

Software Requirements Specification

for

SOCIO - A Social Networking System

Version 2.3 approved

National Institute Of Technology Karnataka, Surathkal

February 21 , 2019

Table of Contents

Table of Contents	1
Revision History	1
1. Introduction	2
1.1 Purpose	2
1.2 Document Conventions	2
1.3 Intended Audience and Reading Suggestions	2
1.4 Product Scope	3
1.5 References	3
2. Overall Description	4
2.1 Product Perspective	4
2.2 Product Functions	4
2.3 User Classes and Characteristics	5
2.4 Operating Environment	5
2.5 Design and Implementation Constraints	9
2.6 User Documentation	9
2.7 Assumptions and Dependencies	9
3. External Interface Requirements	10
3.1 User Interfaces	10
3.2 Hardware Interfaces	10
3.3 Software Interfaces	11
3.4 Communications Interfaces	12
3.5 IDE Interfaces	12
4. System Features	13
5. Other Nonfunctional Requirements	19
5.1 Performance Requirements	19
5.2 Safety Requirements	19
5.3 Security Requirements	19
5.4 Software Quality Attributes	19
5.5 Business Rules	20
6. Other Requirements	20
Appendix A: Glossary	21
Appendix B: Analysis Models	16
Appendix C: To Be Determined List	16
Project Budget Estimation	16

Revision History

Name	Date	Reason For Changes	Version
Shreyas	16-01-19	Created the SRS	1.1
Suyash	16-01-19	Added Interface requirements	1.2
Shreyas	23-01-19	Updated System Features	2.1
Suyash	23-01-19	Updated Non-functional Req.	2.2
Shreyas	21-02-19	Added DFD and use case diagram	2.3

1. Introduction

1.1 Purpose

The SRS will provide a detailed description of the requirements for Socio - A social networking system. This SRS will allow for a complete understanding of what is to be expected from the newly introduced system which is to be constructed. The clear understanding of the system and its functionality will allow for the correct software to be developed for the end user and will be used for the development of the further stages of the project.

1.2 Document Conventions

The document is prepared using Google Docs and has used the font type 'Times New Roman'. The font size that has been used to type this document is 14pt for the headings and 11pt for the corresponding body. Standard IEEE template is the template used to recognize the appearance of the document and its flow.

1.3 Intended Audience and Reading Suggestions

This document is made by keeping in mind different types of readers. This document will be useful for different audience in various ways.

Audience	Use
Developers	They will use this document as a guidance for design and implementation phase.
Managers	They will see all the constraints are covered properly. Time and cost is within limits or not.
Marketing Staff	They can use this document to make advertisements for this android app because by reading this document they will know what the system will do? How this system is different from others.
User	By reading the SRS they can ensure whether their needs are being met by the App or not.

Testers	They will test the implementation of the project according to the SRS base.
Documentation Writer	They will use this document during the documentation of the project.

1.4 Product Scope

- The System developed will enable the users to socialize over the internet. The users can make friends, chat with them, create posts to share among friends, like or comment on posts, create groups and have group chats and also share files. Since everyone is leading a busy and stressful life, socializing means a lot to everyone.

1.5 References

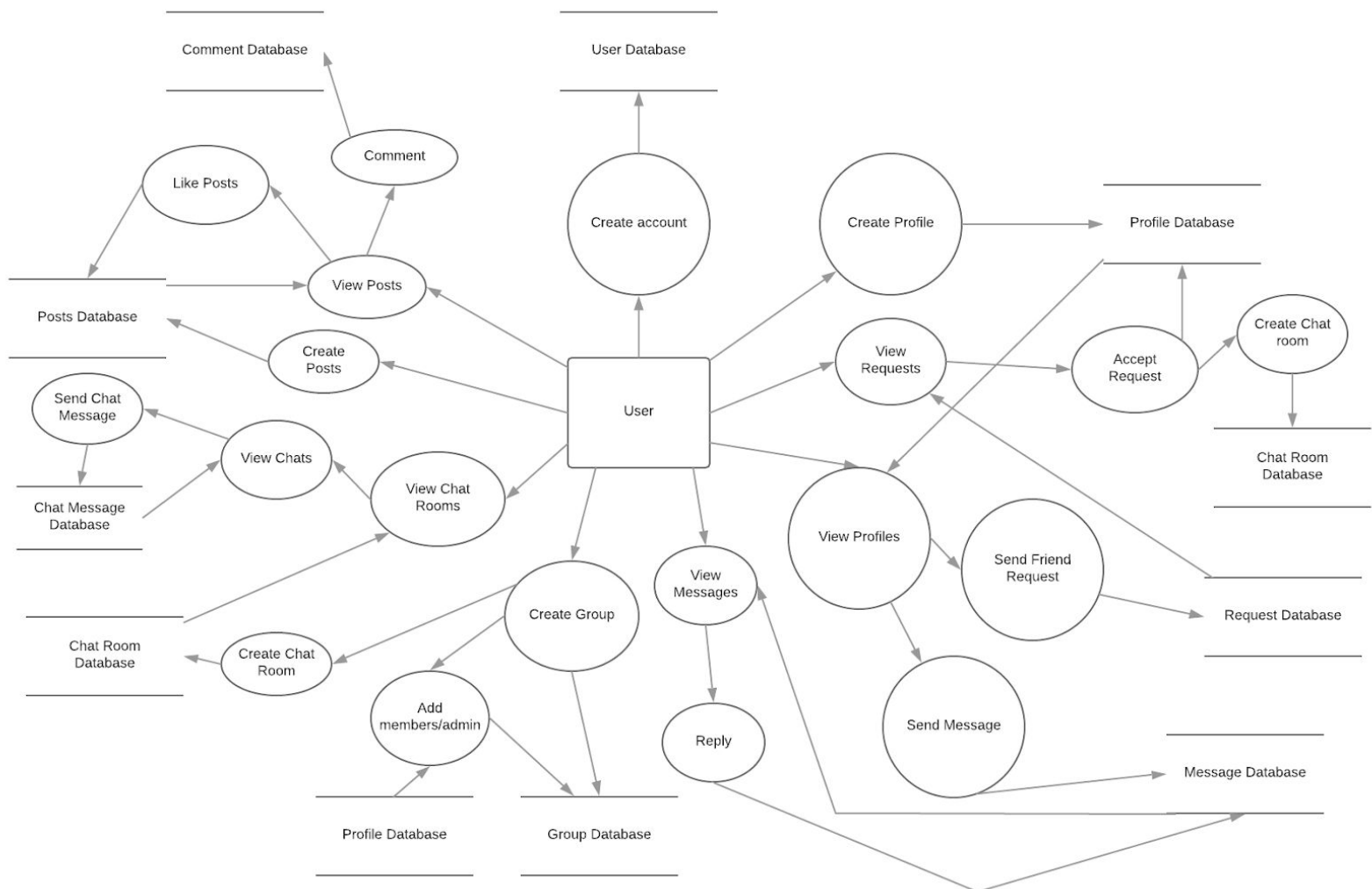
- IEEE. IEEE Std. 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

2. Overall Description

2.1 Product Perspective

The SOCIO - A Social Networking System project is a new, self-contained product intended for use on the Web platform. Socio App enables us to interact with other users through chat, posts, comments, messages, groups and many more.

2.2 Product Functions



The above DFD clearly shows various processes associated with the project and how data flows between an entity and database via a specific process

User registers his account and creates/updates profile.

User views different profiles and sends friend requests to make new friends or messages/shares files with existing friends.

User views received friend requests and accepts or declines. A new chat room is created if accepted.

User views received messages and replies back.

User creates group, add/remove his friends to it and grant/take adminship. A new chat room is created/updated for the group.

User selects chat room and views previous chat send new chat messages in that chat room.

User posts images/files on his profile.

User views posts of friends on the home page and can like and comment on the post.

2.3 User Classes and Characteristics

User	Characteristics
System Admin	It will be the system administrator. He will maintain the overall App.
Users	It includes the people who use the social networking app.

2.4 Operating Environment

Socio is the software application, which will be limited to a web application. The application is not resource- or graphics-intensive, so there are no practical hardware constraints. The app will rely on several functionalities built into Django's framework and API, so ensuring appropriate usage of the API will be a major concern.

The system shall be deployed on a Heroku platform which is a service providing platform to host with Ver. 5.7.21 MySQL Database to maintain the databases. The system shall be accessed on multiple browsers e.g. Google Chrome, Mozilla Firefox etc. The Operating System on the front-end PCs/Laptops can be MS Windows, Unix, Linux or Apple Mac.

Sr No	System	Environment for development	Description

1.	Software used to develop	Backend: Django 2.0	Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.
		Socket Server : NodeJS, Express	Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.
		Database: MYSQL	MySQL is an open-source relational database management system(RDBMS).
		Frontend: HTML, JavaScript, jQuery, AngularJS	<p>HTML: is the standard mark-up language for creating web pages and web applications</p> <p>JavaScript: As a multi-paradigm language, JavaScript supports event driven functional, and imperative (including object-oriented and prototype based) programming style.</p> <p>jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML</p> <p>AngularJS: The AngularJS framework works by first reading the HTML page, which has additional custom tag attributes embedded into it. Angular interprets those attributes as directives to bind input or output parts of the page to a model that is represented by standard JavaScript variables</p>

		Styling: CSS, Bootstrap	<p>Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a mark-up language.</p> <p>Bootstrap is a free and open-source front-end web framework for designing websites and web applications.</p>
		IDE: PyCharm (student version)	It provides code analysis, a graphical debugger, an integrated unit tester, integration with version-control systems (VCSes), and supports web development with Django.
		Version Control: Git, GitHub	GIT is a version-control system for tracking changes in computer-files and coordinating work on those files among multiple people.
		Text Editor: - Sublime	Sublime Text is a cross-platform source code editor with a Python Application programming interface(API)
2.	Server hosting/ Installation (user testing)	<p>Heroku</p> <p>If bugs are found again, coding and unit testing tools will be used along with latest version of software from Git.</p>	Heroku is a cloud Platform as a Service (PaaS) supporting several programming languages. that is used as a web application deployment model.

2.5 Design and Implementation Constraints

The primary design constraint is the web platform. Since the application is designated for any device which can access the web, adaptive screen size and resolution will be a major design consideration. Creating a user interface which is both effective and easily navigable will pose a difficult challenge.

Moreover, since security is a major issue, passwords of the registered users need to be hashed when storing in the database. Also, the privacy of the users needs to be maintained. Only the user who is friends with another user should have the associated rights and not others. Only the group admins should have authority over the groups. Only users of the specific chat room can put up messages in it. User can like a post only once. Any user shouldn't be able to change other's profiles.

2.6 User Documentation

- A brochure will be provided describing the functionality of the Web App.

2.7 Assumptions and Dependencies

- The app requires Django framework and REST API.
- A NodeJs server will be required which will be a socket server for real time chats and posts.
- The project requires a host and server, heroku account to publish the app.

3. External Interface Requirements

3.1 User Interfaces

- The Web application will have an initial login page where users would enter their respective login credentials. It would also have an option to allow new users to Sign up.
- The Sign up page would prompt the user to enter email id, username and password.
- Logging in would land the user on the Home Page. Posts would be displayed on the Home page. There would also be an option for the user to create new posts.
- Home page would have a navigation bar having options to view User Profile, Find Friends, Chat, Logout, etc. Also there is a form for creating posts.
- The posts will have an option to like/dislike and a form for commenting. If one's own post, an option to edit/delete the post will be available.
- The chat page would show chat history with the respective person/group and a form to send a chat message.
- The profile page would have an option of sending/cancelling a friend request to that person if he/she is not yet a friend or it would have an option of sending a message or unfriending that person if he/she is a friend.

3.2 Hardware Interfaces

- The System shall be deployed on Heroku Platform. All the Stakeholders are supposed to login into the Socio website where there will be a specific URL to access the System. Hardware Requirements for stakeholders.
- Pentium 4 processor or higher

- Approximately 100 MB of free harddrive space
- Minimum 128 MB RAM

Hardware Requirements for hosting:

- Minimum 1GB database space
- Minimum 2GB RAM
- Wearable devices will not be supported with this application.

3.3 Software Interfaces

Software Requirements for Hosting:

- Django 2.0
- MySQL

Software Requirements for Stakeholders:

- Browser (Google Chrome, Mozilla Firefox, Safari etc.)
- Operating System supporting the above browsers.

3.4 Communications Interfaces

The System will be operational using standard web-browsers like Safari, Google Chrome and Firefox. Users will connect- through a secured encrypted connection over internet. Since the data communicated over internet is confidential it is imperative that encrypted protocols are used to prevent data leakages.

3.5 IDE Interfaces

- IDE Pycharm for development using Django Framework.

4. System Features

4.1 User Signup

Use Case Name	User Signup
Actor	User
Overview	This use case is about registration of the user in.
Pre-condition	The internet connection is working. Server is not down.
Post-condition	The credentials entered are within constraints.

4.2 User Login

Use Case Name	User Login
Actor	User
Overview	This use case is about authorisation of the user.
Pre-condition	The user is registered to the database.
Post-condition	The credentials entered are correct..

4.3 Create/Edit Profile

Use Case Name	Create/Edit Profile
Actor	User
Overview	This use case is about profile creation/updation by the user.
Pre-condition	The user is logged in.
Post-condition	The profile is created/updated.

4.4 View Profiles

Use Case Name	View Profiles
Actor	User
Overview	This use case is about viewing different profiles.
Pre-condition	The user is logged in.
Post-condition	None

4.5 Send Friend Requests

Use Case Name	Send Friend Requests
Actor	User
Overview	This use case is about sending friend requests to different users.
Pre-condition	The user is logged in. The profile user is not user's friend already.
Post-condition	A request is generated with sender and receiver fields.

4.6 Accept Friend Requests

Use Case Name	Accept Friend Requests
Actor	User
Overview	This use case is about accepting friend requests to different users.
Pre-condition	The user is logged in. The profile user is not user's friend already.
Post-condition	The profiles are added to each other's friend list respectively. A new chat room is created with the users as its members.

4.7 Send Messages/Files

Use Case Name	Send Messages/Files
Actor	User
Overview	This use case is about sending messages/files to friends.
Pre-condition	The user is logged in. The profile is present in user's friend list.
Post-condition	New message is created with sender and receiver fields.

4.8 Reply Messages/Files

Use Case Name	Reply Messages/Files
Actor	User
Overview	This use case is about replying to messages/files received.
Pre-condition	The user is logged in. The user must have received a message/file.
Post-condition	New reply message is created with sender and receiver fields.

4.9 Create Groups

Use Case Name	Create Groups
Actor	User
Overview	This use case is about creating a group. The user can then add/remove members and grant/take away adminship.
Pre-condition	The user is logged in. The user has an active profile.
Post-condition	A group is created with the user as its admin. A new chat room is created for the group.

4.10 Add Members to the Group

Use Case Name	Add Members to the Group
Actor	User
Overview	This use case is about adding members to a group.
Pre-condition	The user is logged in. The user is an admin of the group. The member to be added is present in the user's friend list.
Post-condition	The member is added to the group. The members are added to the group's chat room.

4.11 Remove Members from the Group

Use Case Name	Remove Members from the Group
Actor	User
Overview	This use case is about removing members from a group.
Pre-condition	The user is logged in. The user is an admin of the group. The member to be removed is the group's member.
Post-condition	The member is removed from the group. The member is removed from the group's chat room.

4.12 Add Admins to the Group

Use Case Name	Add Admin to the Group
Actor	User
Overview	This use case is about adding admins to a group.
Pre-condition	The user is logged in. The user is an admin of the group. The member to be given admin rights is the group's member.
Post-condition	The member is given admin rights of the group.

4.13 Remove Admins from the Group

Use Case Name	Remove admins from the Group
Actor	User
Overview	This use case is about removing admins from a group.
Pre-condition	The user is logged in. The user is an admin of the group. The member to be removed as admin is one of the group's admin.
Post-condition	The admin rights of the member are taken away.

4.14 Create Posts

Use Case Name	Create Posts
Actor	User
Overview	This use case is about creating posts. The posts can be posted in one's profile or in a group.
Pre-condition	The user is logged in. The user has an active profile.
Post-condition	New post is created.

4.14 Like/Unlike Posts

Use Case Name	Like/Unlike Posts
Actor	User
Overview	This use case is about liking/unliking posts.
Pre-condition	The user is logged in. The user has an active profile. The user can like/unlike only once. He can toggle between like/unlike.
Post-condition	For like, the user is added to the list of users who like the post. For unlike, the user is removed from the list of users who like the post.

4.15 Comment on Posts

Use Case Name	Comment on Posts
Actor	User
Overview	This use case is about commenting on posts.
Pre-condition	The user is logged in. The user has an active profile.
Post-condition	A comment is created for that post.

4.16 Chat

Use Case Name	Chat
Actor	User
Overview	This use case is about chatting with other users or a group of user.
Pre-condition	The user is logged in. There must be a chat room of which the participants of the chat are members of.
Post-condition	A Chat message is created for that chat room.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

1. Scalability

System should be able to handle a large number of users. For e.g. handling around Thousand users at the same time.

2. Speed

The application should be fast. It should not slow down with increase in the number of users. Search functionality should be fast to enable better end-user experience. The system should be quick enough to be able to respond to the user actions within a short period of time. e.g. the search user functionality of the site should perform a quick search among the users on the database.

5.2 Safety Requirements

There are no specific Safety Requirements.

5.3 Security Requirements

- During user registration, the given email address is validated.
- The password should be at least 8 characters containing at least a small character and a number and a special character.
- Password is stored as a hash value in database.
- We are transferring all data via HTTPS i.e. via SSL so that the data is encrypted during the transit. Thus safeguarding the user information.

5.4 Software Quality Attributes

1. Usability

User interface should be simple and clear to be understood by any user.

2. Availability

- The system should be available at all times. It should be ensured that there should be minimum or no downtime to ensure better user experience
- The system should be reliable. It should yield correct results if a user performs search for a person. Also, if the user sends message, the system should ensure

that the correct message is delivered to the correct destination without any loss or change in content.

3. Maintainability

The system should be developed in such a way that it is extensible. It should be easy to incorporate new features requirements or accommodate a change in the existing requirements.

5.6 Business Rules

1. All Users shall access the system using a login/user-id and password. The login-id/password will be managed in a secured manner.
2. A User cannot add friends, create groups or post without creating a profile.
3. A User can send a message or a file to another user only if he is in the user's friend list.
4. A User can add another user as a group member only if he is in the user's friend list and the user is one of the group's admin.
5. A User can alter adminship of a group member only if he is one of the group's admin.

6 Other Requirements

The system data shall be backed up every night (full back-up) with a cycle of 30 days. This essentially means that there will be a provision to rollback by a month. Post back-up everyday the back-up shall be restored on a dummy production system to ensure completeness and correctness of back-up. Post that the dummy production database shall be purged.

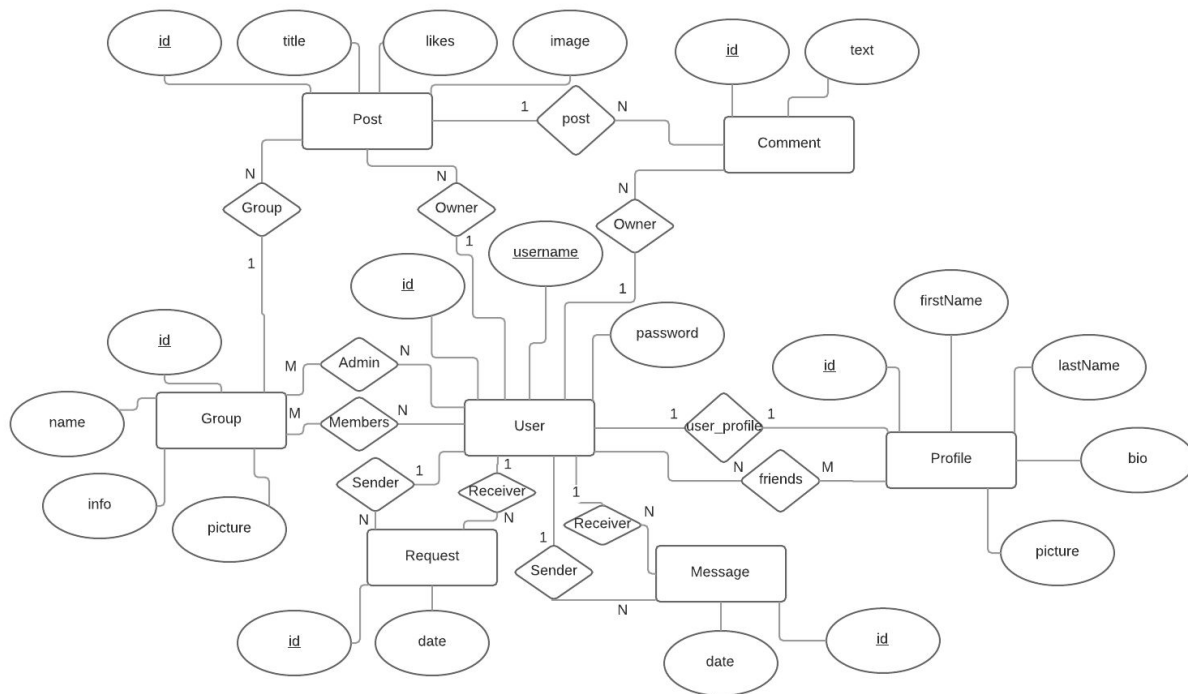
Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

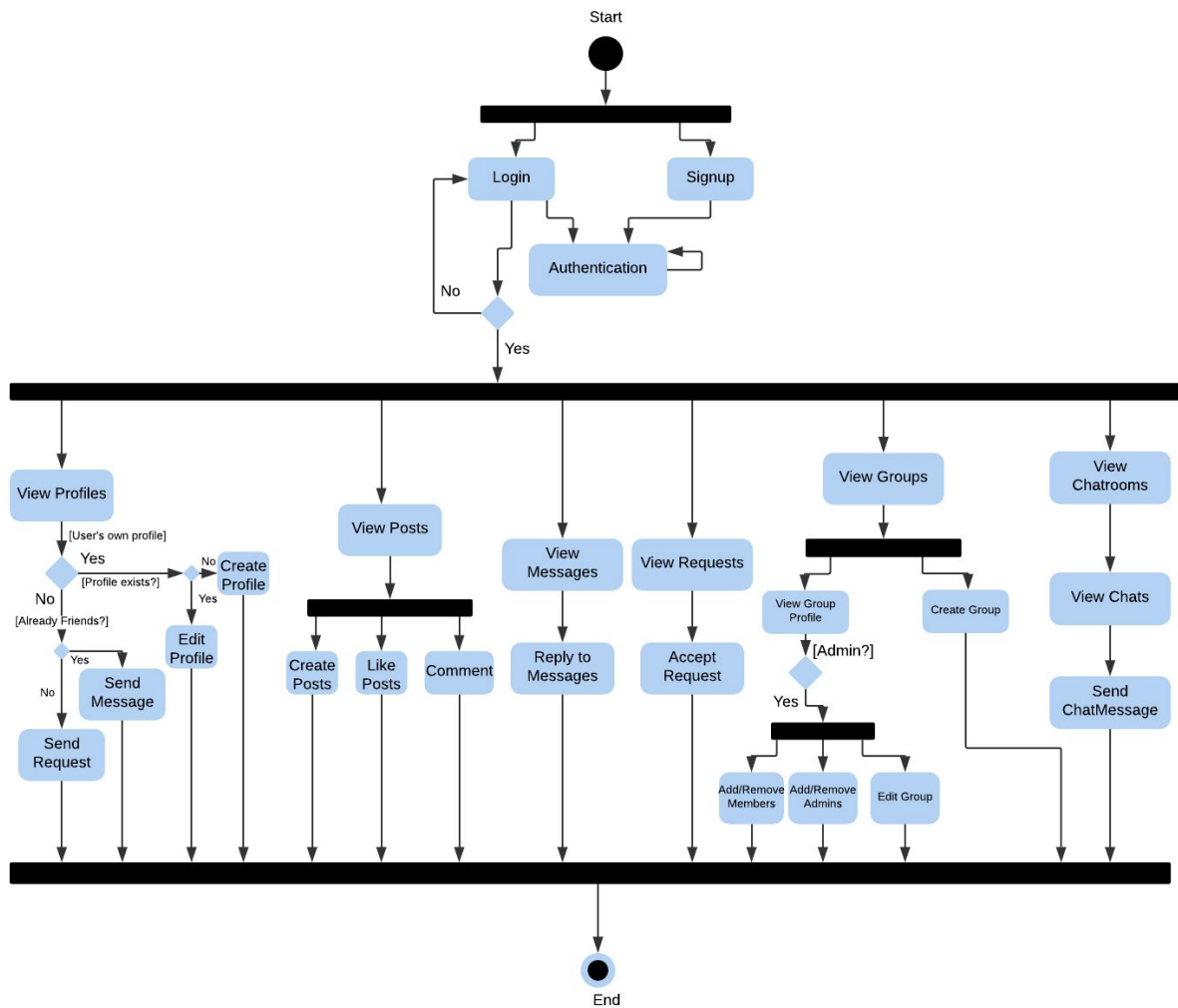
Appendix B: Analysis Models

Use Case Diagram:

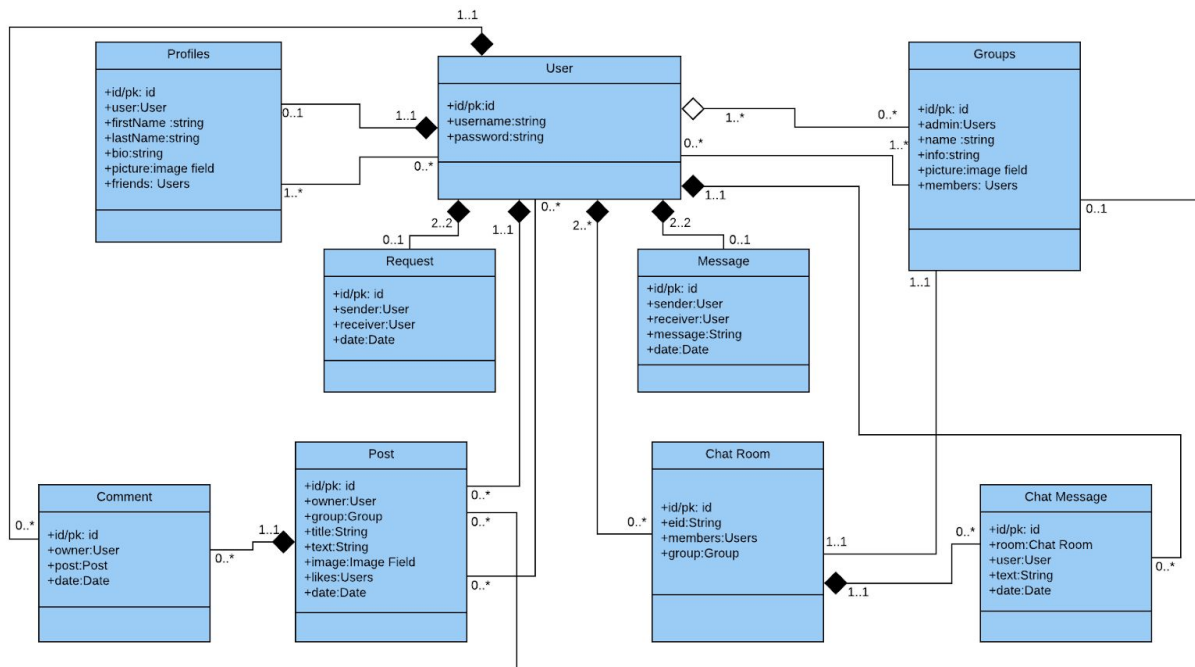


ER Diagram:

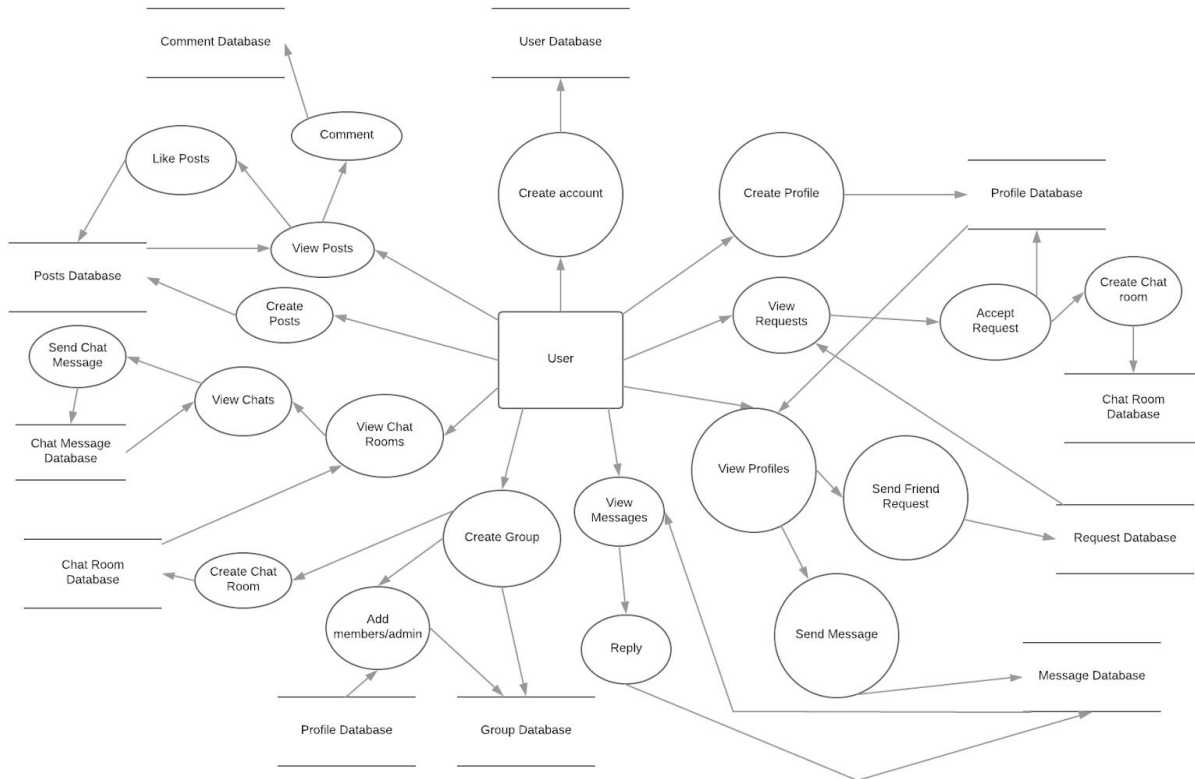
Navigation Diagram



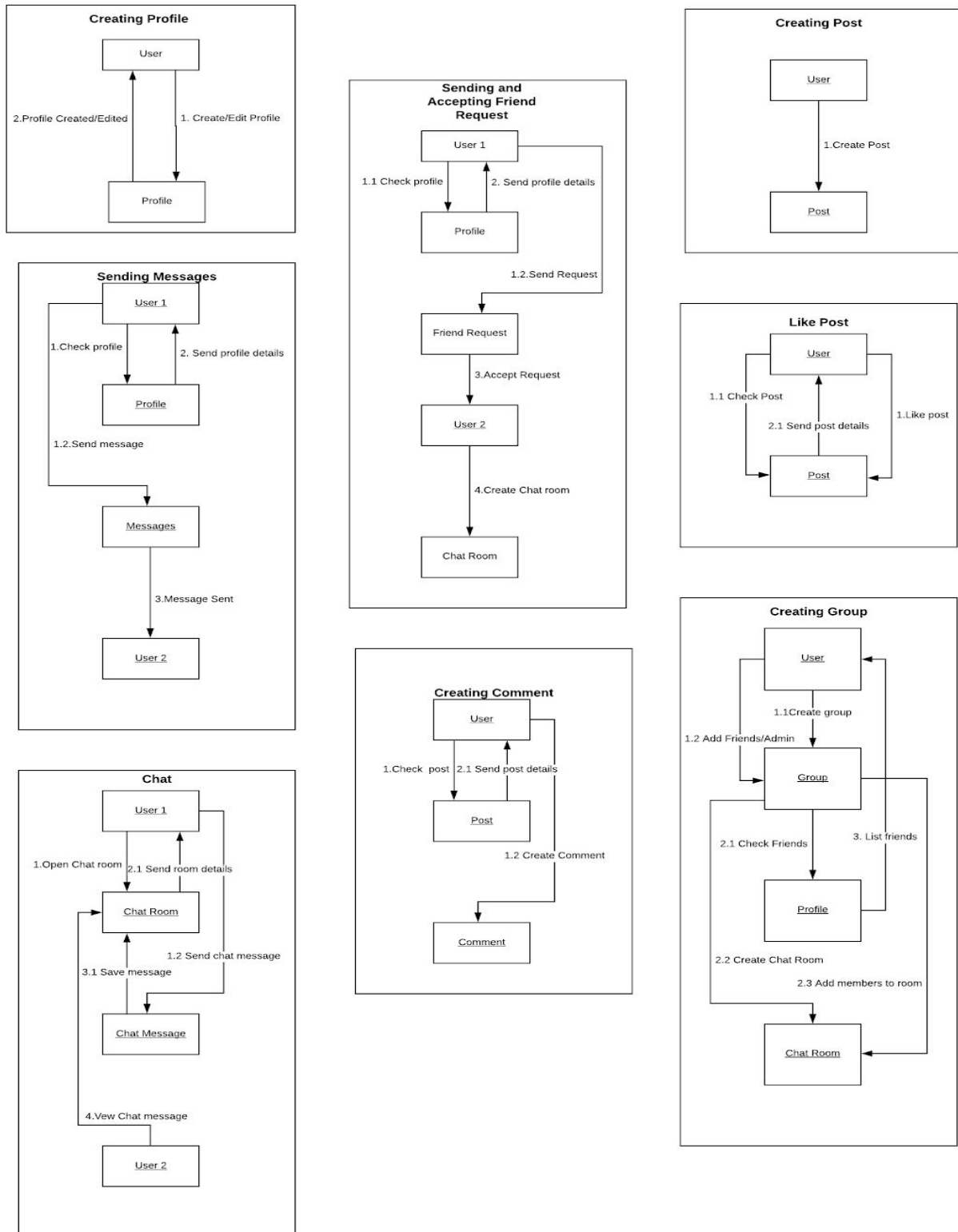
Class Diagram:



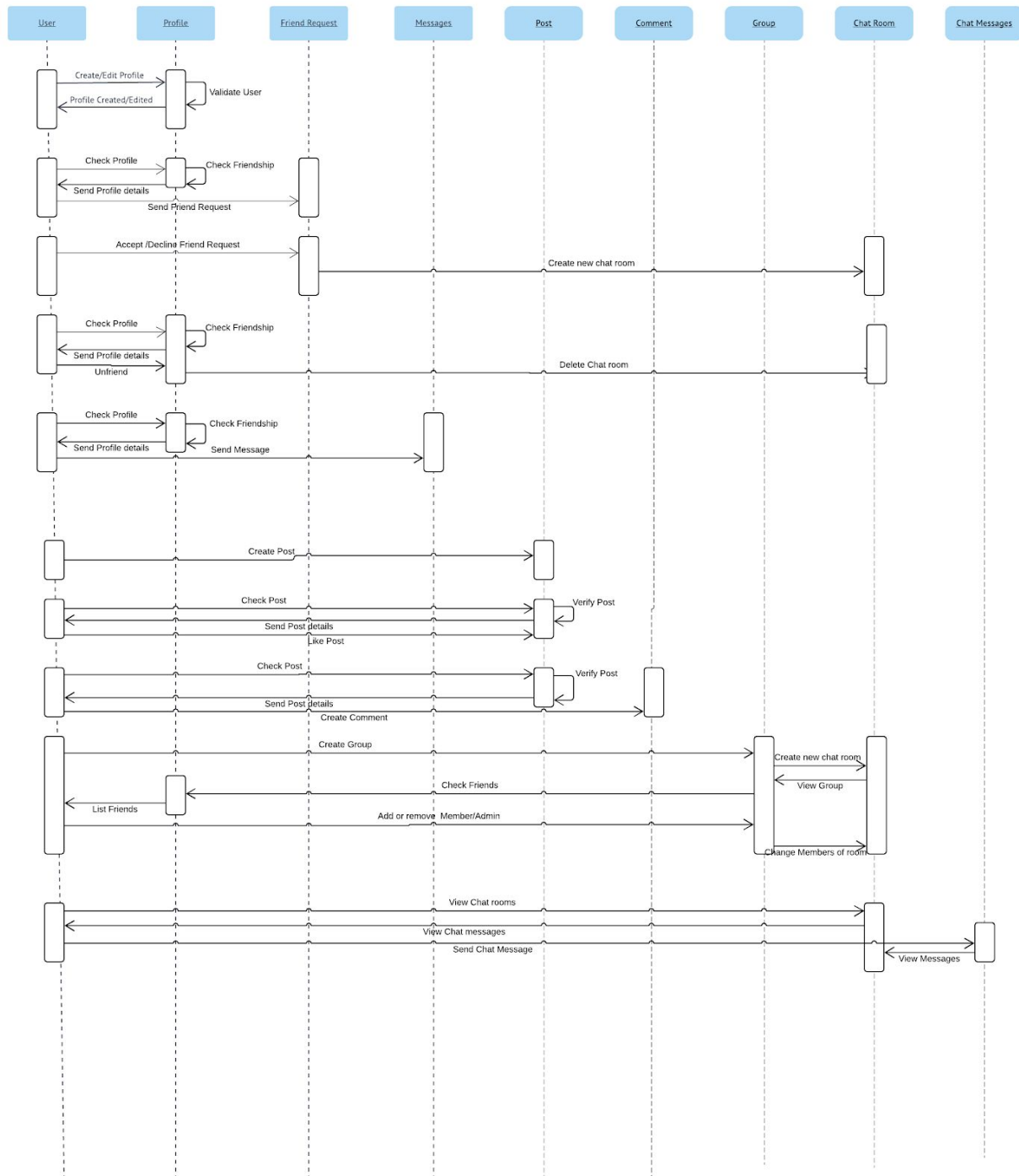
Data Flow Diagram:



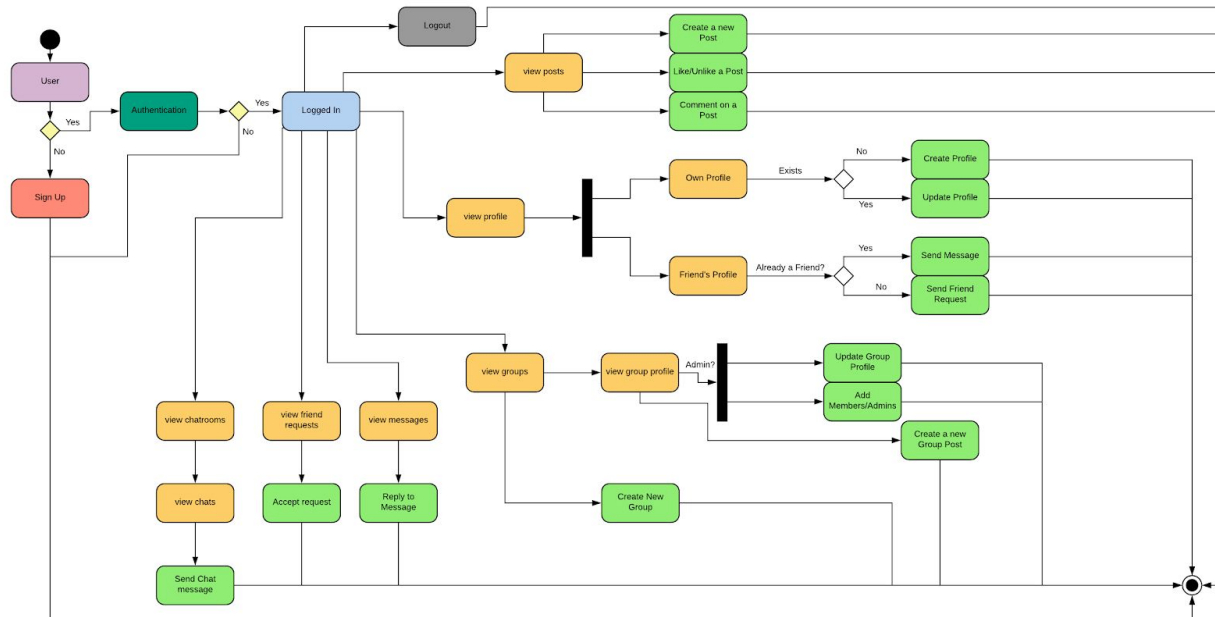
Collaboration Diagram:



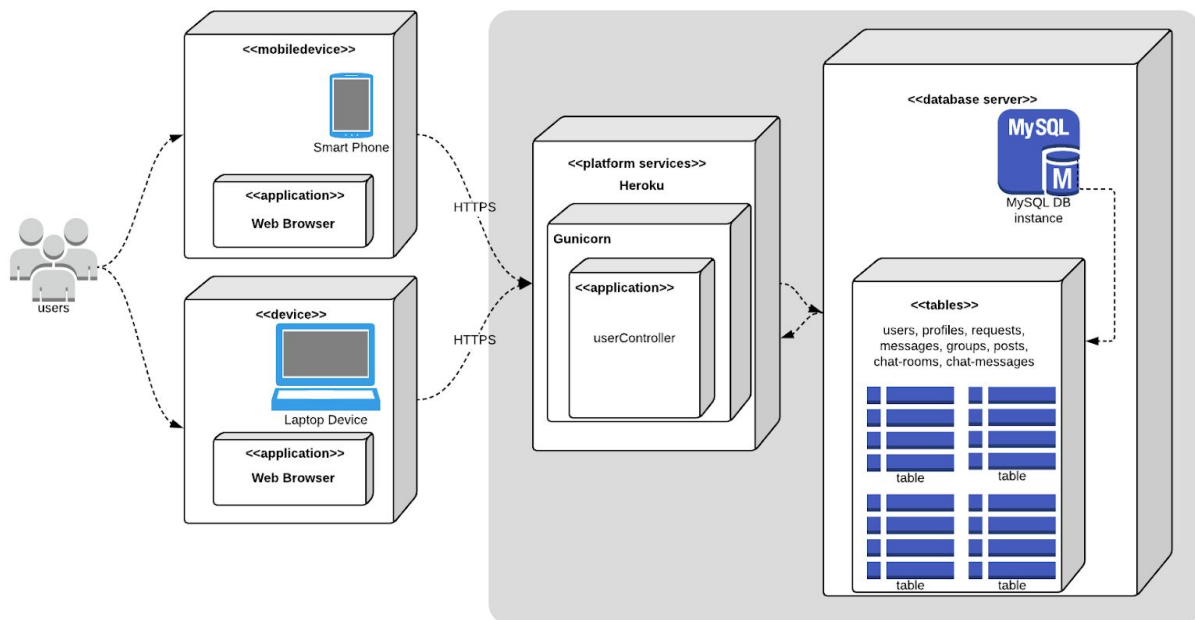
Sequence Diagram:



State Diagram:



Deployment Diagram:



Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>