The Design Process

ADD Step 1: Review Inputs

<table>
<tr><th>Category</th><th>Details</th></tr>
<tr><td>Design Process</td><td>This is a greenfield system from a mature domain. The purpose is to produce a sufficiently detailed design to support the construction of the system.</td></tr>
<tr><td>Primary Functional Requirements</td><td>From the use cases presented from the first deliverable, the primary ones were determined to be:
● UC-1 Monitor System Status: Essential Component of CMS
● UC-5 Course Portal Display: Essential Component of CMS
● UC-6 User Profile Utilities: Essential Component of CMS
● UC-7 Manage Database: Essential Component of CMS
● UC-10 User Login: Essential Component of CMS
● UC-11 User Management: Essential Component of CMS</td></tr>
<tr><td>Quality Attribute Scenarios</td><td><table>
<tr><td>Scenario ID</td><td>Importance to Stakeholders</td><td>Difficulty of Implementation according to architect</td></tr>
<tr><td>QA-1</td><td>High</td><td>High</td></tr>
<tr><td>QA-2</td><td>Medium</td><td>Medium</td></tr>
<tr><td>QA-3</td><td>High</td><td>High</td></tr>
<tr><td>QA-4</td><td>High</td><td>High</td></tr>
<tr><td>QA-5</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-6</td><td>High</td><td>Medium</td></tr>
</table>
Only QA-1, QA-3, QA-4, QA-5, QA-6 are selected as</td></tr>
</table>

	drivers
Constraints	All the constraints discussed in the first deliverable are included as drivers.
Architectural Concerns	All the architectural concerns discussed in first deliverable are included as drivers.

Iteration 1:

This first iteration of the CMS system is to establish an overall system structure.

Step 2: Establish Iteration Goal by Selecting Drivers

This is the first iteration in the design of the CMS system, so the iteration goal is to establish *establishing an overall system structure.*

Although this iteration is driven by a general architectural concern, the architect must keep in mind all of the drivers that may influence the general structure of the system. In particular, the architect must be mindful of the following:

- QA-1: Performance
- QA-3: Availability
- QA-5: Usability
- QA-6: Security
- CON-1: Multiple users without server overload
- CON-2: Relational database
- CON-6: Not able to change info in secondary systems

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 refine is the entire FCAPS system, which is shown in Figure 1. In this case, refinement is performed through decomposition.

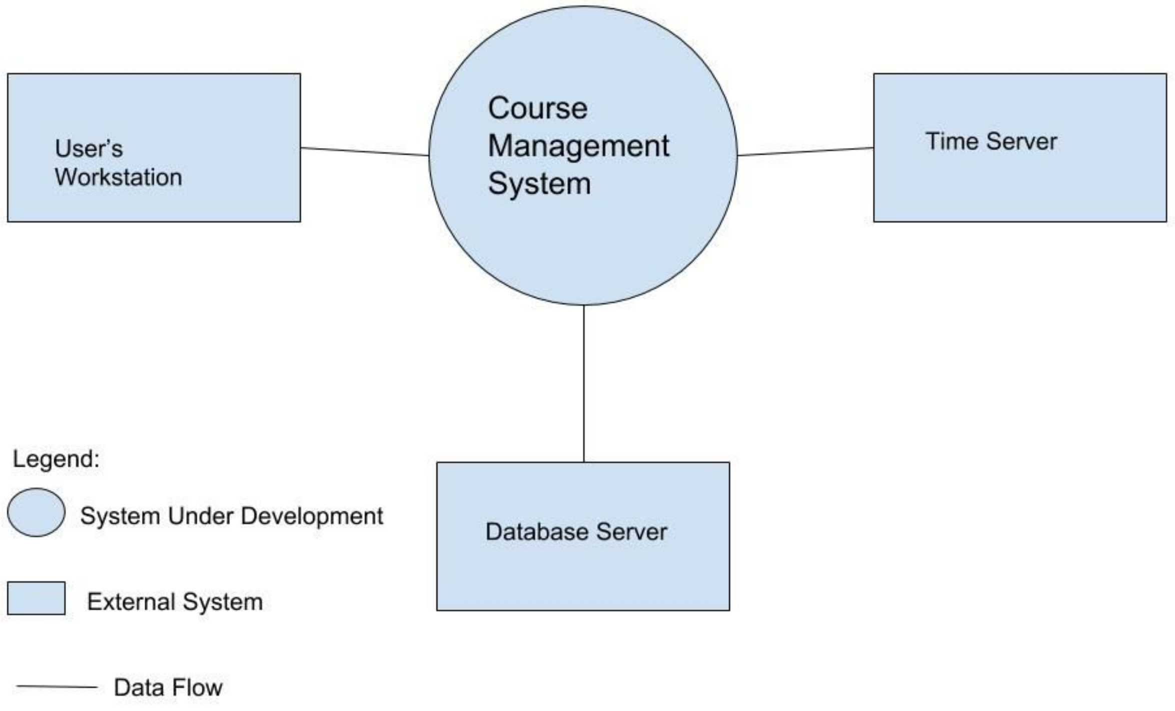


Figure 1: Context Diagram for CMS System

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

Design Decisions and Location	**Rationale**
Logically Structure the client part of the system using Web Application reference architecture	This type of reference architecture must be accessed from the web browser, as evidenced by our own MyCampus. This is the

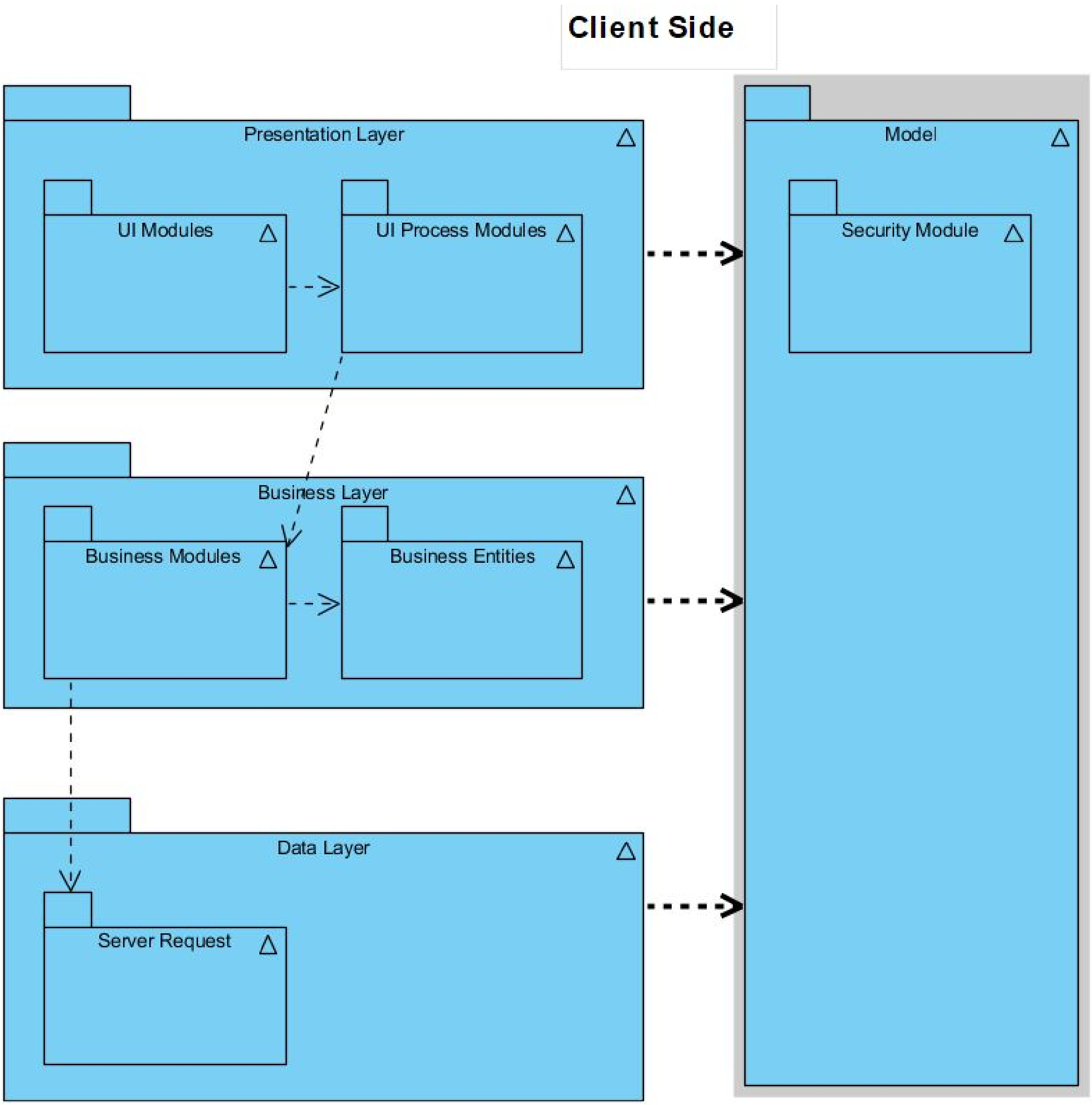
	best option to implement the client part of the system because it has the User Interface, Business Logic and Data Access, which is what we need for this project. Discarded Alternatives: ● Mobile Application: This type of architecture is only geared towards development of applications that are deployed in handheld devices. This alternative was discarded because this type of device was not considered for the CMS System ● Rich Internet Application: This reference architecture is oriented toward the development of applications with a rich user interface that runs inside a web browser. This option was discarded because there was no plugin to run the application. The plugins are used to access the options in the application itself.
Logically structure the server part of the system using Service Application reference architecture.	Service application expose service that are consumed by other application. The Service Application was the best reference architecture to meet the requirements.
Physically structure the application using the **Three-tier deployment pattern**	Since the CMS system is accessed from a web browser, we need to have a database server. Therefore the three-tier deployment is the most appropriate.
Build the user interface of the client application using HTML/CSS and JavaScript	Best to make a clean simple UI, that everyone can use, no matter their experience in technology. Therefore these languages are the best fit and meet the requirements.
Deploy the application using AWS	AWS will be used to deploy the CMS system.

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

The instantiation design decisions considered and made are summarized in the following table:

Design Decision and Location	Rationale
Create a module dedicated to access the database of the Service Application reference architecture	The service agents component from the reference architecture is adapted to access the database. This will ensure the achievement of QA-1, QA-5 and QA-6.

Step 6: Sketch Views and Record Design Decisions



Server Side

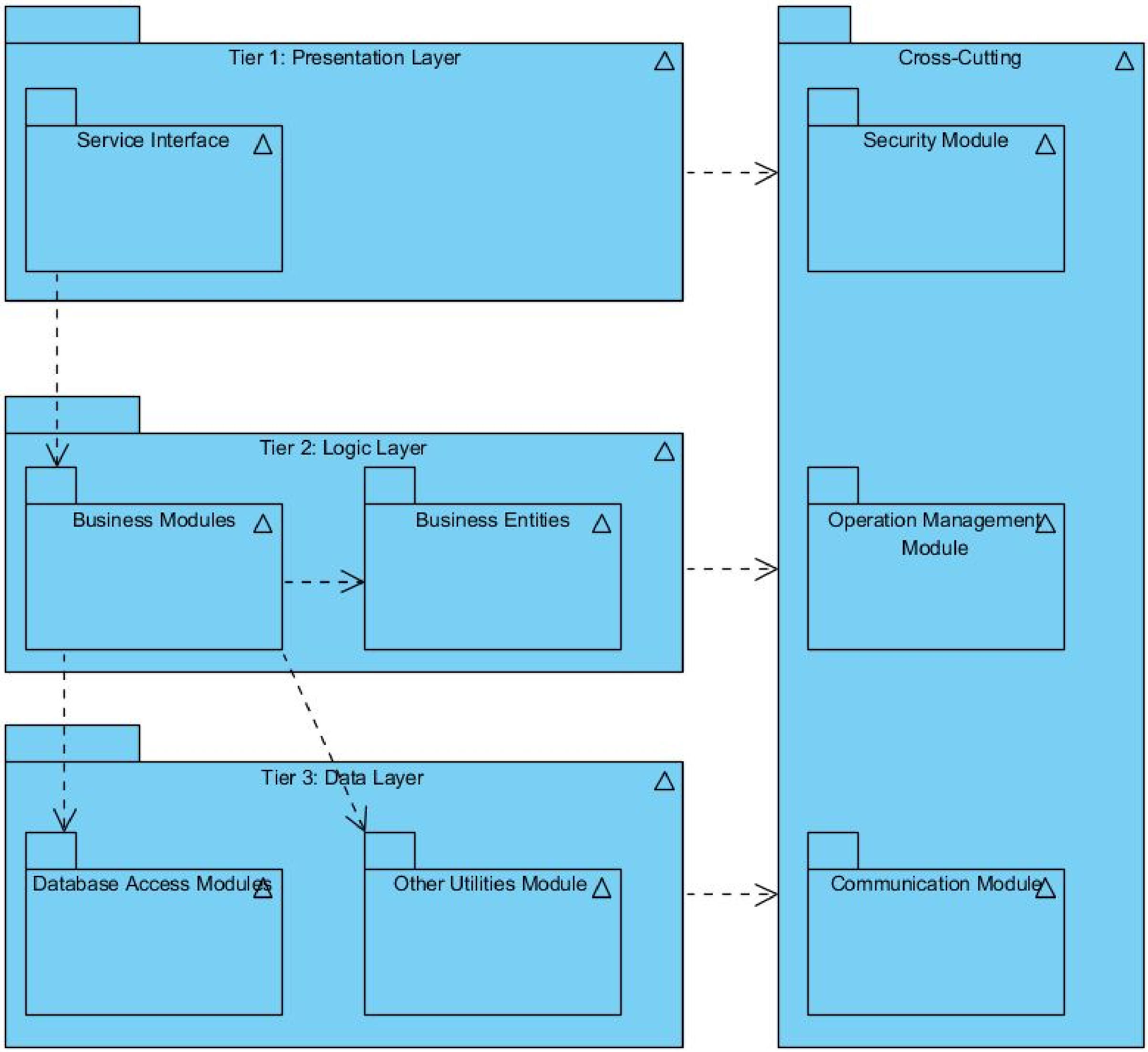


Figure 2: Client Side and Server Side View

Element	Responsibility
Presentation Layer (Client Side)	This layer contains modules that control UI and deals with input and output for users and communicates with the Business Layer
Business Layer(Client Side)	This layer contains modules that perform business logic operations that can be executed locally on client side. This layer also communicates with Data layer.

Data Layer (Client Side)	This layer contains modules responsible for communication with server, and store in database
UI Modules	Render for UI and receive inputs
UI Process Modules	Control flow of all system use case
Business Modules	Implement the business operations
Business Entities	Make up the domain model.
Server Request	Communicate with database to request information
Presentation Layer (Server Side)	This layer contains modules that expose services that are consumed by clients
Logic Layer (Server Side)	This layer contains modules that perform business logic operation that require processing from server side.
Data Layer (Server Side)	This layer contains modules that are responsible for data persistence and for communication
Service Interface	Modules expose service that are consumed by clients
Business Modules	Implement business operations
Business Entities	Make up the domain model
Database Access Module	Responsible for persistence of business entities into the relational database. It performs object oriented to relational mapping and shields the rest of the application from persistent details.
Other Utilities Module	Implement other utilities of the data layer

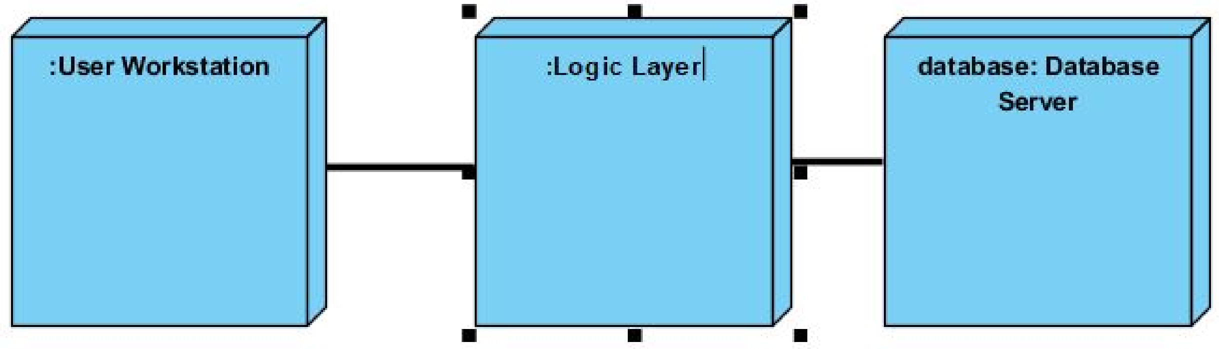


Figure 3: Initial Deployment Diagram of CMS system

Element	**Responsibility**
User Workstation	User browser which host client side logic of CMS system
Logic Layer	Communicating with user workstation and Database server
Database server	Host relational database

Relationship	**Description**
Between User Workstation and Logic layer	Any business logic that user has in interface is calculated here
Between Logic Layer and Database Server	Logic layer communicates with Database Server using SQl

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

The following table summarizes the design progress using the Kanban board technique

Not Addressed	**Partially Addressed**	**Completely Addressed**	**Design Decisions made during the Iteration**

	UC-1		Selected Reference Architecture establish modules that will support this functionality.
	UC-3		Selected Reference Architecture establish modules that will support this functionality
	UC-9		Selected Reference Architecture establish modules that will support this functionality
QA-1			No relevant decisions made
	QA-3		Identification of elements derived from deployment pattern that will need to be replicated
QA-4			No relevant decisions made
	CON-1		3 Tier architecture was used to support multiple users to connect to business layer. Decisions for concurrent access has not been made yet.
		CON-2	Database layer from three tier architecture was used so relation database could be created. Can access the database from data layer of architecture.

CON-3			No relevant decisions made
CON-4			No relevant decisions made
CON-5			No relevant decisions made
CON-6			No relevant decisions made
CON-7			No relevant decisions made
CON-8			No relevant decisions made
CON-9			No relevant decisions made
CON-10			No relevant decisions made