SOFE3650U Final Project: Iteration 2

Group 21

Name	Student ID
Aryan Singh	100748196
Joshua Ramnaraine	100692194
Fredrick Tetteh	100569808

Step 2: Establish Iteration Goal by Selecting Drivers

The primary use cases that are considered to be supporting the primary functionality of the Yeezy Books system are:

- UC-1
- UC-3
- UC-5
- UC-6

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

In the reference architecture in iteration 1, the functionality will require support from the components related to the modules in different layers.

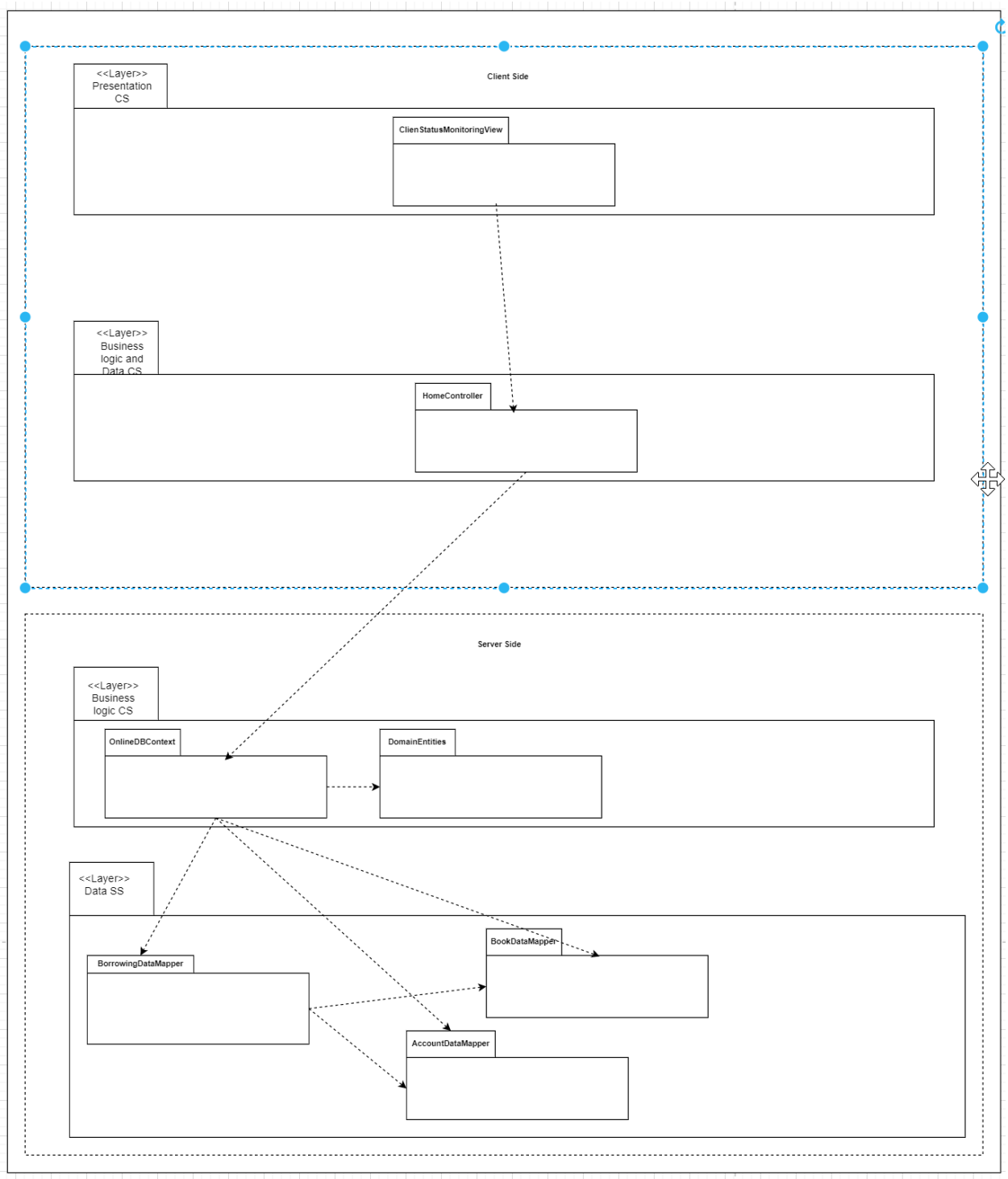
Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers

Design	**Rationale**
Create a domain model for application	To identify the major entities of system, as well as relationships between entities
Identify domain objects that align to the functional requirements	For every functional element in the application, it will have to be stored in it's own domain object.
Use Swing Framework	Provides the necessary java interfaces that ASP.NET uses and follows an MVC pattern like ASP.

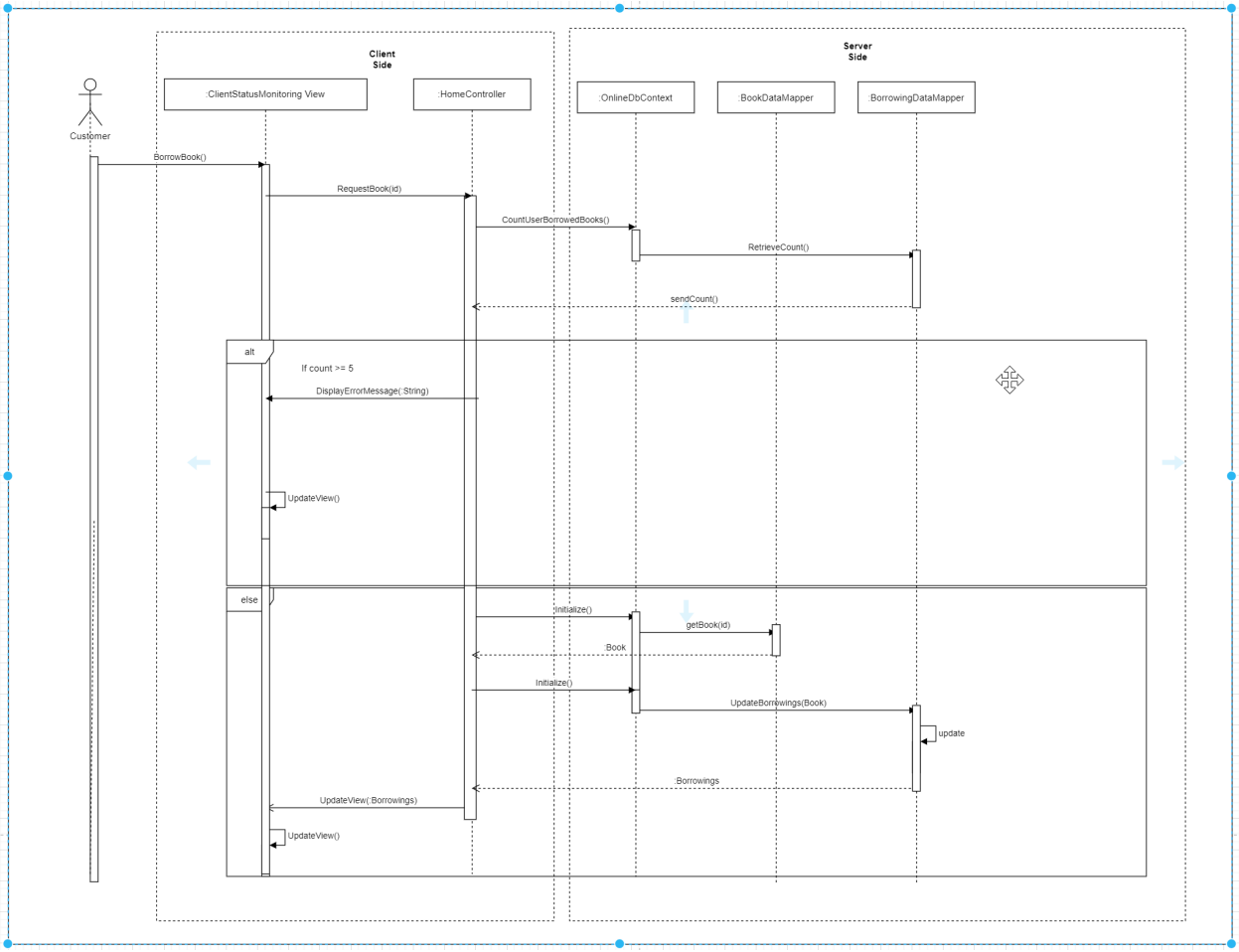
Step 5: Instantiate architectural elements, allocate responsibilities, define interfaces

Design Decisions and location	Rationale
Create an only an initial domain model	The entities/objects associated with the primary use cases need to be identified and modelled but only an initial domain model is created to accelerate this design phase.
Map system use cases to domain objects	Mapping and identification of the system domain objects can be made by analyzing the systems use cases.
Decompose the domain objects across layers to identify layer specific module with an explicit interface	Ensures that modules that support all the functionalities are identified.
Connect module components using SWING framework	This framework follows a single threaded programming model and helps to provide between the code structure and the graphic presentation of a SWING based GUI
Associate frameworks with a module Within the data layer	ORM mapping is encapsulated within modules that are contained in the data layer.

Step 6: Sketch Views and Record Design Decisions



Element	**Responsibility**
UserInterface(Login)	Responsible for the control of the User interaction module when the user logs in
ClientStatusMonitoringView	Displays the desired representation view for an employee or customer. Depending on the login attempt and updates the views based on the events received.
HomeController	Responsible for providing the proper information to the presentation layer for displaying the proper views with the proper updates.
OnlineDBContext	Allow for user interactions
Domain Entities	Contains the entities from the domain model (server-side)
Book Data Mapper	Responsible persistence operations (CRUD) related to the regions
Account Map Data Mapper	Responsible persistence operations (CRUD) related to the regions
Borrowing Data Mapper	Responsible persistence operations (CRUD) related to the regions

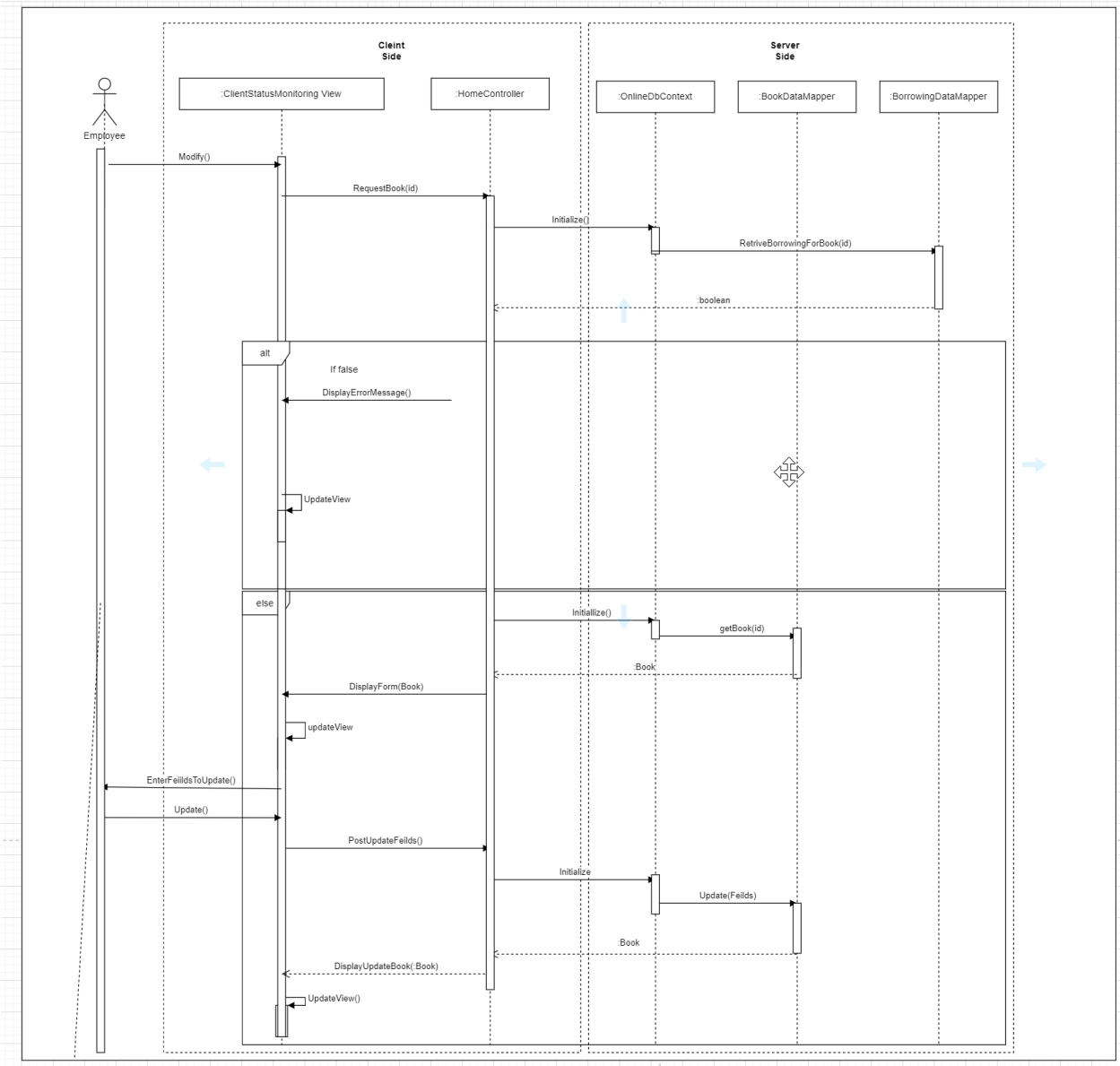


Use Case for UC-6 (Key:UML)

From this interaction, initial methods for the interacting methods for the interfaces can be identified.

Method Name	**Element**	**Description**
RequestBookID()	ClientStatusMontioringView	Responsible for the book id of the book that was clicked to the controller
UpdateView()	ClientStatusMonitoringView	Responsible for updating the view seen by the user based on the parameters passed by the controller
Initialize()	HomeController	Instiatiates access to the database
UpdateView(Borrowings)	HomeController	An abstract method that is

		implemented ClientStatusMonitoringView element
getBook(id)	OnlineDbContext	Method to retrieve a Book Object within the database based on id passed
RetrieveCount(user_id)	OnlineDbContext	Responsible for the retrieving the number of books borrowed books by a user based on the user_id
updateBorrrowings(Book, user_id)	OnlineDbContext	Responsible the passing the parameters in the correct format to the mapper class
Update(Book, user_id)	BorrowingsDataMapper	Responsible for updating the database using the given parameters

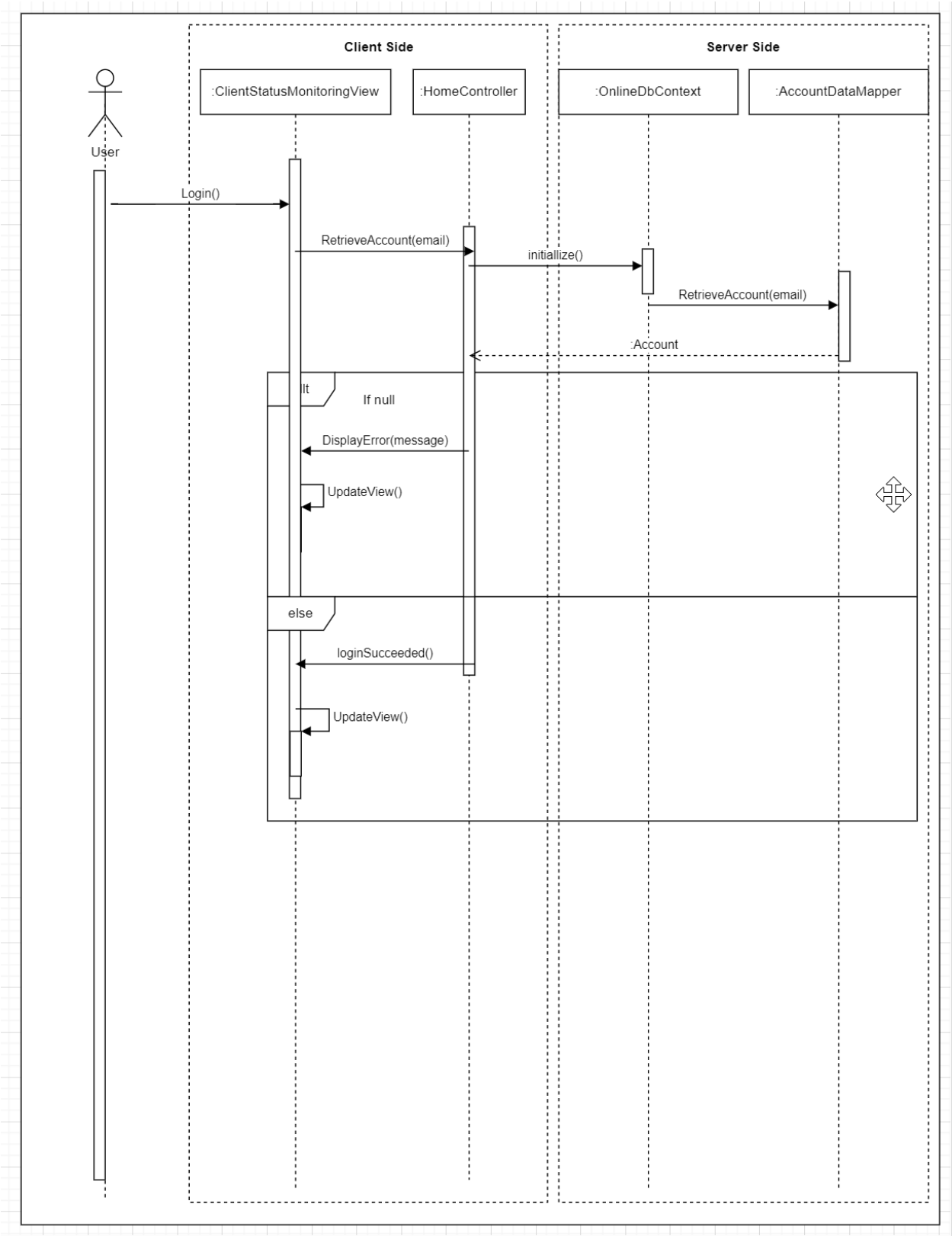


Use Case for UC-5 (Key:UML)

From this interaction, initial methods for the interacting methods for the interfaces can be identified.

Method Name	**Element Name**	**Description**
Modify()	ClientStatusMonitoringView	Used to select the id of book chosen to modified
RequestBook(id)	ClientStatusMonitoringView	Used to pass the id of the book to the controller

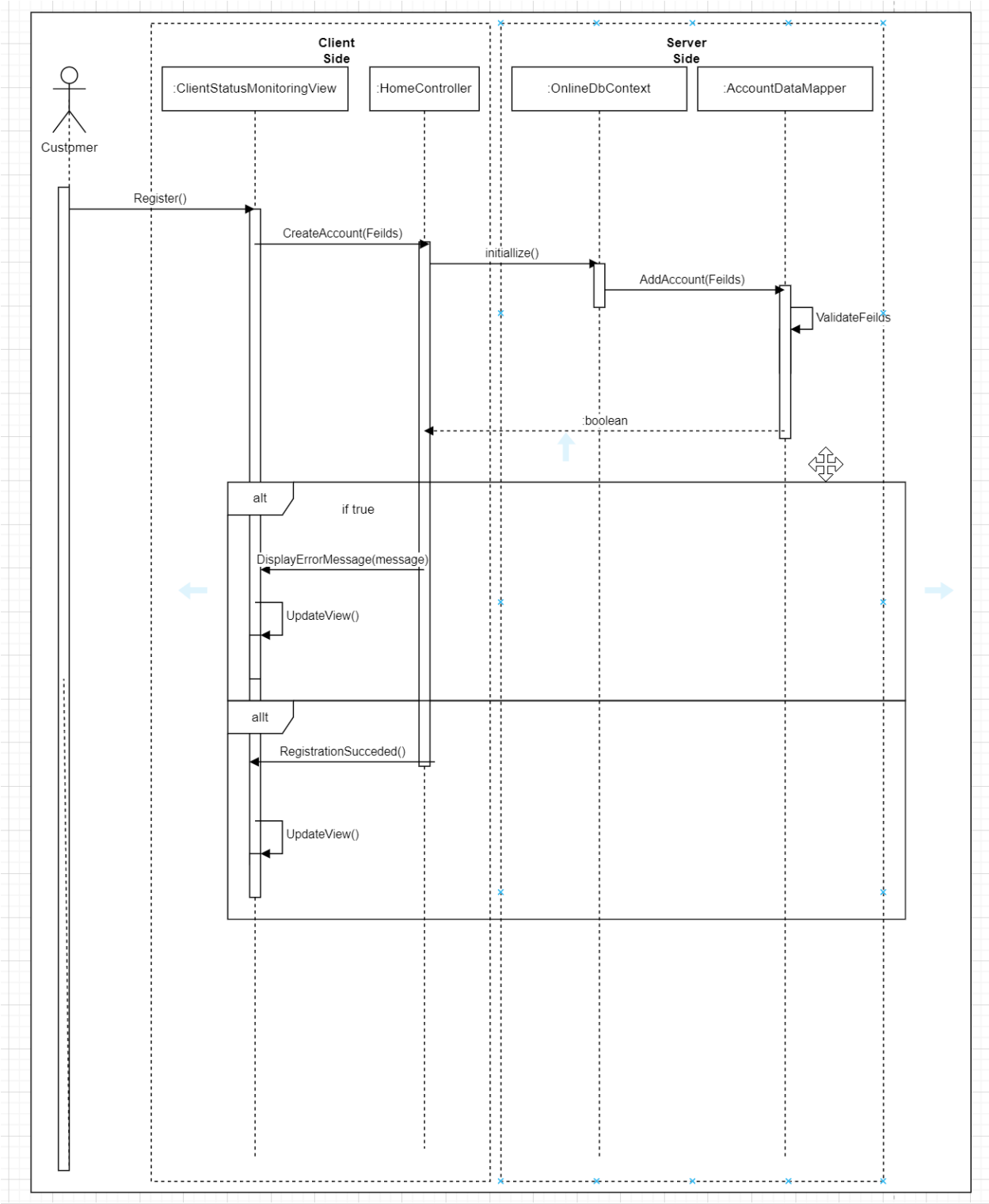
UpdateView()	ClientStatusMonitoringView	Used to Update the view of the user on screen
PostUpdatedFeilds()	ClientStatusMonitoringView	Used pass parameters of the update fields to the controller
DisplayErrorMessage(message)	HomeController	Responsible for passing the error message to be displayed on the View
Update View(Book)	Home Controller	An abstract method to be implemented by the ClientStatusMontitoringView
Initialize	HomeController	Used to initialize the database
UpdateBook(Fields)	OnlineDbContext	Responsible for updating the database using the given fields



Use Case for UC-3 (Key:UML)

From this interaction, initial methods for the interacting methods for the interfaces can be identified.

MethodName	**Element**	**Description**
Login()	ClientStatusMonitoringView	Used to pass the email and password of a user to the controller
RetrieveAccount()	HomeController	Used to Retrieve the account based on the email and password passed on from the view
Initialize()	HomeController	Used to initialize the dataBase
DsiplayErrorMessage	Home Controller	Used to pass the error message to the view
RetrieveAccount()	OnlineDbContext	Used to retrieve the account associated with the email and password



Use Case for UC-1(Key:UML)

From this interaction, initial methods for the interacting methods for the interfaces can be identified.

Method Name	Element	Description
Register()	ClientStatusMonitoringView	Used to pass to obtain credentials and pass to controller
Update View()	ClientStatusMonitoringView	Update the View Accordingly on screen for the user
CreateAccount(Fields)	HomeController	Used to pass the required fields to the database
Initialize()	HomeController	Used to initialize the database
DisplayErrorMessage()	HomeController	Used to pass the message to be displayed to the view
AddAccount(Fields)	OnlineDbContext	Uses the parameters passed on by the HomeController and use it to update the database
ValidateFields()	AccountDataMapper	Validates Fields before updating relation

Step 7: Sketch Views and Record Design Decisions

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions made during the iteration
		UC-1	Modules across the layers and preliminary interfaces to support this use case have been identified
		UC-3	Modules across the layers and preliminary interfaces to support this use

			case have been identified
		UC-5	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
	QA-1		The elements that support associated use case (UC-5) have been identified
	QA-2		The elements that support associated use case (UC-1) have been identified
	QA-4		The elements that support associated use case (UC-1, UC-3, UC-6) have been identified
	QA-5		The elements that support associated use case (UC-5) have been identified

	CON-3		Relationships for primary keys within the database have been identified
	CON-5		Logic for checking the number of copies within the database has been identified