The University of the West Indies

Department of Computing and Information Technology

Undergraduate Project Course | **INFO 3604**

Project Implementation Document

Project Name: **QuickRental**

Members

Michael Ali	814005475
Ishmael Mohammed	816002470
Kady Seecharan	816002036
Gabriela Sewdhan	813001696

Table of Contents

Project Implementation

1. Architectural Design

The Architectural Design includes the ***Context Model***, ***Architecture Diagram*** and ***Component Diagram***. These diagrams explains how the components of QuickRental are connected and work together.

1.1 Context Model

The Context Model includes the main systems for QuickRental as shown below.

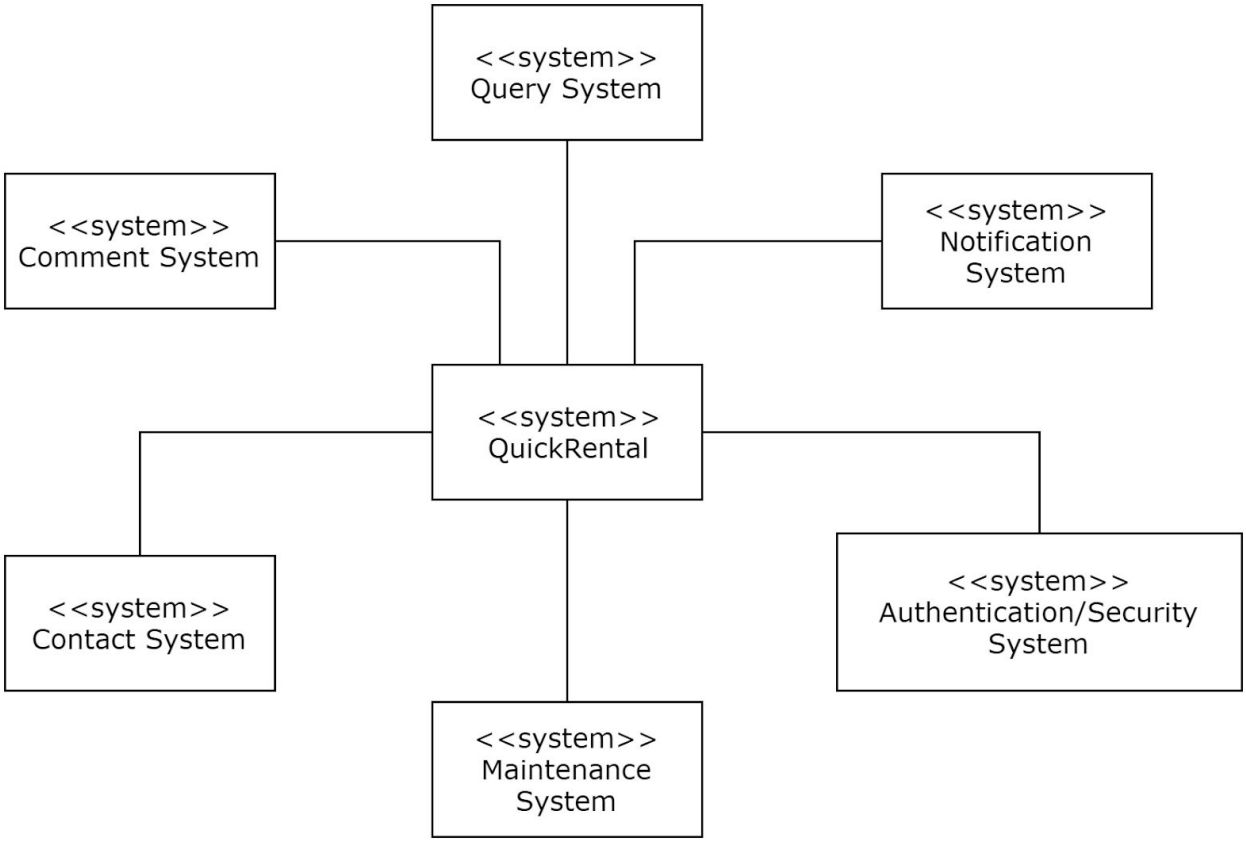

Figure 1: *Context Model for QuickRental*

1.2 Architecture Diagram

The following diagram shows the overall architecture and flow of information between the physical components of the QuickRental System. The communication paths and varying input data is displayed.

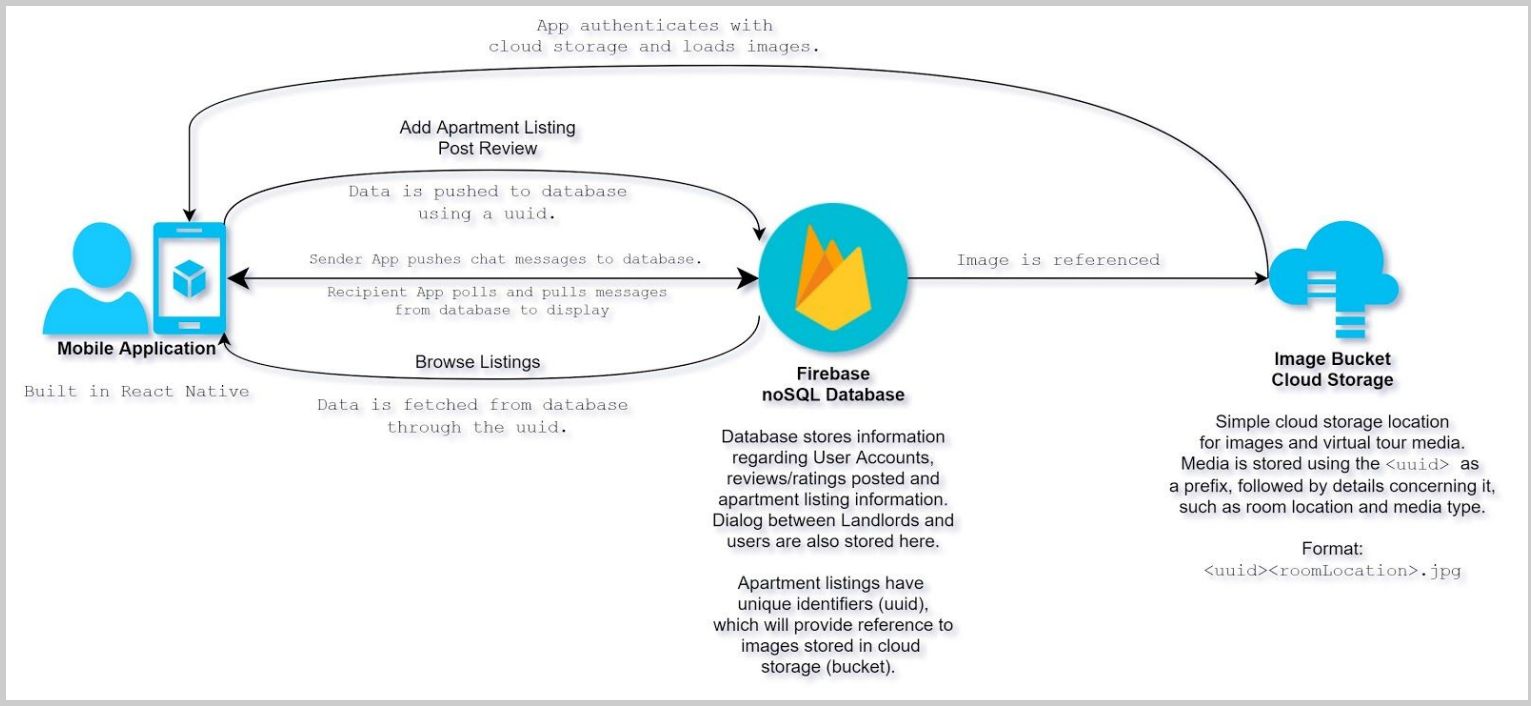

Figure 2: *Architecture Diagram for QuickRental*

1.3 Component Diagram

This diagram shows how all the components of the QuickRental System are connected.

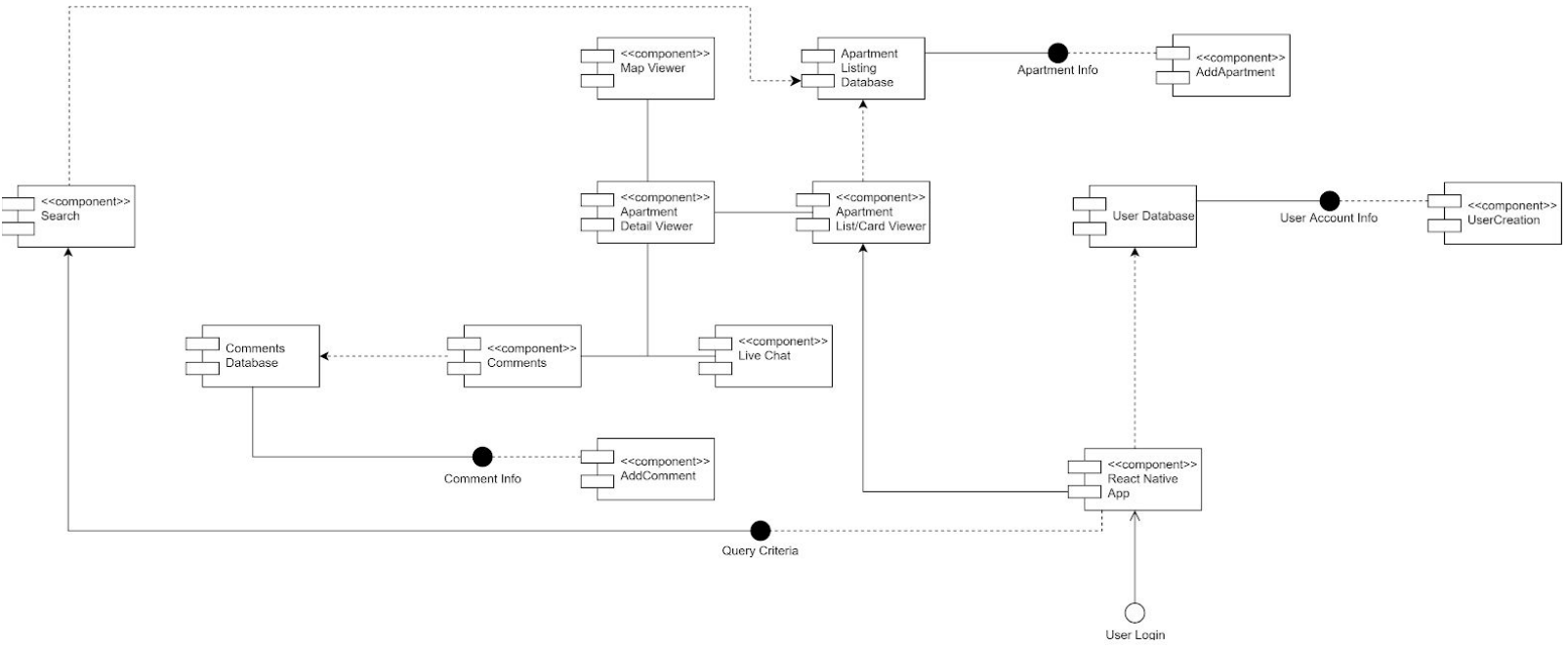

Figure 3: *Component Diagram for QuickRental*

1.3.1 Components Description

1. ***React Native App*** : Interface by which users can login and view the apartment listings, as well as create an account, which is required to access the app.
2. ***User Database*** : Stores the records (Name, Email, Contact etc.) of all users who have created an account using the ***User Creation*** component.
3. ***User Creation*** : Component that enables users to create an account to login and access the app.
4. ***Apartment List/Card Viewer*** : Screen displaying list of apartments as stored in the ***Apartment Listing Database*** *in the form of cards or lists*.
5. ***Apartment Listing Database*** : Contains the records of apartments as created by users using the ***AddApartment*** component.

6. ***AddApartment*** : Component allowing users with the required permissions (*landlords*) to create apartment records that will be stored in the ***Apartment Listing Database***.
7. ***Search*** : Component that allows users to select varying criteria and text input to perform a query of the database that will then be viewable using the ***Apartment List Viewer***.
8. ***Apartment Detail Viewer*** : Screen displaying the details (features, photos, map, virtual tour, reviews, landlord's contact information) of each apartment stored in the ***Apartment Listing Database***.
9. ***Comments***: Allows users to view Comments created by users using the ***AddReview*** component.
10. ***Comments Database*** Stores the records of the Comments (comment body, user ID) created by users through the ***AddComment*** component.
11. ***AddComment*** : Enables users to create comments that will be added to the ***Comments Database*** and viewable on the ***Comments*** component.
12. ***Map Viewer*** : Component allowing an interactive map to be displayed on a screen of the ***React Native App***, displaying the location of the current apartment and estimated walking distance.
13. ***Live Chat*** : Interface by which users can interact via direct messages and images with the Landlord in real-time.

2. Class Diagram

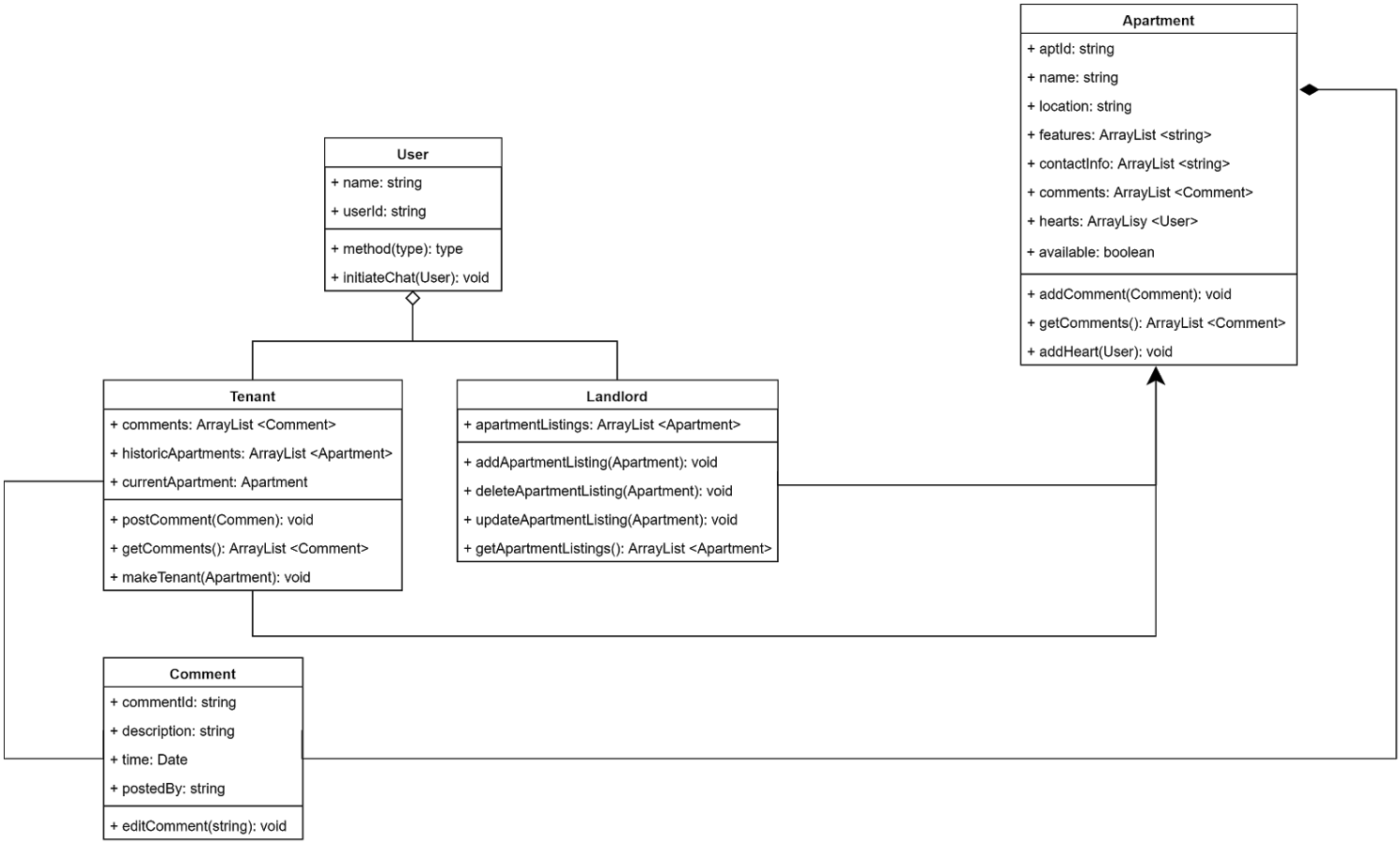

Figure 4: *Class Diagram for QuickRental*

3. Entity Relationship Diagram (ERD)

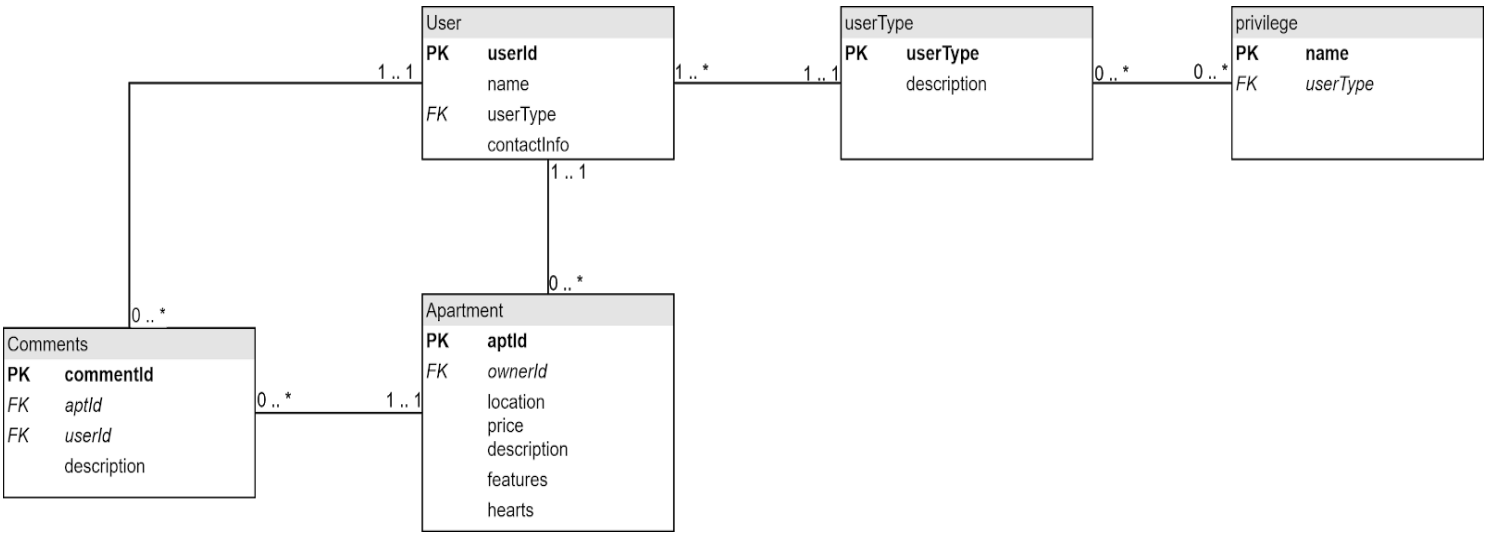

Figure 5: *Entity Relationship Diagram (ERD) for QuickRental*

4. Steps for the Sequence Diagram

To obtain the Sequence Diagram, we formulated the use cases, from which we made a ***Use Case Diagram***. We then ranked the use cases using the ***Priority Matrix Ranking***. The top priority use case was used for the ***Sequence Diagram***.

4.1 Use Case Diagram

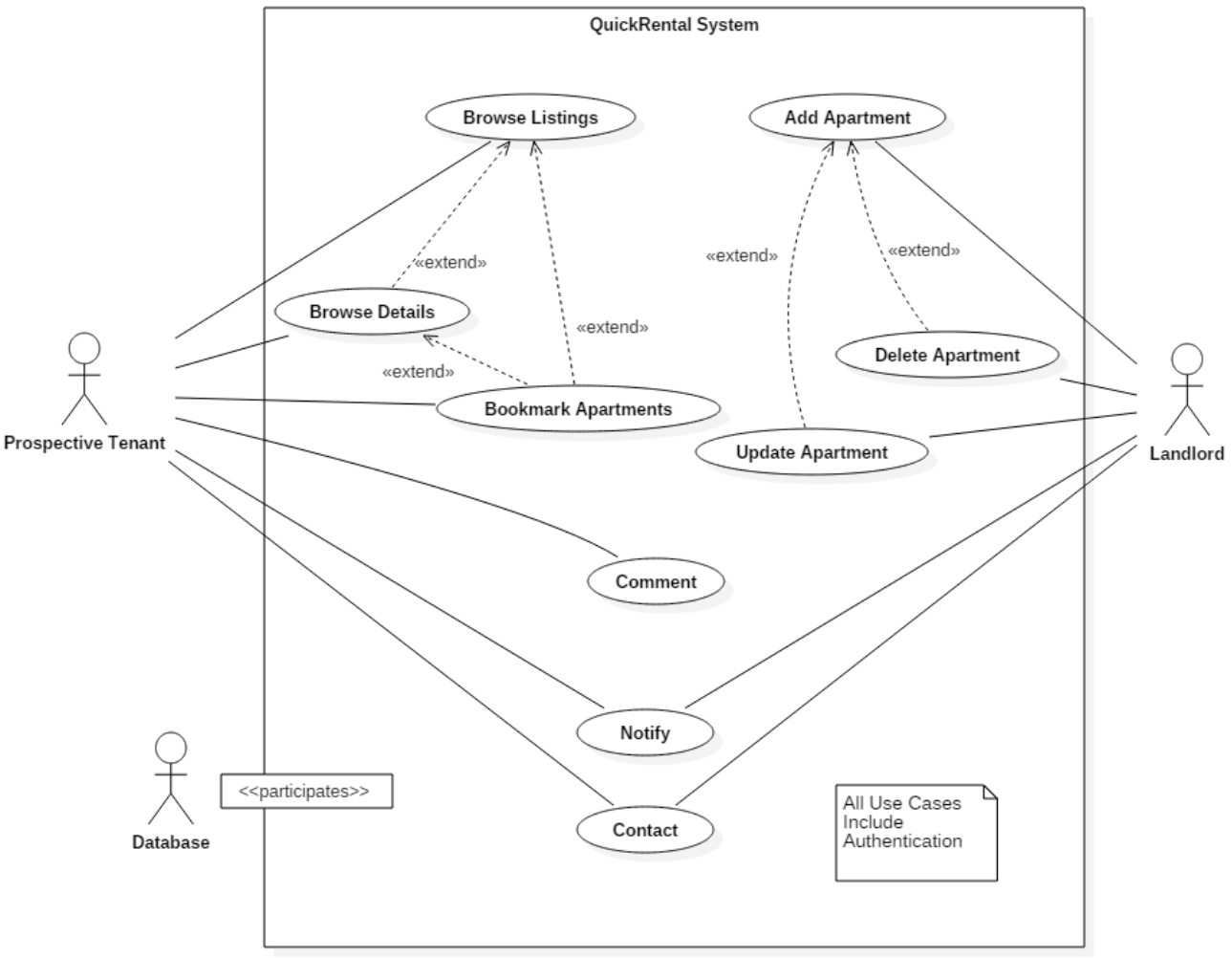

Figure 6: *Use case Diagram for QuickRental*

4.2 Priority Matrix Ranking

Ranking criteria used to evaluate the use cases on a 1-5 scale on these 6 criteria:

1. ***Significant Impact on the architectural design.***
2. ***Easy to implement but contains significant functionality.***
3. ***Includes risky, time-critical, or complex functions.***
4. ***Involves significant research or new or risky technologies.***
5. ***Includes primary business functions.***
6. ***Will increase revenue or decrease costs.***

Use-Case Name	*Ranking Criteria, from 1 to 5*						*Total Score*	*Priority*
	1	**2**	**3**	**4**	**5**	**6**		
Add Apartment	5	5	3	2	5	5	25	High
Browse Details	5	3	5	5	4	2	24	High
Browse Listings	5	5	2	2	5	3	22	High
Contact	2	3	4	4	4	3	20	High
Authenticate	5	5	3	1	5	1	20	Medium
Notify	4	4	2	1	4	3	18	Medium
Update Apartment	4	4	1	1	3	3	16	Medium
Delete Apartment	4	4	1	1	3	3	16	Medium
Bookmark Apartment	3	4	1	1	3	1	13	Low
Comment	3	3	1	1	2	3	13	Low

4.3 Sequence Diagram for Add Apartment

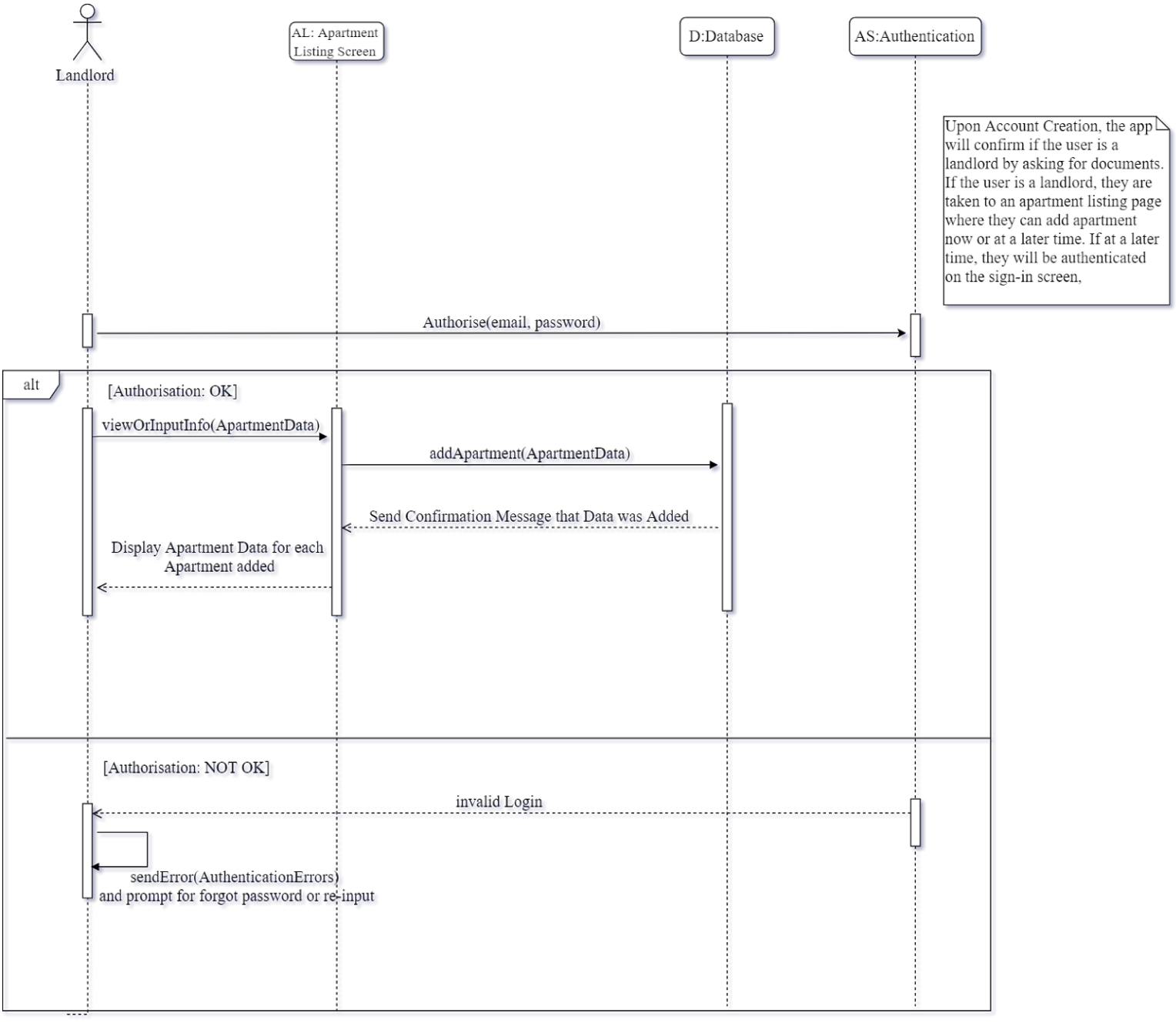

Figure 7: *Sequence Diagram for the highest priority use case, Add Apartment, for QuickRental*

5. Technology Utilisation

Technologies used for QuickRental

- ***React Native*** - React Native is a javascript framework created by facebook used to develop mobile applications. We choose react native since it allows developers to create a single application compatible with both IOS and Android using Javascript.

- ***Expo*** - Expo is an open source toolchain that is built around React Native. We chose Expo as it allows for easy creation, testing and deployment on iOS and Android operating systems. Furthermore, there are Over-the-air (OTA) updates. When the app is launched by the user, Expo searches for an update the developers (QuickRental) have published using expo publish. This allows us to update the app for our users without needing to publish the new .apk to the Play Store. It enabled us to deploy our app among users (students and landlords) without the financial expense of publishing to the Play Store, during this beta phase.

- ***Firebase Real-time Database (NoSQL)*** - The Firebase Realtime Database is a cloud-hosted NoSQL database. We choose firebase as our database as it was the simplest free and scalable solution we were able to implement within the given time. It is used for authentication for the user, to keep track of all added apartments by the Landlord, to keep track of the user's accounts, preferences, bookmarks, comments and likes, to allow the landlord to update or delete their apartment listings, to allow querying of apartment listings. However, we intend to implement a relational database in the future.

- ***Firebase Authentication*** - Firebase Authentication was used to create and authenticate users that registered with QuickRental. Authentication using passwords, phone numbers, and federated identity providers such as Google, Facebook and Twitter, are supported.

- ***Firebase Cloud Storage*** - Firebase Cloud Storage is used for the storage and retrieval of apartment images and verification documents provided by the landlord.

- ***Smartphone GPS*** - Used to detect the location of the apartment.

- ***Expo Maps Library with Google Maps*** - Used to display the location of the apartments to the user. Expo utilises 'Map View', which is a Map component that uses Apple Maps or Google Maps on iOS and Google Maps on Android. The library, react-native-maps, at react-community/react-native-maps is used. When connected to Google Maps, the user can see the distance from their location to U.W.I. St. Augustine. Google Maps is a Web-based service that provides detailed information about geographical regions and sites around the world. Our application uses the Google Maps API to provide our users with accurate location data. We are currently working with a free trial Google Maps API key but we intend to upgrade in the future.

- ***Custom React Native Chat using Firebase Cloud Messaging-*** leveraging custom React Native UI components and Firebase Cloud Messaging, an in-house chat application will be created. Benefits on in-house chat functionality include improved security, customisation, potential for anonymity and message scrubbing. Firebase Cloud Messaging is cross-platform, allowing the live chat to work on both Android and iOS operating systems.

- ***GitHub*** - GitHub was used for version control, to manage feature development in different branches and for easier integration.

Programming Languages Used

- React Native uses ***JavaScript***