Iteration 1: Establishing an Overall System Structure

ADD Step 1: Review Inputs

<table>
<tr><th>Category</th><th>Details</th></tr>
<tr><td>Design Purpose</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 that we have created, the primary ones were determined to be:

UC-2: Because it directly affects the core business.

UC-3: Because it directly affects the core business, and controls user flow.

UC-5: Because it contains technical improvements and issues with it (Related to QA-2 and QA-5)

UC-6: Because it directly affects the core business and functionality.</td></tr>
<tr><td>Quality attribute scenarios</td><td>
<table>
<tr><td>Scenario ID</td><td>Importance to the Customer</td><td>Difficulty of Implementation According to the Architect</td></tr>
<tr><td>QA-1</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-2</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-3</td><td>Low</td><td>High</td></tr>
<tr><td>QA-4</td><td>High</td><td>High</td></tr>
<tr><td>QA-5</td><td>Medium</td><td>Medium</td></tr>
</table>
</td></tr>
</table>

QA-6	Medium	Low

Constraints

ID	Constraints
CON-1	The software must be accessed through the available platforms available only (currently only via web-browser, with support for Chrome and Edge on any OS)
CON-2	If the web-browser gets closed due to any unforeseen issues, it will automatically save the most recent level that the user has completed and store the data into the database. It cannot save the current uncompleted level that the user is currently on.
CON-3	Login domain should be able to handle a minimum user login attempt from at least 20+ users at once.
CON-4	Advertisements need to be implemented without causing excessive attempts to have the user watch it – while providing the ability to skip the advertisements by purchasing the “full” software.
CON-5	Any major updates regarding future downloadable/purchasable content should be relayed in a timely manner to the users with a minimum warning of at least 20- 25 days prior.
CON-6	Hi-score system should be

<table>
<tr><td></td><td>
<table>
<tr><td></td><td>unique to each respective user’s login credentials to the database and allow users to view other users hi-scores that are visible on the leaderboards.</td></tr>
<tr><td>CON-7</td><td>Users should not be able to drag the item/object within the software into unacceptable zones, as they are only allocated within the positional changes governed by the code of the software (typically: up, down, left, right, and any diagonal direction of a movement space of 1 square).</td></tr>
</table>
</td></tr>
<tr><td>Architectural concerns</td><td>
<table>
<tr><td>ID</td><td>Concern</td></tr>
<tr><td>CRN-1</td><td>Establishing an overall initial system structure</td></tr>
<tr><td>CRN-2</td><td>Leverage the team’s knowledge about Java technologies, including Spring, JSF, Swing, Hibernate, Java Web Start and JMS frameworks, and the Java language.</td></tr>
<tr><td>CRN-3</td><td>Allocate work to members of the development team</td></tr>
</table>
</td></tr>
</table>

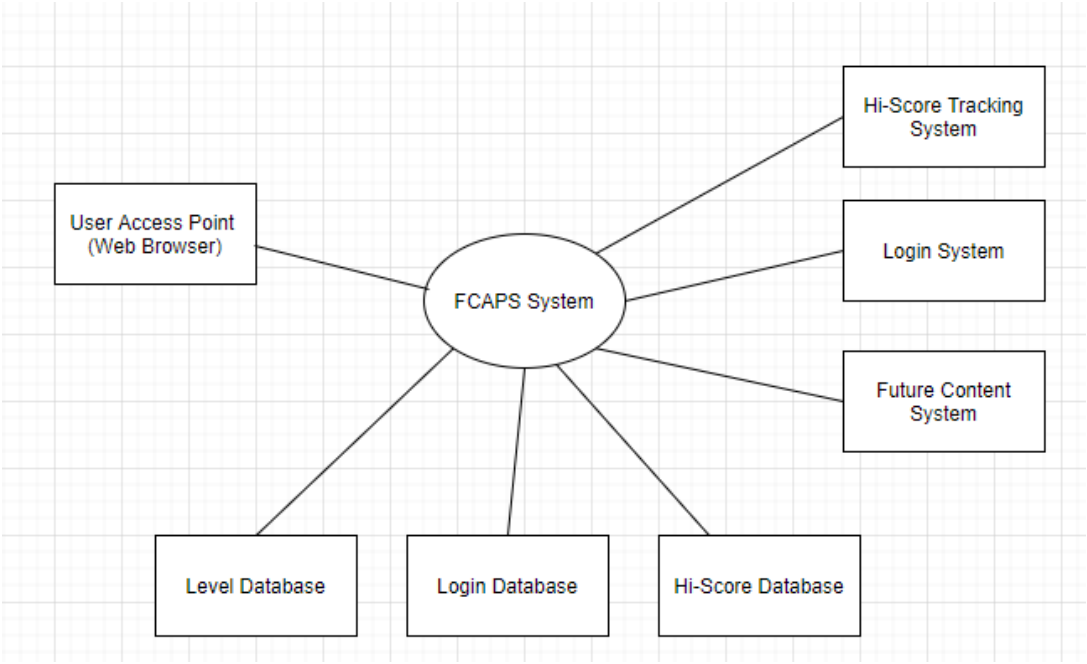
Step 2: Establish Iteration Goal by Selecting Drivers

Drivers to keep in mind for the general structure of the system are as follows:

- QA-1: Performance
- QA-2: Modifiability
- QA-3: Availability
- QA-5: Testability
- QA-6: Security
- CON-1: System must be accessed through the available platforms (currently via web-browser, with support for Chrome, and Edge on any OS)
- CON-2: System will automatically save most recent level data if user disconnects.
- CON-3: Login database should be able to handle a minimum 20+ users at once.

- CON-6: Hi-score system should be unique to each respective user's login credentials and allow them to view their own hi-scores.
- CON-7: Users should not be able to drag item/object into unacceptable zones, as they can only move in the allocated positional changes designed by the program.
- CRN-2: Leverage the use of Java technologies knowledge to help design the system.

Based on the selected drivers, the diagram for the FCAPS system is as follows:



Step 3: Choose one or more Elements of the System to Refine
We will be decomposing the game itself and the FCAPS system.

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

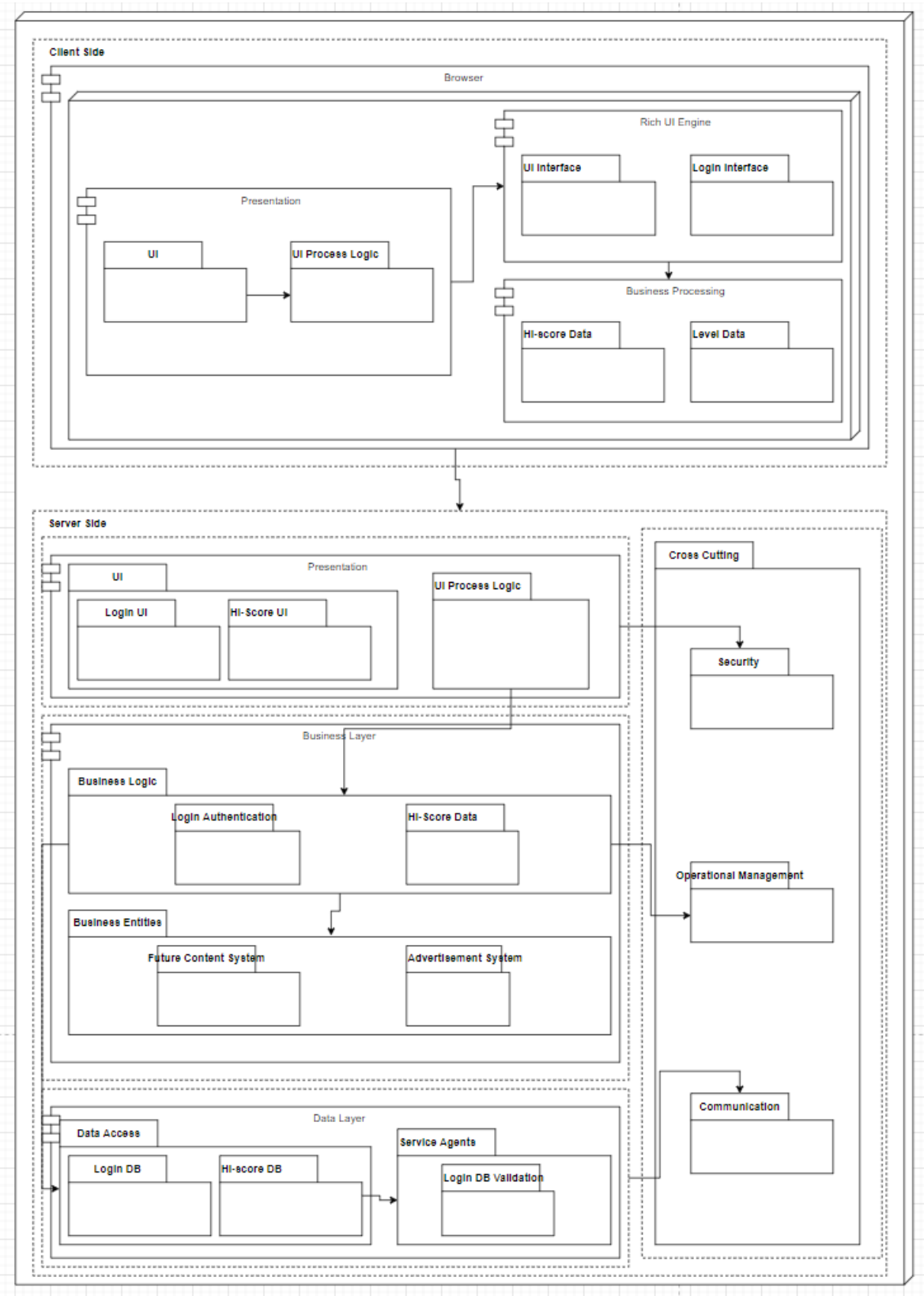
Design Decisions and Locations	Rationale
Logically Structure the client system using the **Rich Internet Applications (RIA)** reference architecture	The Rich Internet Applications (RIA) reference architecture supports the design requirement of designing the system to work on web-browsers accessed by the user. These applications support a rich user interface and have the capabilities to manage user interface with UI logic and being able to run the application without having to install anything on the user machine. This helps meet the requirements for UC-3, CON-1, CON-3, QA-1 and QA-3.
Logically Structure the server part of the system using **Web Applications** reference architecture	Since we are using RIA for the client side system, Web Applications is an excellent counterpart for the server side system. The reason for this is because it uses all the web browser logic we need to provide and access user credentials, as well as providing cross-cutting functionality for security, logging and exception management.
Physically structure the application using the **Three-Tier Deployment** pattern	Due to the system needing to be accessed from a web browser (CON-1), and the two reference architectures selected favor using Three-Tier deployment, this pattern is the most applicable to our system.
Build the user interface of the client application using **Java Swing Framework**	The Java Swing Framework is very effective for building Rich Internet Applications and contains a lot of useful libraries like user interface components, JMenu, JTree, JButton and among other things which are useful to our web-browser system.

Deploy the application using the **Java Web Start Framework**	As this is intended to be a web-browser based application, this can launch the application without problem. It also facilitates updates effectively which is also a constraint (CON-5). It also has very useful design patterns and tactics like: Security and Performance.

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

Design Decision and Location	Rationale
Create a module that connects to the Data Layer of the **Web Applications** server reference architecture that can facilitate login, and hi-score database.	The web application should take in user credentials to provide access to the hi-score database records pertaining to their unique login ID, which is verified from the login system database.
Create the Service agent's component from the Data layer of the **Web Applications** architecture to connect with a new module that contains databases for login and hi-scores.	The service agents' abstract communication mechanisms are important to transfer hi-score and level information of the user to the database. This data can also be shared on another platform at the choice of the user (sharing hi-scores with friends and other public users).
Remove the isolated storage module in the Client section of **Rich Internet Applications (RIA)**	There is no requirement of data being stored into an isolated storage on the client's side. All data will be stored onto the databases according to credentials and data taken (hi-scores, advertisement removals, etc.).
The Rich UI engine should be rendering all UI elements that relate to the login database and hi-score database in the Client section of **Rich Internet Applications**	The user should be able to login into the system, and be prompted to view hi-scores, continue, or start a new level, and/or make changes to any profile settings that are available to them.

Step 6: Sketch Views and Record Design Decisions



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

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions
	UC-2		Reference structure establishes the modules to support this functionality.
	UC-3		Reference structure establishes the modules to support this functionality.
	UC-7		Reference structure establishes the modules to support this functionality.
	QA-1		Login system and Hi-Score tracking system modules enable acceptable standards for user thresholds.
QA-2			**No relevant decisions made. This part of the system has yet to be identified.**
	QA-3		Elements that are deployed and selected through the **Web Applications** and **RIA** that can be replicated to create required system.

QA-5			**No relevant decisions made. This part of the system has yet to be identified.**
QA-6			**No relevant decisions made. This part of the system has yet to be identified.**
CON-1			**No relevant decisions made. This part of the system has yet to be identified.**
CON-2			**No relevant decisions made. This part of the system has yet to be identified.**
	CON-3		Using the **RIA** structure, it allows for a login system with very responsive UI that is connected to the **Web applications** structure.
	CON-6		Use of the **Data Layer** from the **Web Applications** structure uniquely identifies user scores and login, structure support is crucial to functionality.
CON-7			**No relevant decisions made. This part of the system has yet to be identified.**

		CRN-1	Selection based on reference architectures chosen, **Web Applications** and **RIA** benefit system.
	CRN-2		Team's knowledge on different Java technologies allows for possible functionality and ability to leverage features that benefit the system.
CRN-3			**No relevant decisions made. This part of the system has yet to be identified.**