OntarioTech UNIVERSITY

Project Deliverable 3

CRN: 43509

Group 7

Kalapan Kannathasan - 100759041

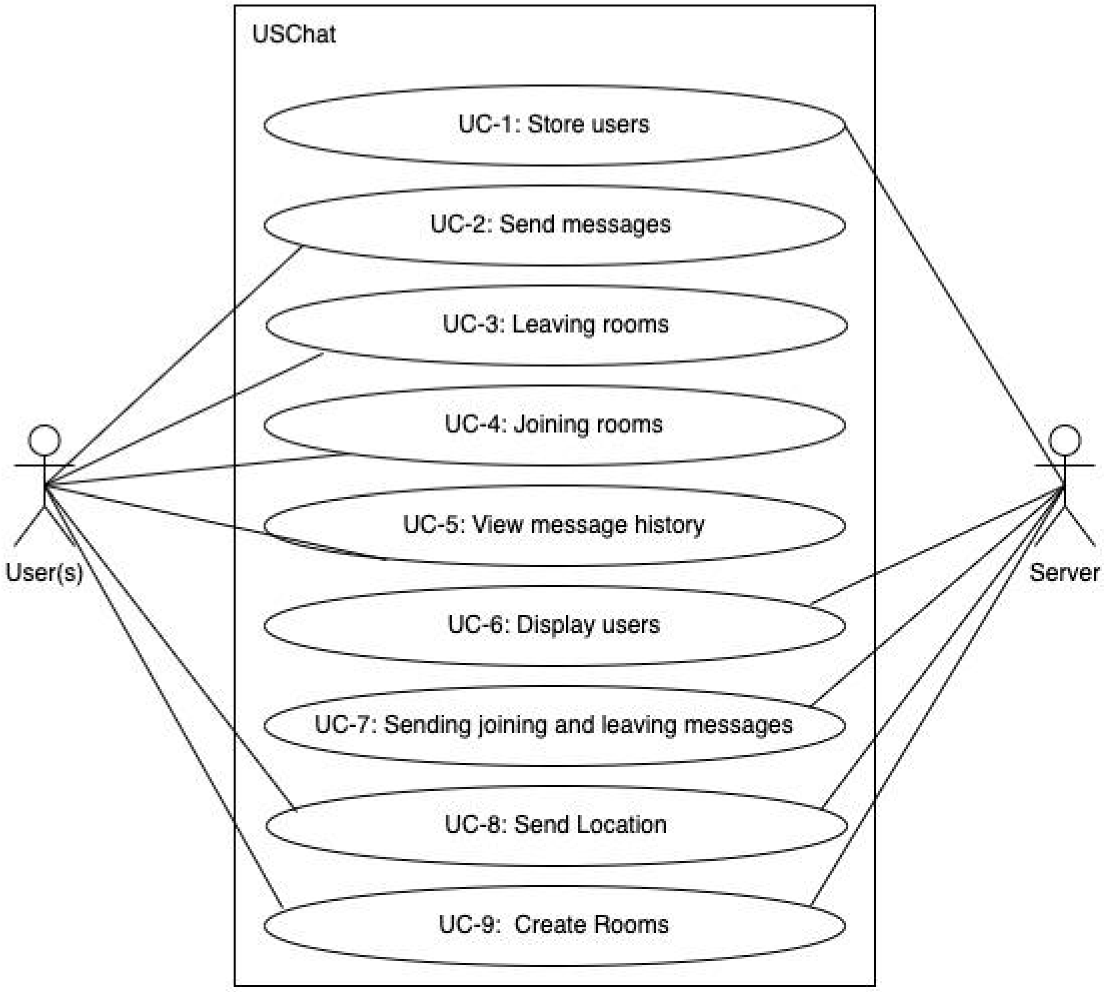
Yash Patel - 100746810

Sujeev Uthayakumar - 100744194

Zirak Mughal - 100749132

Date: November 19, 2021

Use Case Model:



1.1 Iteration 1: The Design Process

This is the first part of the design process, we must translate the requirements and quality attributes that were previously gathered into design decisions. These design decisions will be derived from figure 1.2, which is a list of our quality attributes. This is the beginning of making design decisions that will have further consequences, on the entire design of USChat.

1.2 ADD Step 1: Review Inputs

The first step of the ADD method involves reviewing the inputs and identifying which requirements will be considered as drivers. The inputs for USChat are summarized in the table below.

Category	Details
Design Purpose	The purpose is to build a sufficiently detailed design to support the construction of the USChat program.

Primary functional requirements	UC-1: Because it directly supports the core of the business UC-2: Because it directly supports the core of the business UC-6: Because it directly supports the core of the business UC-9: Because it directly supports the core of the business

Figure 1.1 *Primary functional requirements relevant to USChat*

ID	Quality Attribute	Scenario	Associated
QA-1	Scalability	Create a user and send it to be stored within the server, with an ever-expanding user base.	UC-1, UC-9
QA-2	Usability	User is able to send messages and the timestamps are recorded	UC-2
QA-3	Security	Send location coordinates but shifts it slightly	UC-8
QA-4	Performance	The ability to perform all tasks without comprising real-time functionality. Where messages are seen within a second of being sent.	All
QA-5	Availability	Users are able to see other users as well as the admin	UC-6

Figure 1.2 *Quality attributes relevant to USChat*

ID	Importance to the customer	Difficulty of Implementation According to the Architect
QA-1	Low	High
QA-2	High	Low
QA-3	High	Medium
QA-4	Medium	Medium
QA-5	Medium	High

Figure 1.3 *Quality attributes with an importance to the customer and the difficulty of implementation*

ID	Constraints
CON-1	A minimum of 10 users must be supported
CON-2	Messages should be sent in less than 1 second on either end

CON-3	Users should be authenticated before joining the room
CON-4	Network connection between user and server must have low bandwidth and be reliable
CON-5	The system must be accessed through a web browser such as (Chrome, Firefox Safari, etc)
CON-6	Large amount of messages will have to be stored in a single session
CON-7	An existing relation database server must be used. This server must hold the user and room information will be used to fetch information.

Figure 1.4 *Constraints for USChat*

ID	**Concern**
CRN-1	Establishing an overall initial system structure.
CRN-2	Leverage the team's knowledge about JSON technologies, including handlebars, and socket.io.
CRN-3	Allocate work to members of the development team.

Figure 1.5 *Architecture Concern for USChat*

2.1 Iteration 1: Establishing an Overall System Structure

The focus of this section is to present the results of the activities that are performed in each of the steps in iteration 1 of the ADD design process. Step 2 to till the end of step 7 will be focused on in iteration 1, where we will refine our requirements and quality attributes.

2.2 Step 2: Establish Iteration Goal by Selecting Drivers

This is the first iteration of the design of a USChat system, so the iteration goal is to establish the initial design of the system, where we will establish the overall system structure. This iteration will be driven by general architectural concerns, there will be a focus on all the drivers that may further influence the general structure of the system. In particular, we will focus on items within the:

- QA-1: Scalability
- QA-3: Security
- QA-4: Performance
- QA-5: Availability
- CON-3: Users should be authenticated before joining the room
- CON-4: Network connection between user and server must have low bandwidth and be reliable
- CON-5: The system must be accessed through a web browser such as (Chrome, Firefox Safari, etc)

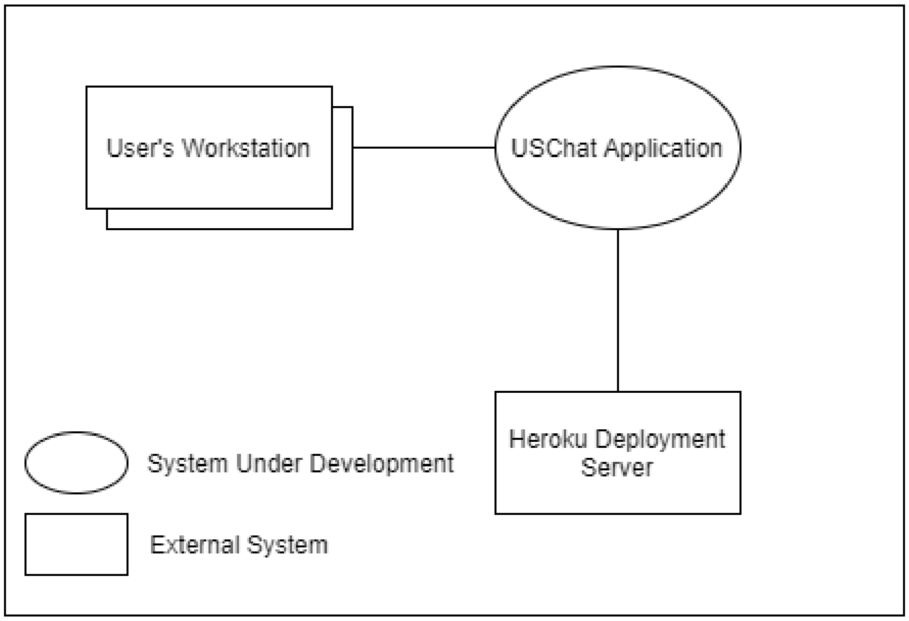


Figure 2.1 *Context diagram for the USChat system*

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

In this case, the element to refine is the entire USChat system. This refinement is performed through decomposition.

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

In this current iteration, the goal of structuring the entire system, design concepts are selected from the criteria presented in Section 1.1. The table below will summarize the selection of design decisions.

Design Decision and Location	Rationale
Logically structure the client part of the system using the **Web Application** reference architecture	The Web Application reference architecture supports the development of a web browser that communicates with the Heroku Development Server. This application supports web applications as the majority of it resides on the server, and its architecture is typically composed of three layers.

Discarded Alternatives:

Alternative	Reason for Discarding
Rich Client Applications	This reference architecture is oriented towards a local application that is stand-alone. This architecture will not support your system as it

	requires a server to allow users to use it concurrently. This was discarded due to the usage of local components.
Rich Internet Applications	This reference architecture is oriented towards a browser application, however, it is using asynchronous javascript and XML. This was discarded due to not using some of the supported languages.
Mobile Applications	This reference architecture is oriented towards handheld devices. This was discarded as this device was not considered for using and accessing the system.

Physically structure the application using the **Three-Tier Deployment** pattern	Since this system must be accessed from a web browser (CON-5) where (CON-3) and (CON-4).
Deploy the application using **Heroku Deployment**	Access to the application is obtained via a web browser (CON-5). This technology also facilitates (CON-1).

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

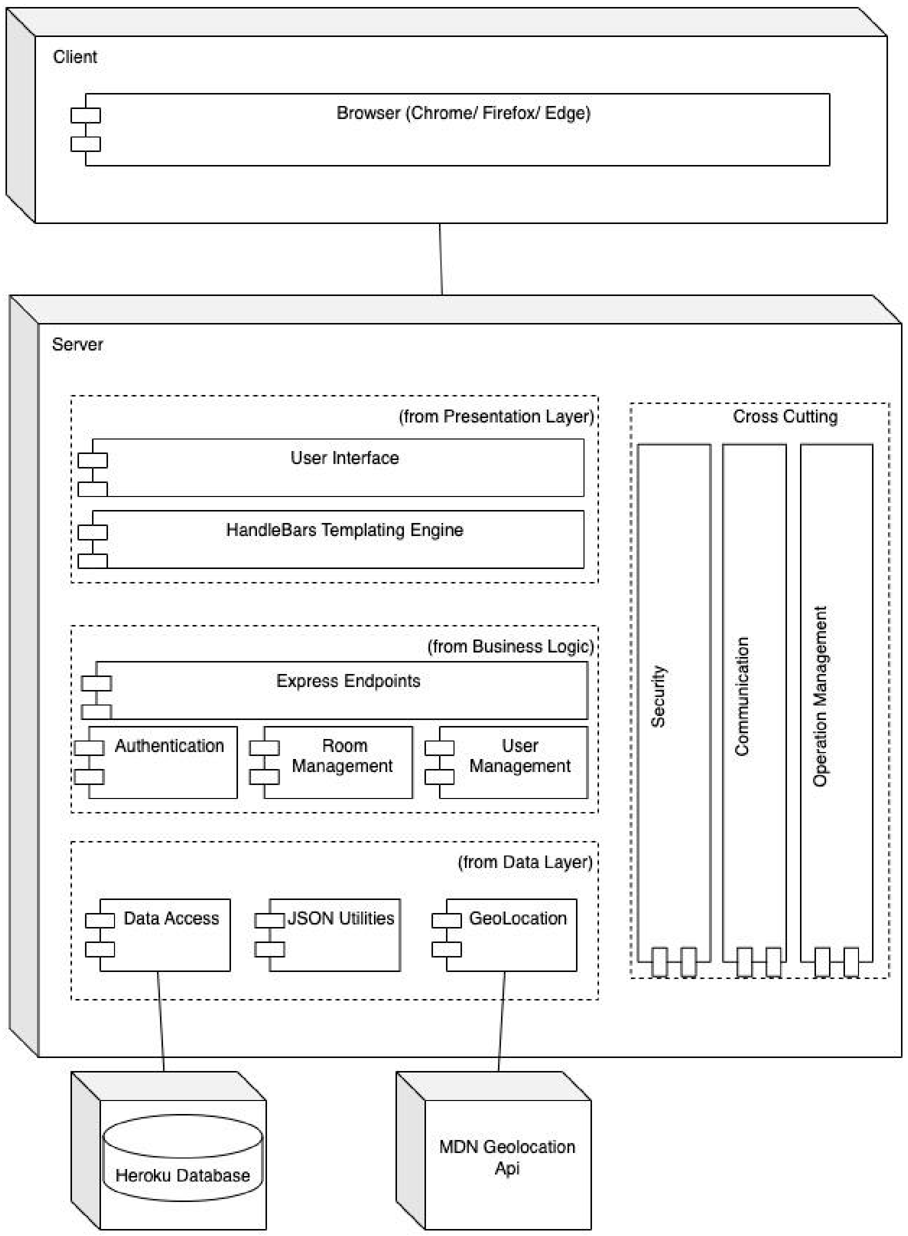
The instantiation design decisions considered and made are summarized in the following table:

Design Decision and Location	**Rationale**
Create a module to store user information and create a module to create rooms in the business layer within the **Web Application**	The service agent component from the reference architecture is updated to store the room and user information. This will facilitate QA-1 and will help with UC-1 and UC-9.
The client module within the **Three-Tier Deployment** pattern will be implemented using Handlebars for users	A component is added to the reference architecture to help GUI. This will facilitate QA-1 and QA-4 as well as will help with UC-6.

The data source module that is referenced in the **Web Application** will be implemented using Heroku as the main data source	A component from the reference architecture is updated to allow access to Heroku, a hosting application. Facilitates all quality attributes, as well as constraints.

The results of these instantiations decisions are recorded in the next step. In the initial iteration, it is typically too early to precisely define functionality and interfaces. The next iteration will cover how each will be defined in more detail.

2.6 Step 6: Sketch Views and Record Design Decisions



Element	Responsibility
Browser	A web browser running on the client machine.
User interface	These components are responsible for receiving user interactions and presenting information to users.
HandleBars	Templating engine, which takes information from server-side and prints it to client-side.
Express Endpoints	Express.js is used for endpoints, allowing to build a REST based project, it allows to define GET and POST methods. Also for the program, to send messages between requests and to send responses back to the client.
Authentication	Responsible for ensuring the user was created and registered within the database correctly.
Room Management	These components contain the ability to fix or change the room in any form and to display room information.
User Management	These components contain the ability to fix or change the user account in any form and to display user information.
Data Access	These components encapsulate the persistence mechanism and provide common operations used to retrieve and store information.
JSON Utilities	These components contain functionality common to the user and message but not specific to an entity.
GeoLocation	These components contain functionality to obtain location from the user and output a location that has been shifted for security.
Security	These components include cross-cutting functionality that handles security to the database information.
Operation Management	These components include cross-cutting functionality that handles management of the whole application
Communication	These components include cross-cutting functionality that handles communication mechanisms across layers and physical tiers.

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

The following table summarizes the design progress within iteration 1.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During this Iteration
	UC-1		Selected reference architecture establishes modules that will support

	this functionality.
UC-2	Selected reference architecture establishes modules that will support this functionality.
UC-6	Selected reference architecture establishes modules that will support this functionality.
UC-9	Selected reference architecture establishes modules that will support this functionality.
QA-1	Addition of modules within the reference architecture in the data layer to store new users. The details have not been defined.
QA-3	Supported with the inclusion of the Heroku development. But no further details were given.
QA-4	Introduction of three-tier deployment using handler bars. Further details will be provided in later iterations.
QA-5	Addition of data source using Heroku. But no further details were given.
CON-3	Support with the three developments.
CON-4	No relevant decisions were made.
CON-5	Selection of reference architecture architectures and deployment patterns.