SOFE 3650U: Software Design & Architecture

Establishing Overall System Structure - ADD Iteration 1

Due Date: November 12, 2021

Group Members:

Foram Gandhi (100699245)

Kinjal Shah (100743551)

Rutvi Shah (100747171)

Danial Shaikh (100698628)

Iteration 1: Establishing Overall System Structure

Step 1: Review Inputs

<table>
<tr><th>Category</th><th>Details</th></tr>
<tr><td>Design purpose</td><td>This is a Theatre Ticket Booking System which allows users to book tickets and seats for their preferred movie.</td></tr>
<tr><td>Primary functional requirements</td><td>From the use cases provided in the project progress report, we determined the primary ones to be:
<b>UC-1:</b> Directly Supports Core Business
<b>UC-6:</b> Directly Supports Core Business
<b>UC-7:</b> Directly Supports Core Business
<b>UC-8:</b> Directly Supports Core Business
<b>UC-9:</b> Directly Supports Core Business</td></tr>
<tr><td>Quality attribute scenarios</td><td>
<table>
<tr><th>Scenario ID</th><th>Importance to the Customer</th><th>Difficulty of Implementation According to the Architect</th></tr>
<tr><td>QA-1</td><td>High</td><td>High</td></tr>
<tr><td>QA-2</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-3</td><td>Medium</td><td>Medium</td></tr>
<tr><td>QA-4</td><td>High</td><td>High</td></tr>
<tr><td>QA-5</td><td>High</td><td>Low</td></tr>
<tr><td>QA-6</td><td>Medium</td><td>Low</td></tr>
<tr><td>QA-7</td><td>High</td><td>Medium</td></tr>
</table>
</td></tr>
<tr><td>Constraints</td><td>All of the constraints will be added as drivers.</td></tr>
<tr><td>Architectural concerns</td><td>All architectural concerns below are included as drivers:
<b>CRN-1:</b> Establishing an overall system structure
<b>CRN-2:</b> Leverage team’s knowledge about the WAMP stack including its integration with Javascript, HTML and CSS.
<b>CRN-3:</b> Allocate work to members of the development team</td></tr>
</table>

Step 2: Establish Iteration Goal by Selecting Drivers

The drivers that the architect must be mindful of for the first iteration are:

- QA-1: Availability
- QA-2: Interoperability
- QA-4: Performance
- QA-5: Security
- QA-7: Usability
- CON-2: Developer and Technician communication relationship
- CON-4: System accessibility and availability
- CON-5: Untreated faults leading to compromisation of system security
- CRN-2: Use the teams knowledge of Web Programming

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

The element chosen to be refined for this Theatre Ticket Booking effort is the entire system (shown in **Fig. 1**). The refinement will be done through decomposition.

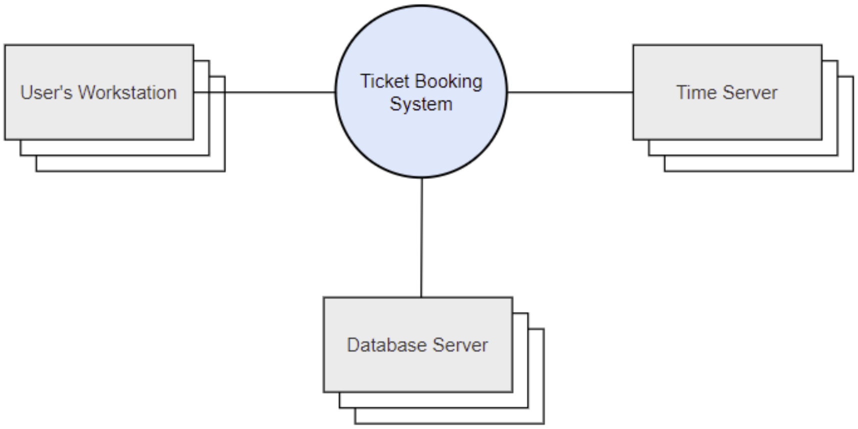

Fig 1: Context diagram for the Theatre Ticket Booking System consists of User's Workstation, Time Server & Database Server as elements to be refined.

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

Chosen Alternatives:

Design Decision and Locations	**Rationale**
Logically structure the client and server part of the system to use **Web Application** reference architecture	The Web Application reference architecture supports access to applications though the web browser. Plus this application does not require a rich user interface thus this reference architecture would be perfect to implement with minimal effort. Furthermore, this architecture also facilitates deployment and updating plus it provides high portability of the application.

Discarded Alternatives:

Alternative	**Reason for Discarding**
Mobile Applications	This reference architecture enables the use of handheld mobile devices, however since our system does not support this type of device, the architecture was not considered for accessing our system.
Rich Client Applications	This reference architecture requires the installation of applications on the user's PC without network connectivity . It also does not run in a web browser, unless an external technology is used. This was not considered for accessing our system since we do not require external applications to be installed and only want web browser capability.
Rich Internet Application	The RIA reference architecture is oriented towards the implementation of rich user interfaces in web applications that run inside a web browser. The system does not need a rich user interface, plus there is no need to make the architecture more complex than it needs to be.

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

Design Decision and Location	**Rationale**
Remove Service Agents from the Data Layer	There are no external services being utilized in this system thus there is no need for it.
Create a module dedicated to accessing time servers in the Data Layer	The reference architecture is modified to conceptualize the access to time servers. This further facilitates the achievement of QA-4. The addition of this module also addresses UC-1 to UC-5.

Step 6: Sketch Views and Record Design Decisions

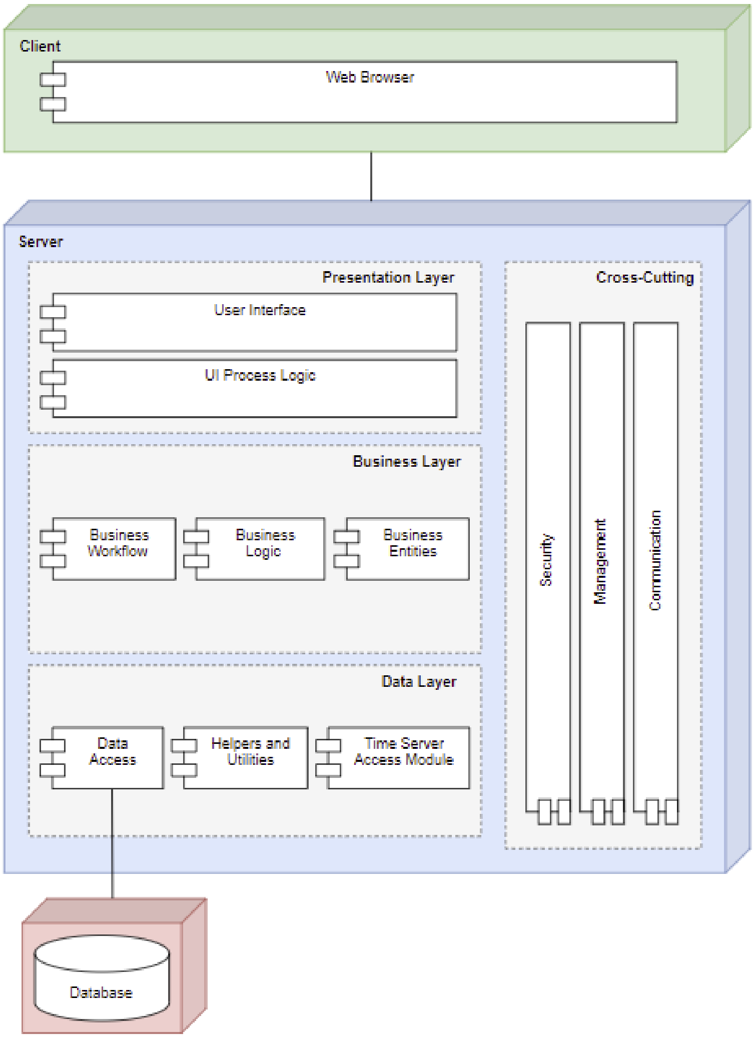

Fig 2: Shows the sketch of a module view of the reference architectures that were selected for the client and server applications. These have now been adapted according to the design decisions we have made.

The following table summarizes the information that is captured:

Layer	**Module**	**Responsibility**
Client	Web Browser	Movie Ticket Booking Application runs on the client's web browser.
Server - Presentation Layer	User Interface	Receives user interactions and presents the information

		to the users. It contains UI elements as buttons and text fields as well as a form in the ticket checkout page.
Server - Presentation Layer	UI Process Logic	Manages the control flow of the application’s use cases and provides data coming from the business layer to the user interface components. It is also responsible for data validation. An example is the Login/Signup Page.
Server - Business layer	Business Workflow	Involves the execution of multiple use cases (such as ticket selection and seat selection since seats selected must be equal to number of tickets previously selected)
Server - Business Layer	Business Logic	Retrieves and processes application data and applies business rules on the data (such as seat booking page and checkout page).
Server - Business Layer	Business Entities	Represents the entities from the business domain and the associated business logic.
Server - Data Layer	Data Access	Provides the common components needed to retrieve and store information. It uses the OMDB API for retrieval of movies and movie info, as well as retrieval from the database to verify/check login/signup info.
Server - Data Layer	Time Server Access	This module isolates and abstracts operations to support communication between different types of time servers partially addressing QA-4.

Server - Cross-Cutting Layer	Security - User authentication - Password encryption/decryption	Handles security aspects such as user authorization and authentication, as well as password encryption/decryption
Server - Cross cutting Layer	Management	Handles validation, exception management and logging across all layers.
Server - Cross-Cutting Layer	Communication	Handles communication mechanisms across layers and physical tiers
External Data Sources	Database	Stores all data from the system, later accessed by the DataAccess module in the Data Layer (using MySQL to store databases).

Step 7: Perform Analysis of Current Design and Review Iteration

Not Addressed	**Partially Addressed**	**Completely Addressed**	**Design Decisions Made During the Iteration**
		UC-1	Selected reference architecture establishes the modules that will support this functionality.
	UC-2		Selected reference architecture establishes the modules that will support this functionality.
	UC-3		Selected reference architecture establishes the modules that will support this functionality.
	UC4		Selected reference architecture establishes the modules that will support this functionality.
	UC-5		Selected reference architecture establishes the modules that will support this functionality.
	UC-9		Selected reference architecture partially establishes the module

			that will support this functionality.
		UC-10	Selected reference architecture establishes the modules that will support this functionality.
		UC-11	Selected reference architecture establishes the modules that will support this functionality.
		UC-15	Selected reference architecture establishes the modules that will support this functionality.
		QA-1	Reference of this quality attribute in the selected reference architecture. The application will be available on the internet 24/7 unless there is down time taken.
QA-2			No relative decisions made in this iteration for the quality attribute.
	QA-4		The addition of Time Server Access in the Data Layer partially addresses this QA by establishing communication with Time Servers.
	QA-5		Reference of this quality attribute in the selected reference architecture.
		QA-7	Selected reference architecture establishes the modules that will support this quality attribute.
CON-1			Decisions for the work efficiency of the technician have not been made yet.
CON-2			Decisions for the communication relationship between the developer and technician have not been made yet.
	CON-3		Constraint is referenced by the architecture model although no relative decision has been made.
		CON-4	Referenced architecture provides

			the modules that will support this constraint.
		CON-5	Referenced architecture provides the modules that will support this constraint.
CON-6			No relative decision has been made for this constraint.
		CRN-1	Selection of reference architecture and deployment pattern.
	CRN-2		Technologies and reference architecture that have been considered up to this point have taken into account the knowledge of the developers.
CRN-3			No relative decisions have been made for this particular concern.