Iteration 2: CMS System

This section presents the results after refining the architecture designed from iteration 1. The result is having components with interfaces that are able to support the primary functionalities of the system.

Table of Contents

2.2 Step 2: Establish Iteration Goal

The goal of this iteration is to identify all the elements that support the primary functionality. The inputs are elements that will also support the primary drivers.
In this second iteration, the architect considers the systems primary use cases:

- UC-1
- UC-2
- UC-3
- UC-6

2.3 Step 3: Choose Elements to Decompose

Since functionality is supported by the elements that are spread across the layers of the system, the elements to decompose will be the modules that were defined in the layers of the selected reference architecture from the previous iteration, which was Web Applications. In this scenario, the layers are the presentation layer, business, data and cross-cutting.

2.4 Step 4: Choose design concepts that satisfy the inputs considered in the iteration

Table 2.1, goes further to refine what frameworks and databases are to be chosen for the system and how it satisfies the drivers

Design Decisions and Location	**Rationale and Assumptions**
Use MongoDB database	MongoDB supports web applications and it is a hierarchical database. It is easy to use as records are stored in a similar fashion as JSON
Use Bootstrap Framework for the UI	Bootstrap is easy to use and easy to integrate framework that helps make highly functional and user-friendly interfaces. This would help in maintaining UC-2, QA-6 and CON-5

TABLE 2.1 Design Decisions to refine

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

The architecture model from iteration 1 (Figure 1.2) gave a basic design for the system. Here, the design is refined into components. Table 2.2 shows all the elements and their responsibilities from the extended module view which is located in Step 6, Figure 2.1.

Element	Responsibility
Request Manager (Client Side)	Responsible for communication with the server-side logic
Network Status Controller (Server Side)	Responsible for detecting the status of the network to the server and status of the user's session. This will complete UC-1
Request Receiver (Server Side)	Provides a facade that receives requests from clients
Request Handler (Server Side)	Responsible for handling requests received from the service layer.
Data Mapper (Server Side)	Responsible for taking the data collected and directing it to the proper data source. Responsible for persistence operations related to the events
Data Retriever (Server Side)	Responsible for getting the data from the data sources.UC-2
Backup Manager (Server Side)	Responsible for maintaining backups of the session. This will achieve UC-6.

TABLE 2.2 Table of refined components and their responsibilities

2.6 Step 6: Sketch Views and Record Design Decisions

Step 6 displays a refined model created from step 4 and step 5. It further defines the elements from iteration 1 by creating components with interfaces and methods.

Extended Module View

Figure 2.1 shows a sketch of a module view with modules that are derived from the elements within the layered diagram that also relate to the primary use cases that are defined as inputs for this iteration.

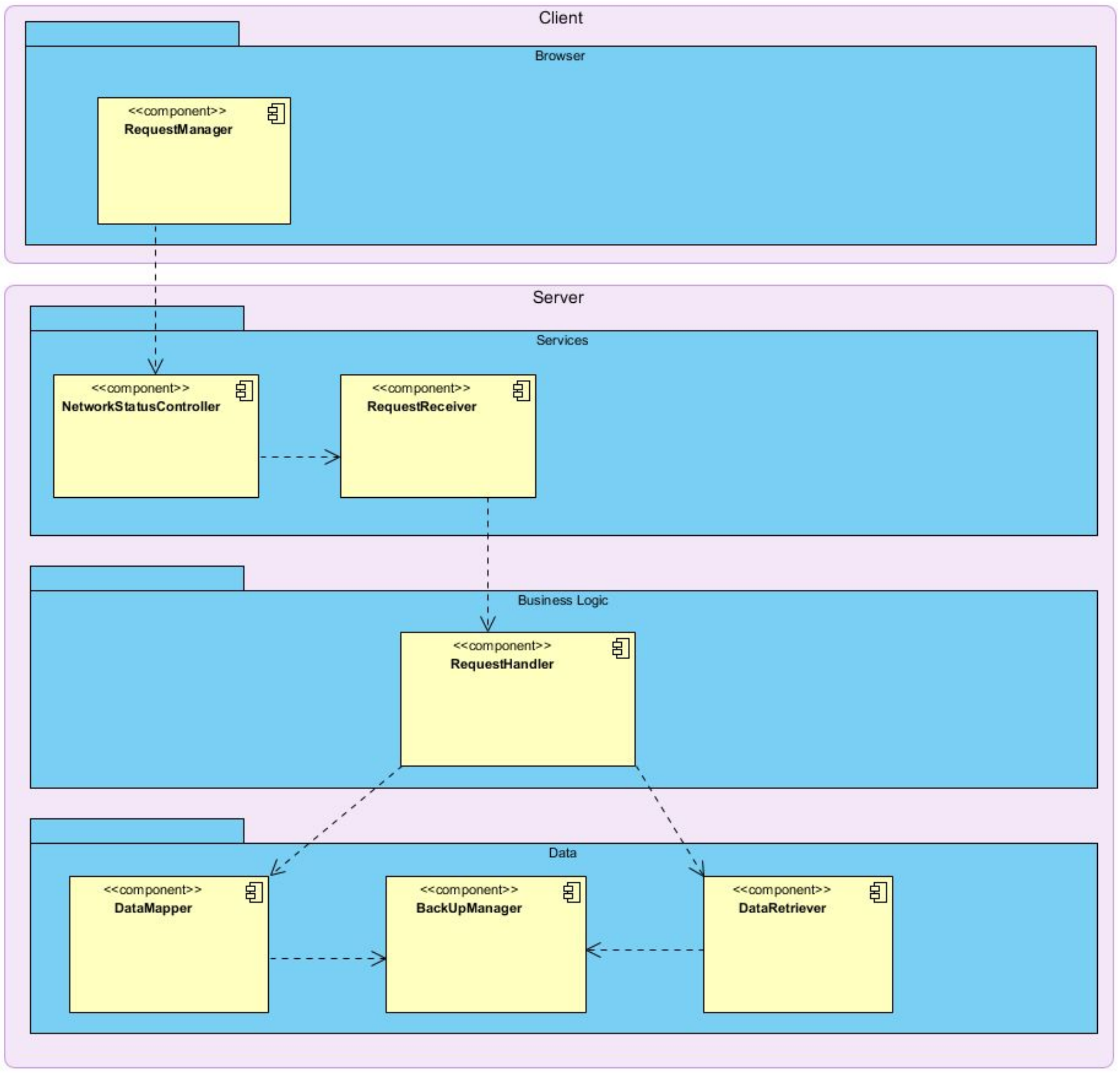


FIGURE 2.1 Extended Module View

UC-1 Login

Figure 2.2 shows an initial sequence diagram for UC-1(log in). It shows how the user opens the browser and logs in using their credentials. To verify the login, the login request is sent and received and then compared to the data in the database. Upon successful login, the user is notified through the browser.

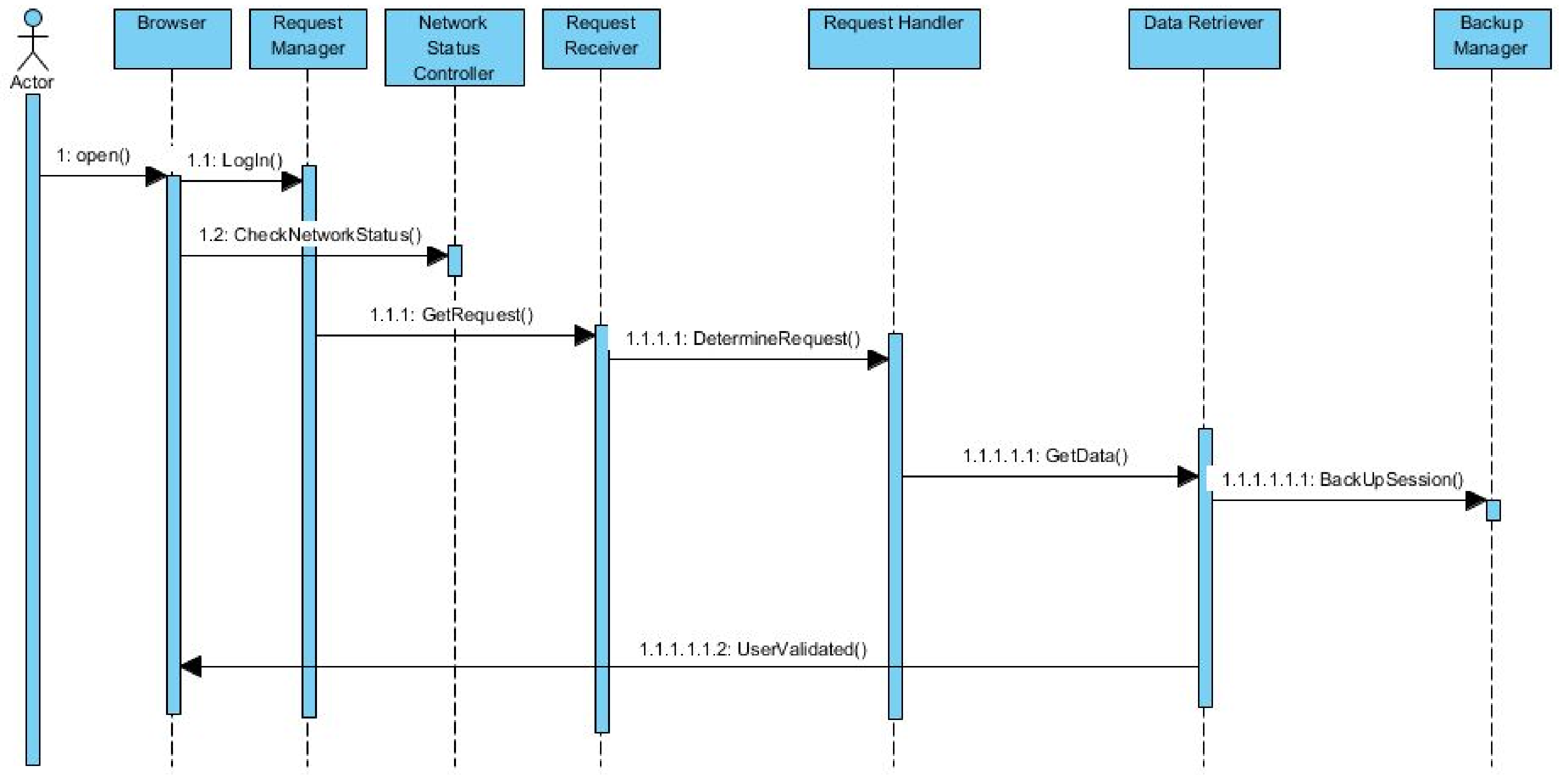


FIGURE 2.2 Sequence Diagram of Use Case 1

UC-1 Methods

From the sequence diagram, methods were interpreted that can be applied to help with logging in a user. Table 2.3 states the methods and a short description of them.

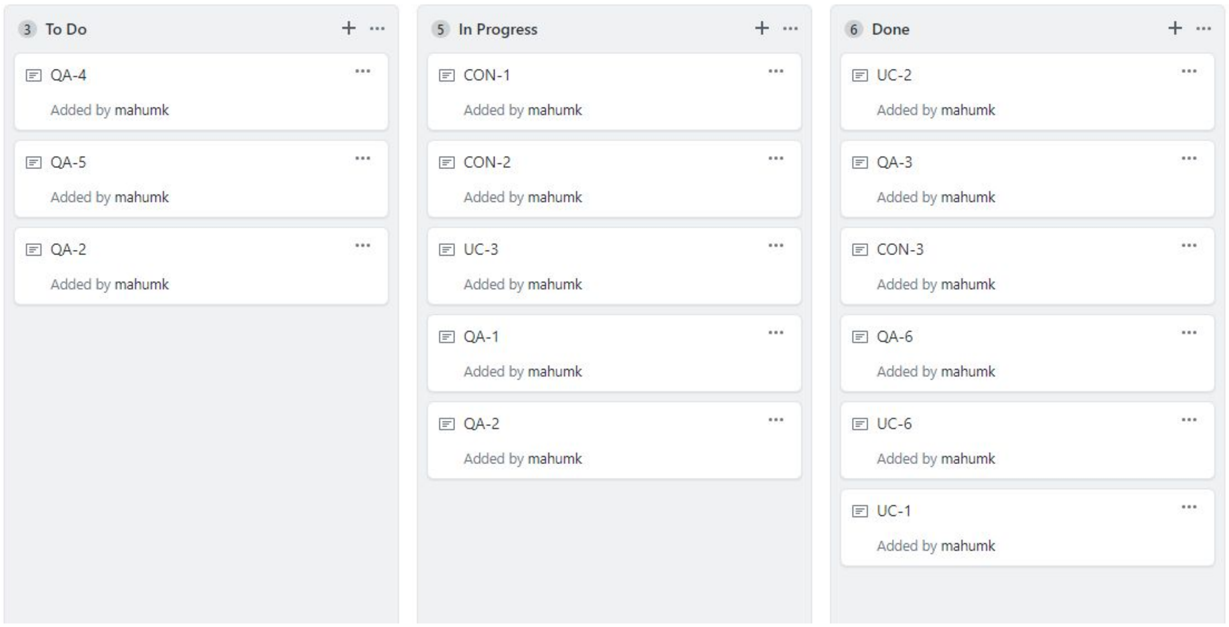
Method Name	**Description**
Element: Request Manager	
logIn()	This method allows the actor to input their credentials into the form. Then sends a request to verify the input
Element: Network Status Controller	
Boolean CheckNetworkStatus()	This method checks to see if there is a connection made to the server
Element: Request Receiver	

getRequest()	Gets request from the user. In this case, it is getting a request to log into the system
Element: Request Handler	
determineRequest()	This method determines whether the request
Element: Data Retriever	
getData();	This method tries to get the data from the data sources. In this case, it checks to see if the user is in the system.
Element: BackUp Manager	
Void backUpSession()	This method logs the login attempt.

TABLE 2.3 Component MethodsMethods

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

The following Kanban board (Figure 2.3) summarizes the design progress made after the second iteration



3 To Do	5 In Progress	6 Done
QA-4 Added by mahumk	CON-1 Added by mahumk	UC-2 Added by mahumk
QA-5 Added by mahumk	CON-2 Added by mahumk	QA-3 Added by mahumk
QA-2 Added by mahumk	UC-3 Added by mahumk	CON-3 Added by mahumk
	QA-1 Added by mahumk	QA-6 Added by mahumk
	QA-2 Added by mahumk	UC-6 Added by mahumk
		UC-1 Added by mahumk

FIGURE 2.3 Kanban board after iteration 2