, daily to check for any faults arising in system and any complaints logged by users
QA-6			No relevant decisions made
	CON-1		A backup zone implemented into the system for weekly routine backups
	CON-2		Modules for database connection have been established and the server is running consistently.

			Maintenance team to do daily checks
CON-3			No relevant decisions made
CON-4			No relevant decisions made