TAUHEED BUTT
01-134191-068
AHMAD HASSAN
01-134191-002

ERP, CRM and Complaint Management System for Real Estate

Bachelor of Science in Computer Science

Supervisor: Dr. Fatima Khalique

Department of Computer Science
Bahria University, Islamabad

October 2022

Certificate

We accept the work contained in the report titled "ERP, CRM and Complaint Management System for Real Estate", written by Mr. Tauheed Butt AND Mr. Ahmed Hassan as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by . . . :

Supervisor: Dr. Fatima Khalique (Assistant Professor)

_____

Internal Examiner: Ali Irfan (Assistant Professor)

_____

External Examiner: Name of the External Examiner (Title)

_____

Project Coordinator: Ms. Maryam Khalid Multani (Assistant Professor)

_____

Head of the Department: Dr. Arif Ur Rahman (Head of Department / Sr. Associate Professor)

_____

October 28^{th}, 2022

Abstract

Estate+ is a cross platform application built to cater a real estate project's client and owner needs. It allows users to place a booking in real estate projects that have already completed their construction or still in one. The owners of such projects can also post their ads on the app for the clients to see. Installments paid by clients are also approved through the application as the admin of the owner authenticates them through a web portal. Owner gets to create admin accounts who manage their projects for them through the Web Portal. Admins further get support from Customer Relation Officers who can chat with users to deal with their queries through the same portal.

Acknowledgments

Supervisor at PrograminStudio where Tauheed Butt did his internship provided great help and understanding regarding Mobile Development in React Native. They also lend a hand in building UI for the mobile interface. Dr Fatima Khalique also laid the footpath for process to follow for the projects development which we are also greatful for.

TAUHEED BUTT, AHMED HASSAN
Islamabad, Pakistan

October 2022

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

ERP	Enterprise resource planning [2]
CRM	Customer relationship management [3]
CRO	Customer Relationship Officer
SQL	Structured Query Language
PERN	PostgreSQL, Express, React, and Node [4]
JWT	JSON Web Token [5]

Chapter 1

Introduction

1.1 Project Background/Overview

As time goes on, population of world increases and that population needs a roof over their head and places to work at. That requires more infrastructure to be constructed. To cater that growth, real estate comes in to play. One company could be managing thousands of different real estate projects and they need a central platform to do such tasks for both the clients and owners of those projects. Here Estate+, the proposed project, leads a helping hand with its cross-platform ability to cater all types of users.

1.2 Problem Description

Managing numerous real estate projects requires extreme amount of time and work dedication. A lot of paper work is involved and meetings with clients is day to day basis. If a client needs to get in contact with the owner, they need to contact the owner's managers who then schedule a meeting. Also to follow up their investments through the project's life span, client has to go through a hectic routine of processes to do so. Moreover, new projects in development which require investments from different clients is needed. It could be hard to reach potential clients who are looking for investments and have no knowledge of the correct process. Having a central platform which provides solutions to all the above issues would be a great helping hand.

1.3 Project Objectives

To cater the problems described in previous section, a cross-platform application is developed which can run on Web, Android and iOS. The mobile version of the application is

where all the clients and owners of real estate projects are targeted. However, the web will be accessible to authorized personal by the Owner such as Managers and CROs.

1.4 Project Scope

Clients can view the projects that are for sale, looking for investments etc. They can also track their payment records that are involved in a specific project. Client will also have the ability to submit their payment to the owner through the application which will be recorded in a database. Moreover, to cater customer services, client will be able to schedule meetings with the owner through the app and chat with Customer Relations Officer as well. Owners will interact with the application through mobile as well. They will have their dashboard on the phone which shows all the projects they own and payments involved in those projects. They will also have the ability to create Admin accounts for their managers which can be used to manage their business.
Managers of the owners will have access to the application on the Web Portal. They will be able to see all the projects created by their Owner and the transactions involved in them. All the scheduled meetings with the owner will be displayed on the web portal which will be first approved by the manager. Manager will also be able to create accounts for CROs which will be used for customer services purposes.
CROs will have all their chats with customers displayed on the web portal to communicate with the customers

Chapter 2

Literature Review

2.1 Zameen.com

[6] This is the leading real estate based application in Pakistan. Customers have developed strong belief in it for their search in real estate projects. However, it only connects the client with the owner. Client is unable to make payments through the app and keep them in track. Neither Owner can manage those payments through the application. This is the point where Estate+ will differ from it and attract those long lasting users of Zameen.com.

2.2 Graana

[7] It is also a similar real estate application compared to Zameen.com. It also only connects client and owner together rather than providing them ability to make payments. This point can also be used in the proposed project's favour and establish a strong position in it.

Chapter 3

Requirement Specifications

3.1 Existing System vs Proposed System

Table 3.1: Comparisons.

Application Name	Limitations	Proposed Solution
Zameen.com	No Project booking in app No Payment through app	Project booking in app No Payment through app Track payment in app
Graana	No Project booking in app No Payment through app	Project booking in app No Payment through app Track payment in app

3.2 Functional CRM Requirements

3.2.1 CRO Customer Service

CROs will be able to chat with customers through the Web Portal. Figure 4.42

3.2.2 Client Customer Services

Client will be able to contact the customer services of the projects they are affiliated with. They will do so by having a chatting interface. Figure 4.49

3.3 Functional Client Requirements [Client View]

3.3.1 Explore as Guest

Clients will have availability to explore the application without registering to it. However, when they decide to make a booking, they will be prompted with a message to login first.

3.3.2 Explore as Registered Account

Clients will be able to Register and Login to the application to keep their data saved in the database. This will allow users to change their device without having to worry of losing their data.

3.3.3 Edit Registered Account

Clients will be able to change their profile's details through the application

3.3.4 Book place in project

Clients will be able to make booking in a project they desire. They will provide paid invoice in the start to authenticate their booking. Once authenticated by the admin, they will be shown in the application.

3.3.5 Track Investments

All the investments the client made will be shown in the app where they can track their payment records and make further payments.

3.3.6 Schedule a Meeting

Client will be able to schedule meetings with the owner through the app which will be approved by the admin through Web Portal

3.3.7 Search

Clients will be able to Search through the application with different filters and sorting options to best suit their needs and requirements.

3.3.8 Add to Wish-list

Clients will be able to add projects they like to their wish-list for later interest. Guests will have wish-lists stored on their device whereas logged in users will have it stored on the server.

3.3.9 Add new Project

Owners will be able to add new projects to their tally for clients to take interest in through the mobile app.

3.3.10 View Scheduled Meetings

Owners will be able to see the approved meetings by the admin on their phone and also reschedule them if needed.

3.3.11 Create Admin Account

Owners will be able to create admin accounts for their managers.

3.4 Functional ERP Requirements

3.4.1 Manage Meetings, Payments and Projects

Managers who are given the admin credentials would be able to manage the projects, payments and meetings of their owner. Figure 4.51

3.4.2 Authenticate Payments

Managers will be able to authenticate payments made by clients through web portal. Figure **??**

3.4.3 Create CRO account

Managers will be able to create new CRO accounts for customer services. Figure 4.59

3.5 Functional Complaints Requirements

3.5.1 Register a complaint

Clients will be able to register a complaint for the project they have invested in. Figure 4.43

3.5.2 Re send complaint

Clients will be able to send a response to complaint for the project they have invested in. Figure 4.43

3.5.3 Reply to a complaint

Admins will be able to reply to different complaints they receive. Figure Figure 4.61

3.6 Non Functional Requirements

3.6.1 Encryption

Passwords will be stored in encrypted form in the database. Other data that needs encryption such as payments will also be done so to keep proper security of the project.

3.6.2 Performance

Least amount of local storage will be used when data is fetched from the API. Only the required data such as tokens, wish lists, roles etc. will be stored. It will be noted that the API calls will be kept to minimum to avoid multiple API calls where a single one would be sufficient.

3.6.3 Usability

The User Interface would be kept extremely simple to understand and follow a color pattern to keep the user interaction simple. Things like submit buttons, back buttons will have a single component styling to help user memorize the functionality of each simply.

3.7 Use Cases

3.7.1 Diagrams

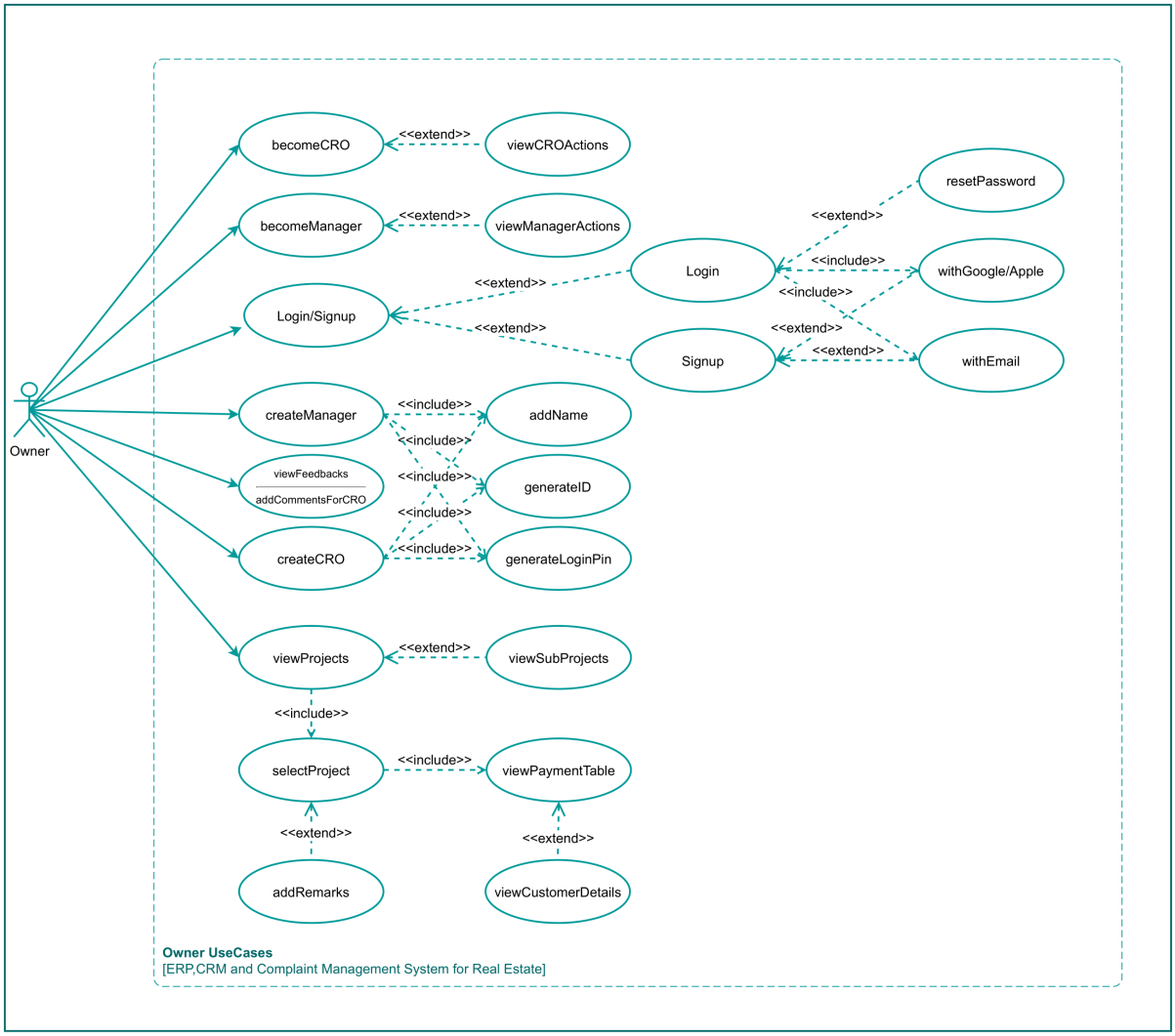

Figure 3.1: Owner Use Case

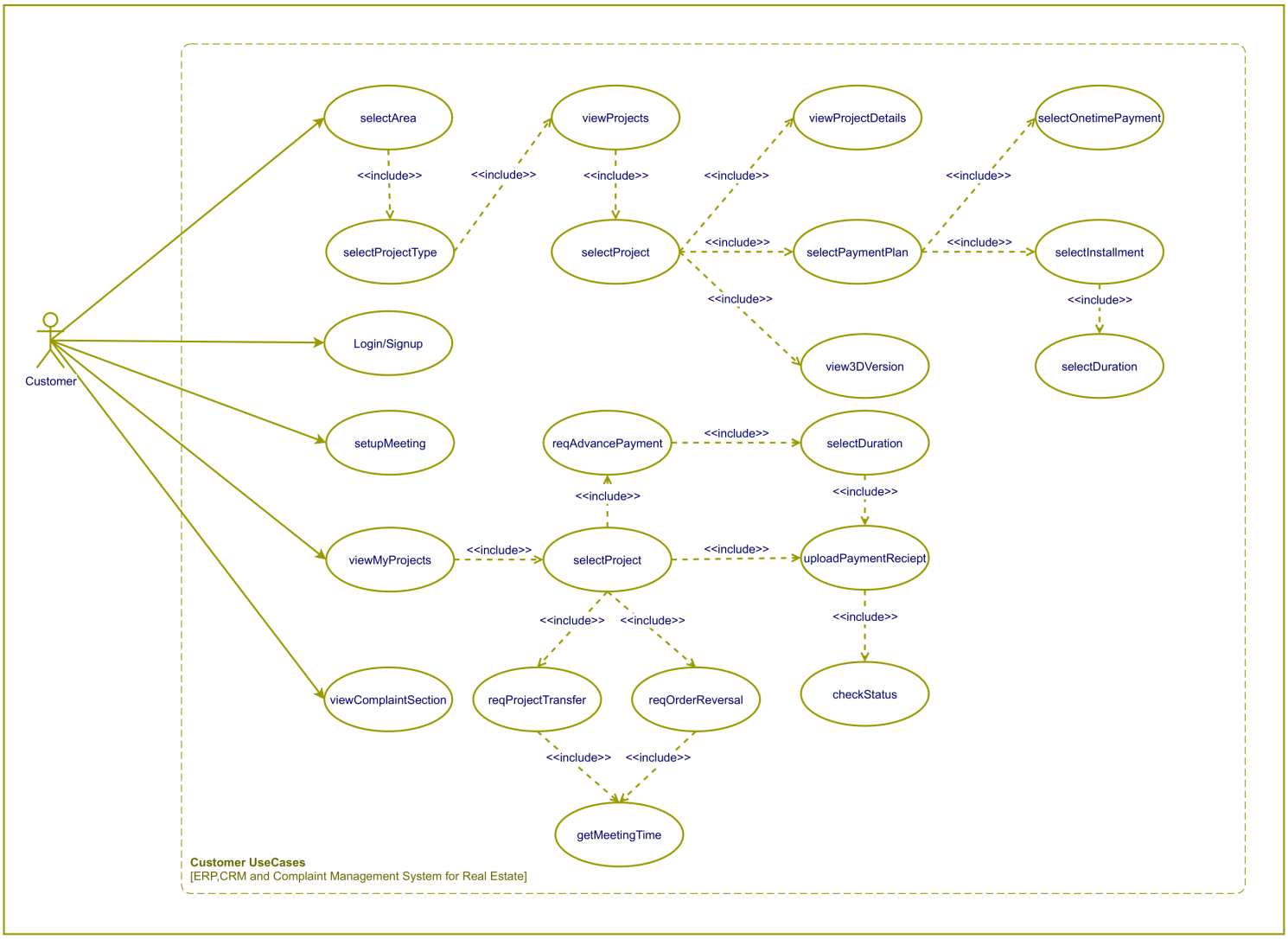

Figure 3.2: Customer Use Case

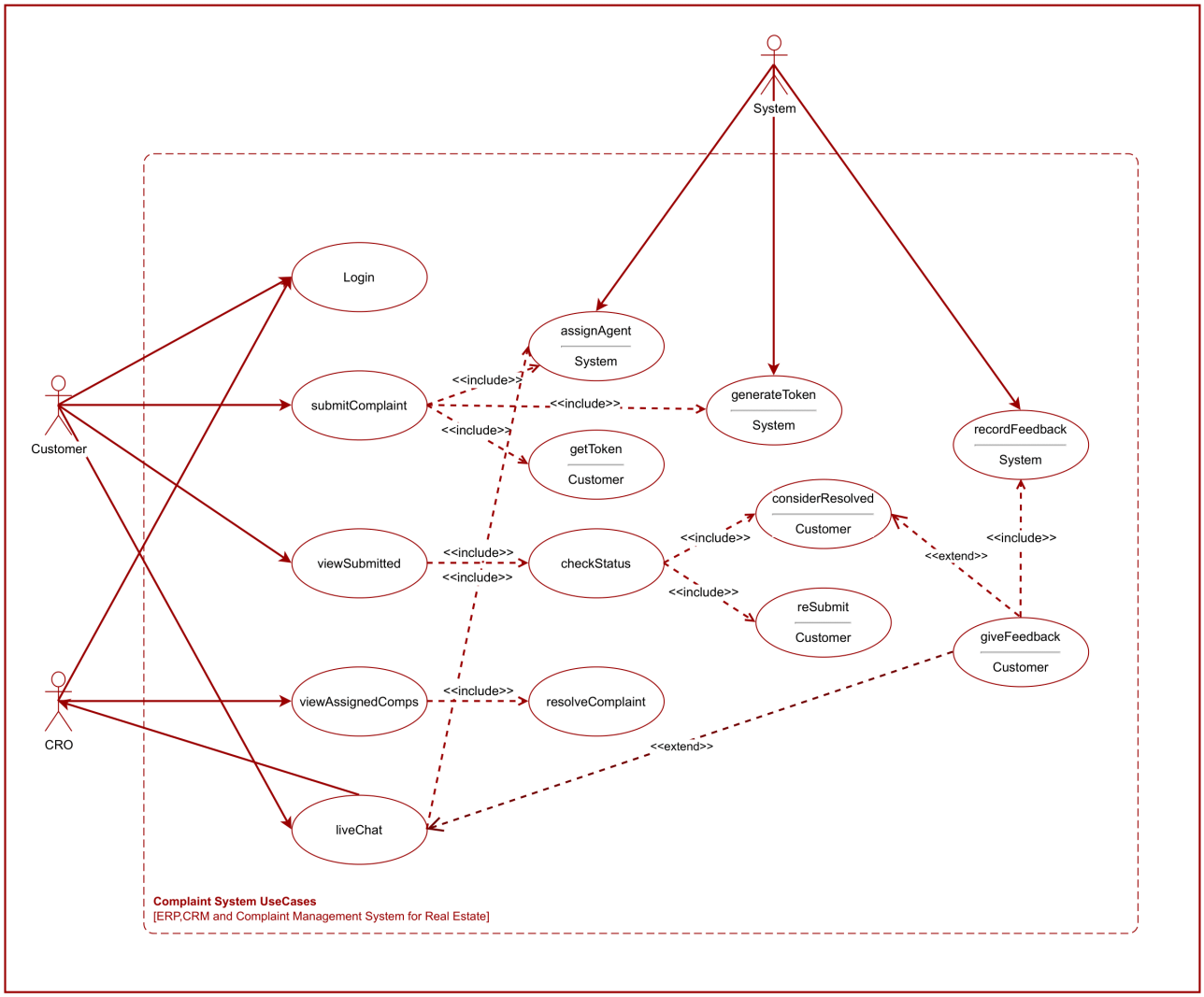

Figure 3.3: CRO Use Case

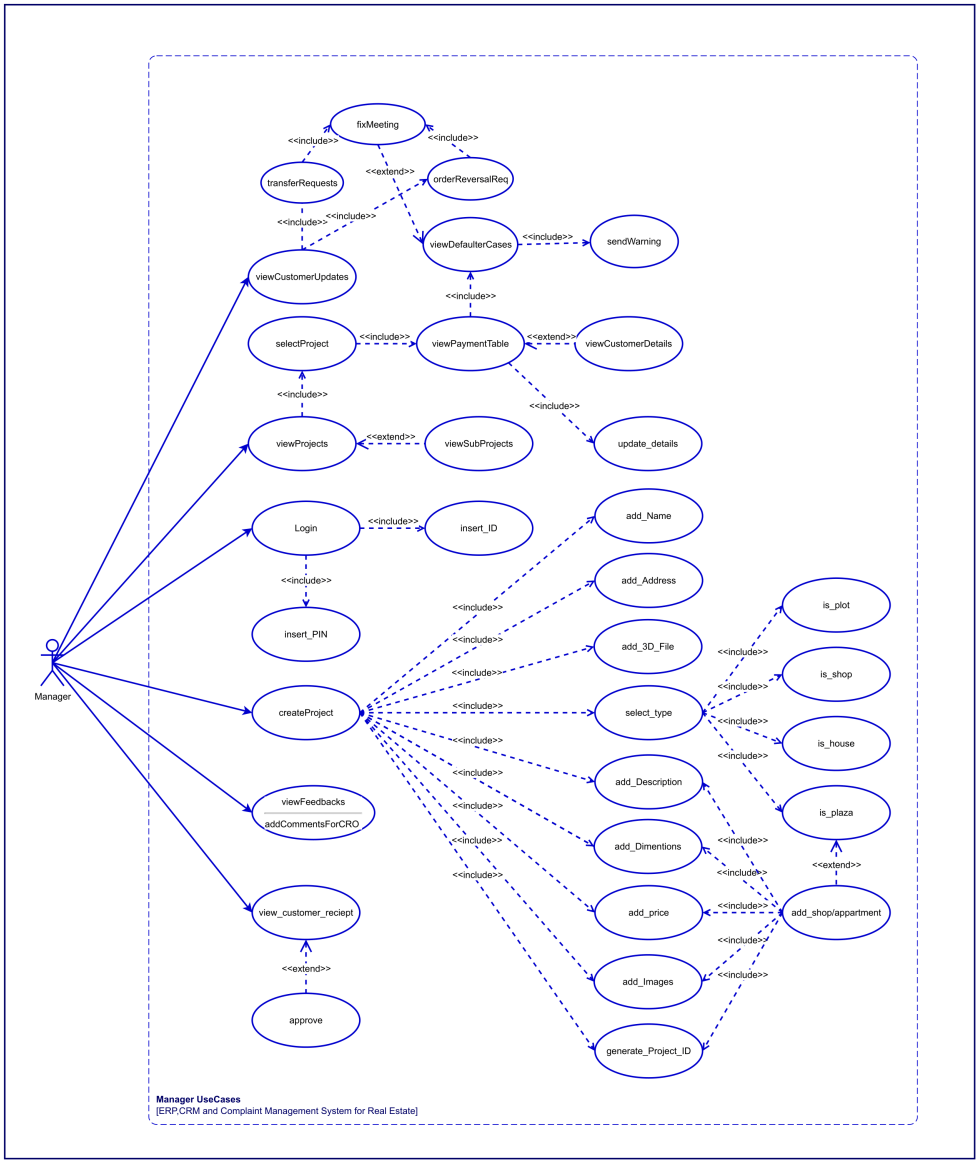

Figure 3.4: Manager Use Case

3.7.2 Description Tables

Table 3.2: UC-001

Use Case ID	**UC-001**	
Use Case Name	Login	
Actors	Owner, Manager, Customer, CRO	
Data	Username, Password	
Trigger	Press on Login Button/Profile Tab	
Pre-condition	Actor should already have registered credentials.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:** Users enter credentials in given fields	**System Response** **Step 2:** If the login is successful, they will Have access to their profile. If the Login is unsuccessful, they will be Prompted with a message to try again Or signup for account. In case of Manager and CRO, they Will be told to ask their Owner/Manager to provide them with credentials.
Alternate flow of events	· Login Fails	
	· Prompted with message	
Main success scenarios	Login succeeds	
Post-condition	User gets access to their profile.	

Table 3.3: UC-002

Use Case ID	**UC-002**	
Use Case Name	Sign up	
Actors	Owner, Customer	
Data	Email or Google/Apple ID.	
Trigger	Press on Sign Up Button	
Pre-condition	Have access to Application	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	Actor can use their Google/Apple ID to create account.	If same email or invalid email is being used for sign-up, user will be prompted to change it.
	They will have majority of the fields of sign-up form already filled.	If the sign-up completes, they can use their account to login.
	Actor can also use their email to create account.	On failure of sign-up, they get prompted with respective reasons
	They would provide details in required fields of sign-up columns.	
Alternate flow of events	· Signup Fails	
	· Prompted with message	
Main success scenarios	Signup succeeds and mail is sent to user for authentication.	
Post-condition	Actor can login using the credentials provided during signup	

Table 3.4: UC-003

Use Case ID	**UC-003**	
Use Case Name	Reset Password	
Actors	Owner, Customer	
Data	Email/Username associated with account	
Trigger	Select Forgot Password button	
Pre-condition	Have access to application	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	User will provide their username/email associated with the account. They will be mailed a link which will reset their password. On that link, they will be able to provide new password. The link will expire after 5 minutes	User will be prompted to check their mail.
Alternate flow of events	· Site is down	
	· No email is sent	
Main success scenarios	Resetting of Password Succeeds.	
Post-condition	User can use new password to login to their account	

Table 3.5: UC-004

Use Case ID	**UC-004**	
Use Case Name	Create a Manager Account	
Actors	Owner	
Data	Manager's email and other details.	
Trigger	Click on add a manager button	
Pre-condition	Already logged in	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:** User will provide manager's details (name, dob etc.).	**System Response** **Step 2:** If the manager with same email exists, user will be prompted.
	Will also select projects of which the manager will be a manager of.	A unique username and password will be generated. It will be emailed to the manager which they will be able to use to login to app.
Alternate flow of events	· Creation Fails	
	· Prompted with message	
Main success scenarios	Manager's Account gets created.	
Post-condition	Manager will be able to login. User will be able to perform CRUD operations on created manager.	
Comments		

Table 3.6: UC-005

Use Case ID	**UC-005**	
Use Case Name	Create a CRO Account	
Actors	Manager	
Data	CRO's email and other details.	
Trigger	Click on Add a CRO button.	
Pre-condition	Already logged in	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:** User will provide manager's details (name, dob etc.). User will select projects on which CRO will receive complaints for.	**System Response** **Step 2:** If the CRO with same email exists, user will be prompted. A unique username and password will be generated. It will be emailed to the CRO which they will be able to use to login to app.
Alternate flow of events	· Creation Fails	
	· Prompted with message	
Main success scenarios	CRO's account gets created successfully.	
Post-condition	CRO will be able to login. User will be able to perform CRUD operations on created manager.	

Table 3.7: UC-006

Use Case ID	**UC-006**	
Use Case Name	Create a Project	
Actors	Owner	
Data	Project's Details (name, address, 3D file etc.)	
Trigger	Click on Add a Project Button.	
Pre-condition	Already logged in	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	User will add project's details.	The project gets created if all required provided.
	They can also provide Autodesk files for 3D view of their project.	The user will be given option to make it public or keep it private.
		Once public, customers will be able to perform respected actions on it.
Alternate flow of events	· Creation Fails	
	· Prompted with message	
Main success scenarios	Project gets added successfully.	
Post-condition	User will have access to project.	

Table 3.8: UC-007

Use Case ID	**UC-007**	
Use Case Name	View a Project	
Actors	Owner	
Data	—	
Trigger	Clicks on a project.	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action**	**System Response**
	Step 1:	**Step 2:**
	User can modify the project.	Once project modified, all the users associated with it will be notified.
	User will be able to view sub projects.	Customers that are way behind their payment times will be highlighted.
	User will be able to see customer's payments.	
	Can also view customer's details.	
Alternate flow of events	· API doesn't return project details · Error shown	
Main success scenarios	Project gets opened on application.	

Table 3.9: UC-008

Use Case ID	**UC-008**	
Use Case Name	Delete a Project	
Actors	Owner	
Data	—	
Trigger	Clicks on Delete Button	
Pre-condition	Project opened on application.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action**	**System Response**
	Step 1: User will provide reasons for project removal. User will provide password to confirm deletion.	**Step 2:** All the users associated with project will be notified. Customers who had invested will get their meetings scheduled with the manager to discuss payments.
		On invalid password, the user will be prompted. If 5 wrong attempts are made, the user won't be able to delete that project for next week.
Alternate flow of events	· Project isn't deleted · Prompted with message	
Main success scenarios	Project is deleted.	

Table 3.10: UC-009

Use Case ID	**UC-009**	
Use Case Name	Become a Manager/CRO	
Actors	Owner	
Data	—	
Trigger	Click the respective button	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:** User will select the projects for which they wish to become the manager/CRO.	**System Response** **Step 2:** User will be shown their roles on their dashboard.
Alternate flow of events	· If not possible, option isn't shown to the user	
Main success scenarios	User successfully acquires the role they wished to achieve.	
Post-condition	User will have access to CRO/manager Roles.	

Table 3.11: UC-010

Use Case ID	**UC-010**	
Use Case Name	View Scheduled Meetings	
Actors	Manager	
Data	—	
Trigger	Click the respective button	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	Actor will view all the meetings that are pending with customers They will be able to see them in different tabs according to their category Clicking on a single meeting will allow them to fix a date with the customer. Either they choose the date suggested by the customer or propose a new one to the customer.	Any action taken regarding the scheduled meeting will send a notification to the customer. If new data is proposed, customer will be notified respectively. On creation of new meeting, the customer will also be notified.
	Actor will also be able to schedule new meetings with customers by mentioning the reason for it as well.	
Alternate flow of events	· API doesn't return meetings details	
	· Error shown	
Main success scenarios	User successfully viewed all the meetings	
Post-condition	User will have access to all the scheduled meetings.	

Table 3.12: UC-011

Use Case ID	**UC-011**	
Use Case Name	View a Project	
Actors	Manager	
Data	—	
Trigger	Click on any shown project.	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	User will be able to edit project's details, price etc.	Once a project is opened, user will be able to see all the customers involved in it.
	User will be able to notify customers regarding their payment's issues.	Each customer's last payment will be shown.
	User will also be able to schedule a meeting with the customer.	Ones who are behind the payment, they will be highlighted.
Alternate flow of events	· API doesn't return project details	
	· Error shown	
Main success scenarios	User successfully opens the Project.	
Post-condition	User will have access to all the elements involved in the Project.	

Table 3.13: UC-012

Use Case ID	**UC-012**	
Use Case Name	View Payments	
Actors	Manager	
Data	—	
Trigger	Click on respective button.	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	Actor will be able to view all the payments. Actor will be able to approve or disapprove the payments shown. Actor will view the transcript submitted by actor in pdf format. Actor will trace it by viewing transactions in bank account. Actor can provide new deadline for user for re-submission of invoice in case of disapproval.	On successful payment approval, a popup will be shown. If payment disapproved, actor will be prompted new set of possible actions. These actions will include provide new deadline, remove customer from project, schedule a meeting etc.
Alternate flow of events	· API doesn't return transactions details · Error shown	
Main success scenarios	User successfully opens the Payments Section.	
Post-condition	User will have access to all the payments.	

Table 3.14: UC-013

Use Case ID	**UC-012**	
Use Case Name	View Payments	
Actors	Manager	
Data	—	
Trigger	Click on respective button.	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	Actor will be able to view all the payments.	On successful payment approval, a popup will be shown.
	Actor will be able to approve or disapprove the payments shown.	If payment disapproved, actor will be prompted new set of possible actions.
	Actor will view the transcript submitted by actor in pdf format. Actor will trace it by viewing transactions in bank account.	These actions will include provide new deadline, remove customer from project, schedule a meeting etc.
	Actor can provide new deadline for user for re-submission of invoice in case of disapproval.	
Alternate flow of events	· API doesn't return transactions details · Error shown	
Main success scenarios	User successfully opens the Payments Section.	
Post-condition	User will have access to all the payments.	

Table 3.15: UC-014

Use Case ID	**UC-014**	
Use Case Name	Search for a Project	
Actors	Customer	
Data	—	
Trigger	Click on search bar.	
Pre-condition	Have app installed on mobile device.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action**	**System Response**
	Step 1: The actor will enter any text in the text bar. Actor will also be able to filter the search results by specifying area, dimensions price range etc.	**Step 2:** On pressing enter or search icon, the entered text will be searched across the database. The results will be shown on user's screen. If none found, user will be prompted respectively.
Alternate flow of events	· No items found	
	· Error shown	
Main success scenarios	User gets shown the search results.	
Post-condition	User will have be able to open up searched projects.	

Table 3.16: UC-015

Use Case ID	**UC-015**	
Use Case Name	Book a Project	
Actors	Customer	
Data	—	
Trigger	Click on book button.	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action**	**System Response**
	Step 1:	**Step 2:**
	Actor will provide the payment plan, whether its instalment or onetime payment.	On successful booking, the actor will have to wait for it to be approved. It will be promoted as well. Actor will be able to explore the app while the process completes.
	On case of instalment, actor will select durations offered by the owner of the project.	In-order to view approval process, actor will have to go pending tab.
	Actors' information will be entered automatically in shown fields by using data provided upon signup.	In case of unsuccessful booking, respective error will be shown.
Alternate flow of events	· Project already booked by customer	
	· Project has no bookings opened	
Main success scenarios	User books a project successfully.	
Post-condition	Project will get added to user's profile.	

Table 3.17: UC-016

Use Case ID	**UC-016**	
Use Case Name	Manage Personal Projects	
Actors	Customer	
Data	—	
Trigger	Click on Personal Projects tab.	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action**	**System Response**
	Step 1:	**Step 2:**
	Actor will be able to view all the projects they are involved in. Whether they are pending booking approval or already in instalment process.	Upon payment's approval, actor's remaining required payment for the project will be updated. Also, the remaining payments duration will also be updated.
	Upon clicking on a project, Actor will be able to see project's current process and their own involvement's current process.	Projects whose payment is lagging will be highlighted and organized in separate tabs as well.
	If any project has payment pending, actor will be able to upload the invoice from there as well. Actor will be able to request for extending or shortening the instalment duration (for example adding 6 more months or removing 3 months from current plan).	In case of removal of project, user will be prompted to schedule a meeting with the project manager.
	Actor will be able to schedule a meeting with project's Manager as well. Meetings reasons can involve need to transfer project, reverse project order etc.	
Alternate flow of events	· API doesn't return personal projects	
	· Error shown	
Main success scenarios	User gets shown all their personal projects.	
Post-condition	User will have access to all their personal projects	

Table 3.18: UC-017

Use Case ID	**UC-017**	
Use Case Name	Get Help	
Actors	Customer	
Data	—	
Trigger	Click on Help button	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	Actor will be shown different common problems.	Some problems will be able to solve themselves within the app.
	Upon selection of any problem, Actor will be guided through all the steps required to handle that problem.	If the problem doesn't get solved, the user will be asked to talk with the Customer Relationship Officer.
	In case of problem not being listed, actor will be able to start a chat with the Customer Relationship Officer.	System will automatically allocate a CRO to the user.
	Actor will be able to send a detailed query to the CRO without starting a chat.	Upon updates received from CRO, the customer will be notified respectively.
		The chat can be terminated by CRO or the system automatically if no response received from actor in 10 minutes.
		Once chat is terminated, the user will get assigned a new CRO in case of new complaint registration.
Alternate flow of events	· No CRO available · Error shown to user	
Main success scenarios	User gets their problem solved.	
Post-condition	User will be able to register any problem with the CRO.	

Table 3.19: UC-018

Use Case ID	**UC-018**	
Use Case Name	View Registered Complaints	
Actors	Customer	
Data	—	
Trigger	Click on Complaints Section	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:** Actor will see all the complaints they have registered and their follow-up along with it. If the follow-up doesn't exist, actor will be able to re-submit the complaint with a click of a single button. If the user doesn't need the complaint to be handled anymore, they can remove it from their and CRO's end as well by clicking a single button.	**System Response** **Step 2:** In case of no follow-up, actor will be provided a button to re-submit the complaint.
Alternate flow of events	· No complaints registered.	
	· Error shown to user	
Main success scenarios	User views their complaints' status.	
Post-condition	User will have access to their complaint.	

Table 3.20: UC-019

Use Case ID	**UC-019**	
Use Case Name	View assigned complaint	
Actors	CRO	
Data	—	
Trigger	Open their portal	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	Actor will view all the customers whose complaints need handling on the page.	New message or complaint form customer will be shown as a small pop-up notification on one side of the screen.
	Upon clicking on a customer, the actor will view their complaint and take actions to solve it. Upon completion of a complaint, actor will be able to terminate the chat. Actor will be able to sort the complaints alphabetically, first registered, or custom. Actor will also be able to categorize the complaints on basis of projects etc.	Upon termination of chat, user will be notified respectively.
Alternate flow of events	· No response	
Main success scenarios	Actor gets to view all the complaints	
Post-condition	Actor has access to all the complaints assigned to them.	

Table 3.21: UC-020

Use Case ID	**UC-020**	
Use Case Name	Resolve a complaint	
Actors	CRO	
Data	—	
Trigger	Open a complaint	
Pre-condition	Already logged in.	
Assumptions	Actors have access to internet	
Normal flow of events	**Actor Action** **Step 1:**	**System Response** **Step 2:**
	Upon complaint's contents, actor will access those parts of the system that are concerned. Actor will be able to view system's databases related to that customer in-order to resolve their query. Actor will be able to view all sorts of transactions performed by the customer. On unsuccessful handling, the customer will have meeting scheduled with the respected manager by the ACTOR.	On successful handling of problem, user will be notified, and chat will be terminated.
Alternate flow of events	· Meeting gets scheduled with the manager and customer.	
Main success scenarios	Actor successfully resolves the complaint.	
Post-condition	Complaint gets removed from Actor's dashboard.	

Chapter 4

Design

4.1 System Architecture [1]

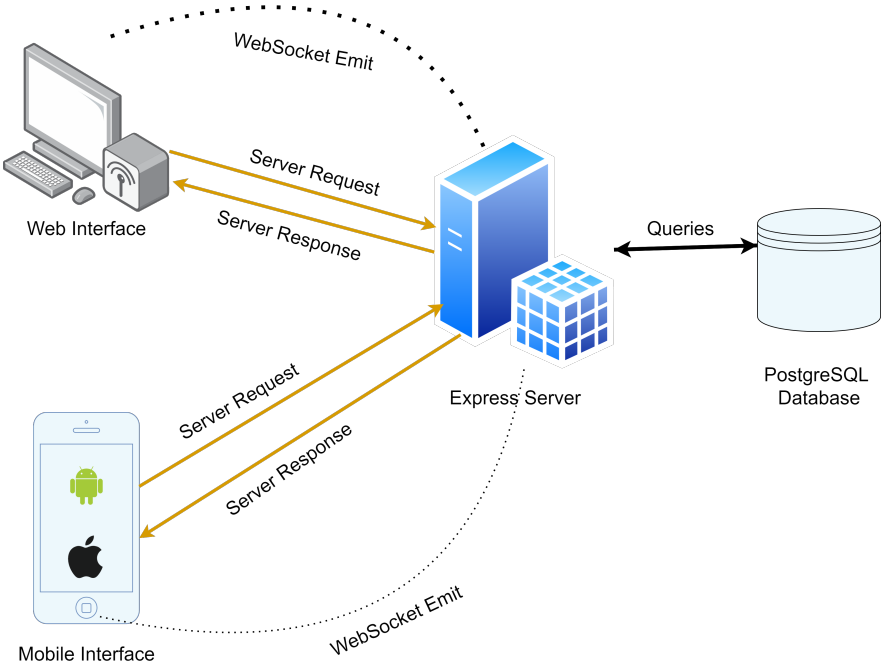

Figure 4.1: Architecture Diagram

4.2 Design Constraints

The database is hosted at Heroku Server [8]. This server limits the connections up to 20 and maximum of 10,000 rows. For a real-world implementation, this can cause issue as the space would run out quickly and maximum of 20 users will be able to use the

application. However, since this approach has no expenses, it saves the project from monetary investments. The middle server is also hosted on a free solution on Heroku which also runs at a slower speed rather than a paid one.

4.3 Design Methodology

Functional Decomposition [9] is followed for the creation of the design as the front-end was designed first which holds all the functionality. After the front-end was successfully designed, only than the database was designed. To design the front-end for mobile, other relative applications such as real-estate, e-commerce apps were looked at. The screens and components that were being repeated over there were made compulsory for this application to have. Since on the server, mobile and web application, screens and components were designed using functional components, it is safe to say it followed the structured methodology. However, for database, a lot of fields and entities are involved which fall under the object-oriented methodology.

4.4 High Level Design

We have shown the system architecture design in Section 4.1. Now we are going to elaborate the working with the help of package diagram and deployment diagram. We have tried to present our system in the most understandable manner.

4.4.1 Conceptual or Logical View

Authentication Package needs Login, Forget and Reset Package for successful login. Authentication is same for Customer, Admin, Owner and Customer Sales Agent. This package will be further elaborated in next section with the help of interaction diagram.

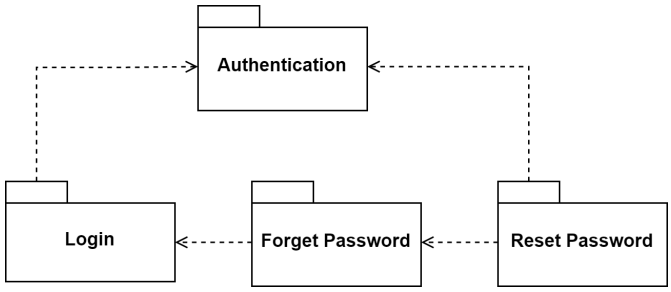

Figure 4.2: Authentication Package

In Customer Relation Management Portal we have Disputes Package that presents list of assigned disputes by using Chat package to chat with the customer and Profile package to edit profile.

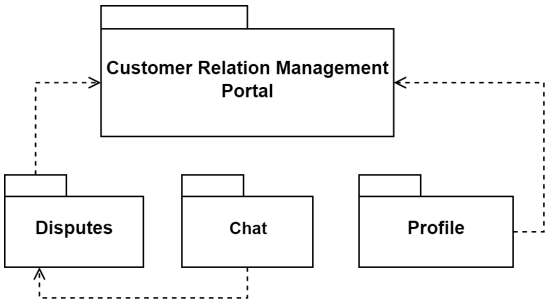

Figure 4.3: CRM Portal Package

Admin portal Package includes Projects, Transaction, Meeting etc, for full functionality.

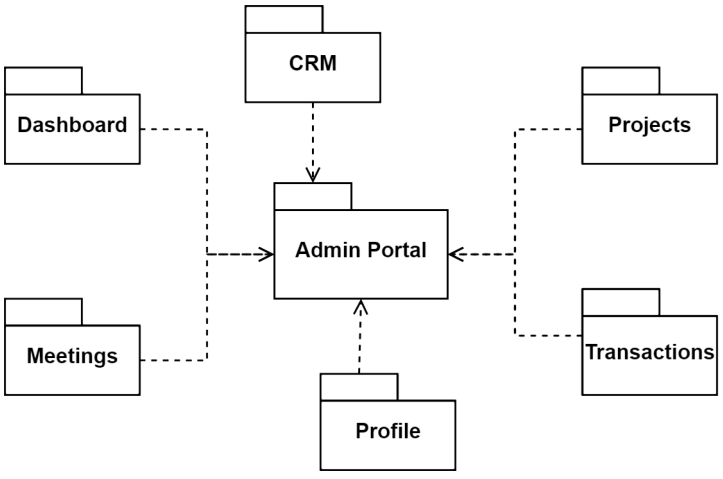

Figure 4.4: Admin Portal Package

By using below packages we get complete web portal package that redirects users to their specific Portal according to their roles.

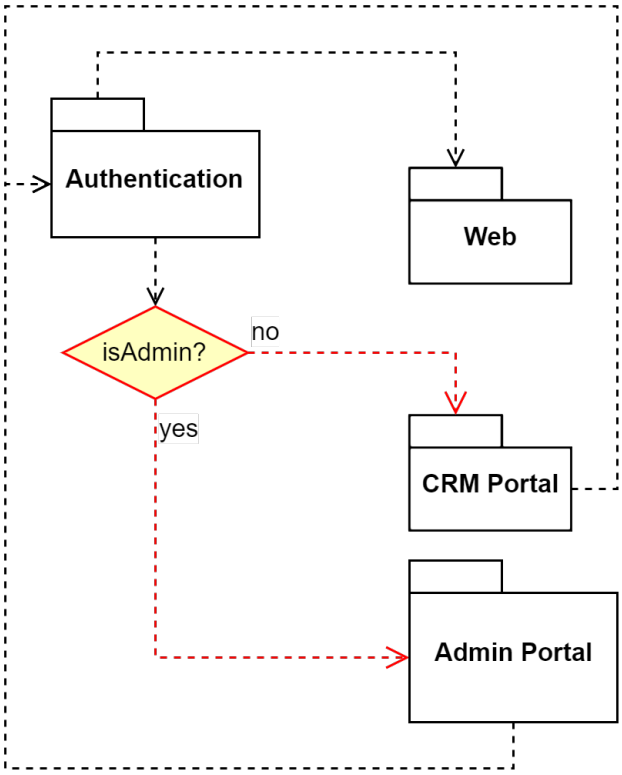

Figure 4.5: Complete Web Package

4.4.2 Process

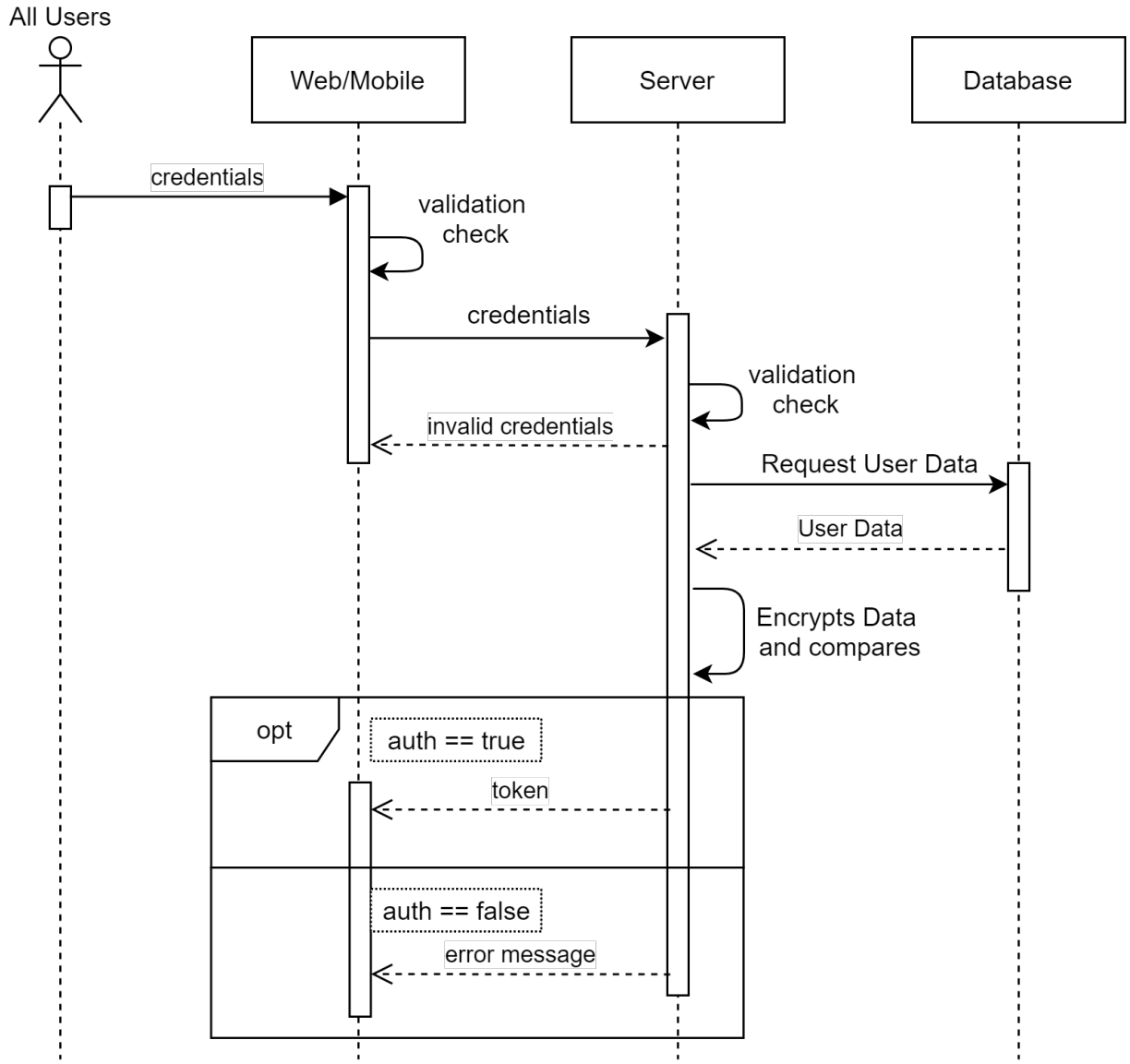

Figure 4.6: Login

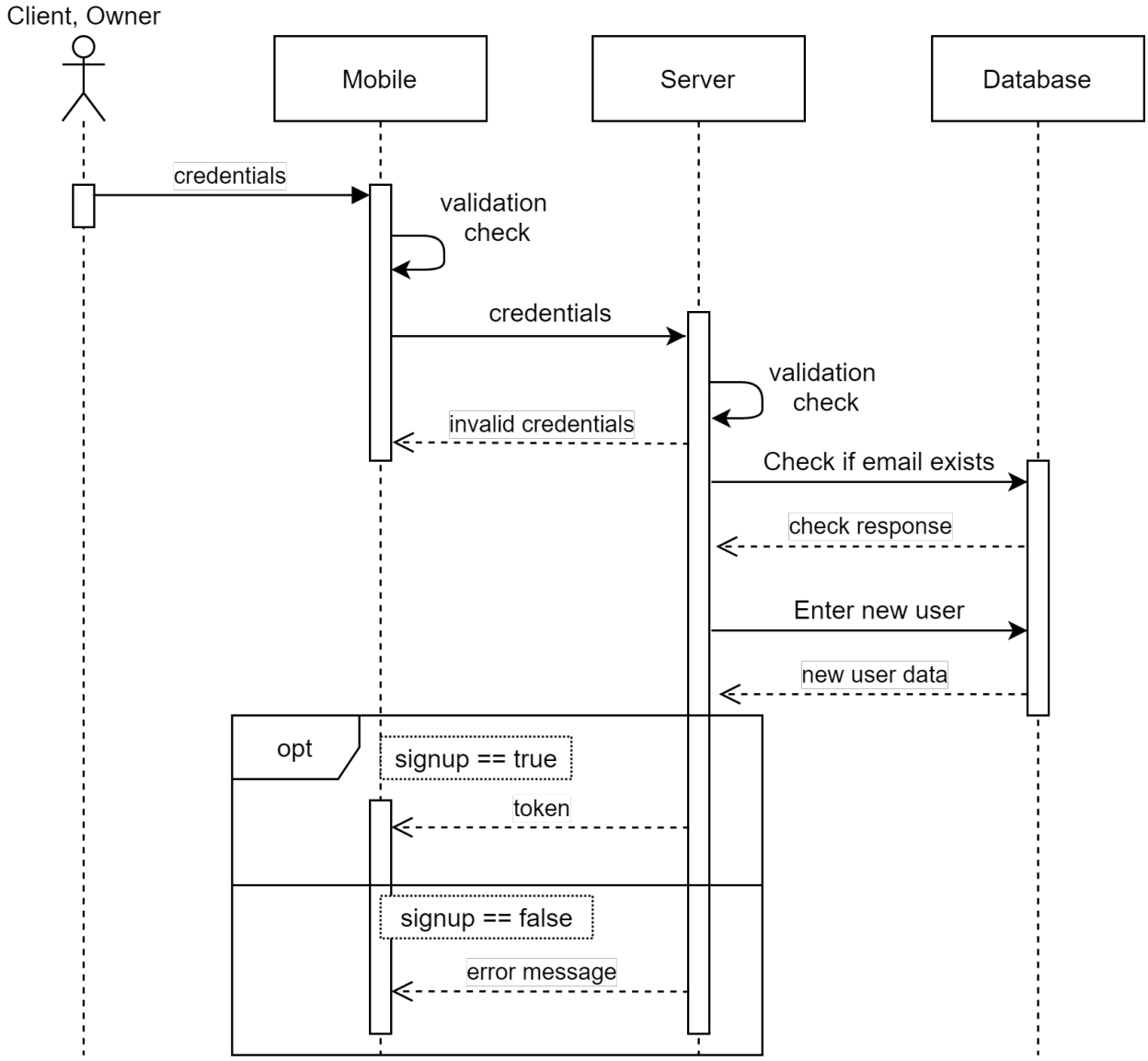

Figure 4.7: Register

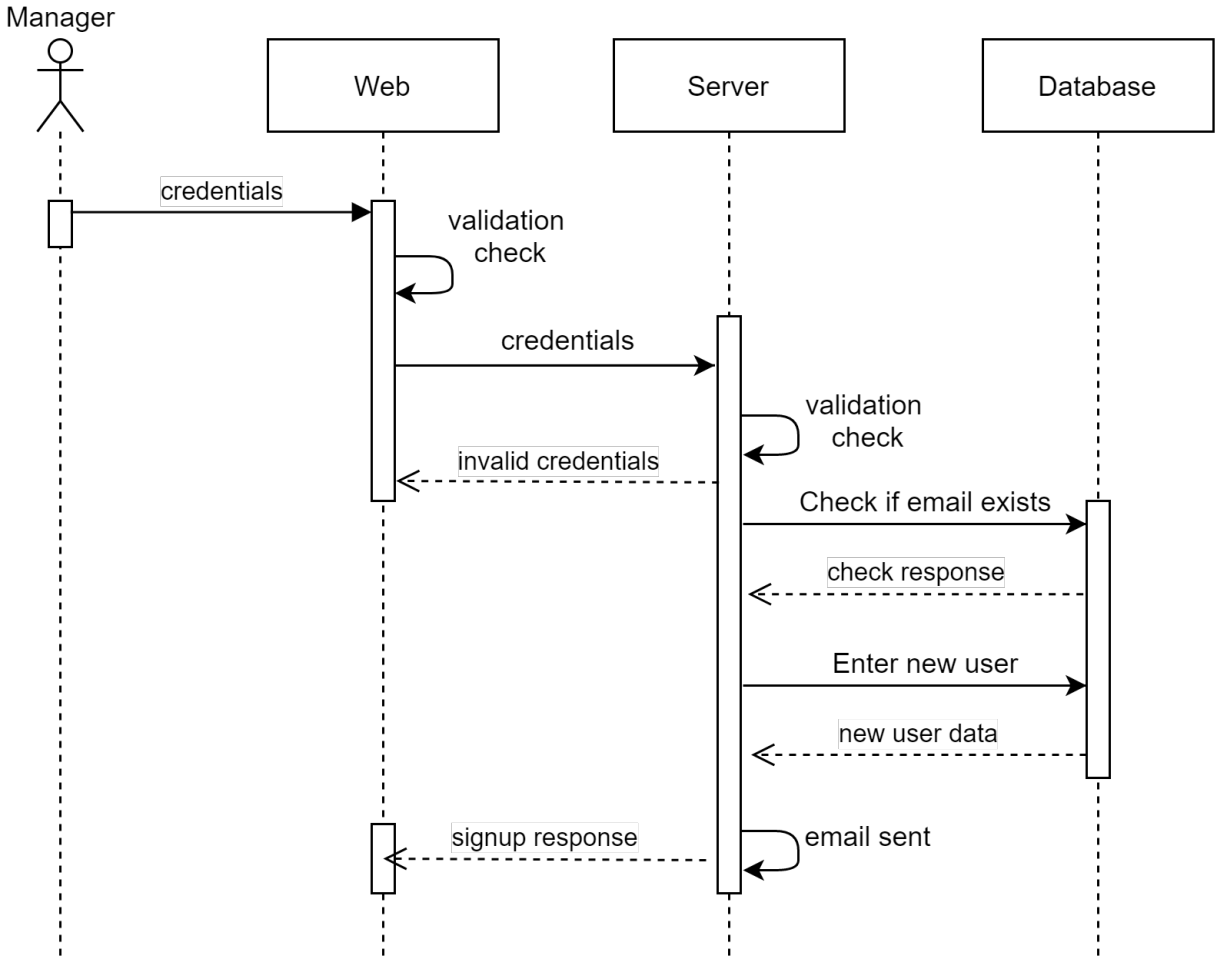

Figure 4.8: New CRO

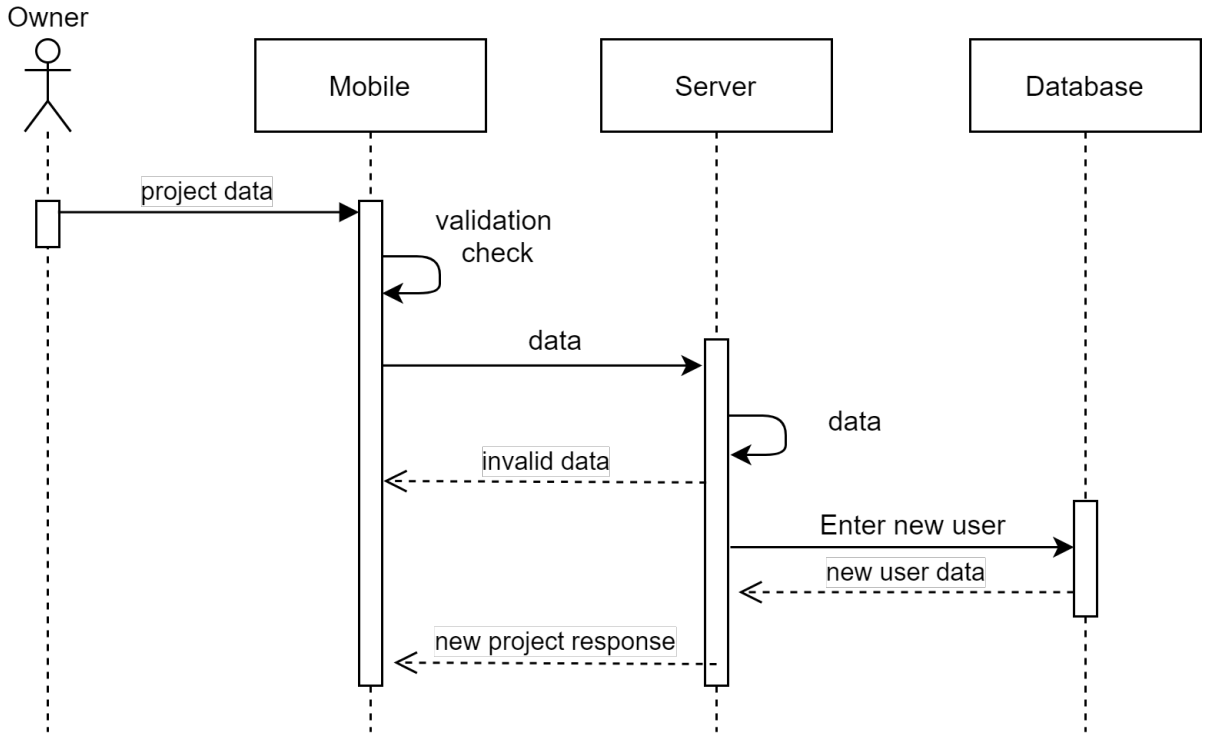

Figure 4.9: New Project

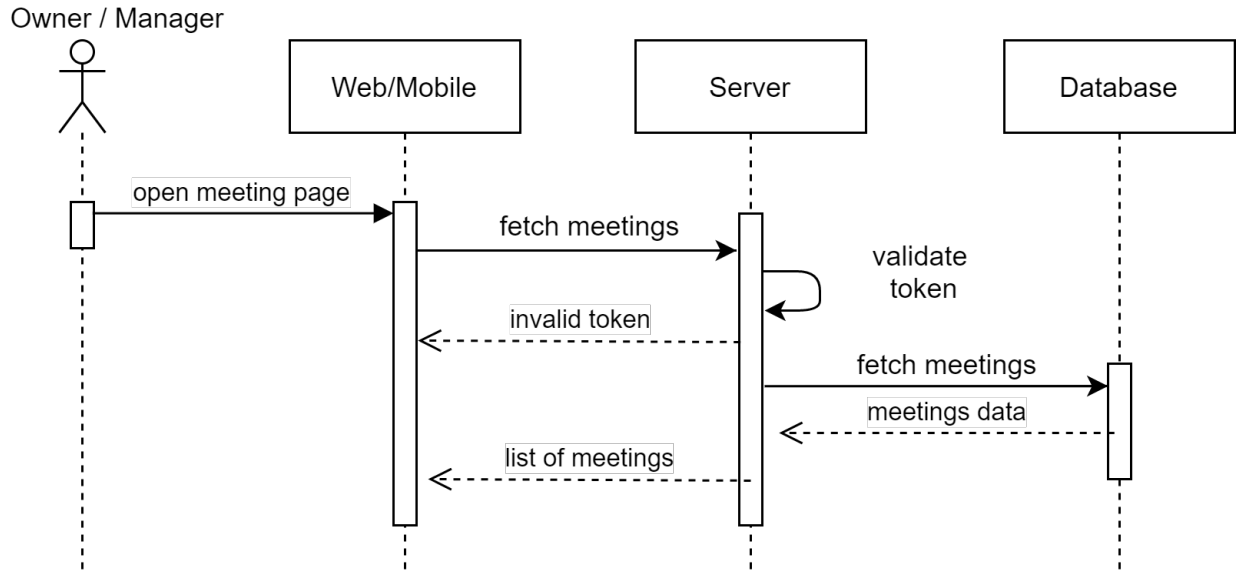

Figure 4.10: Fetch Meetings

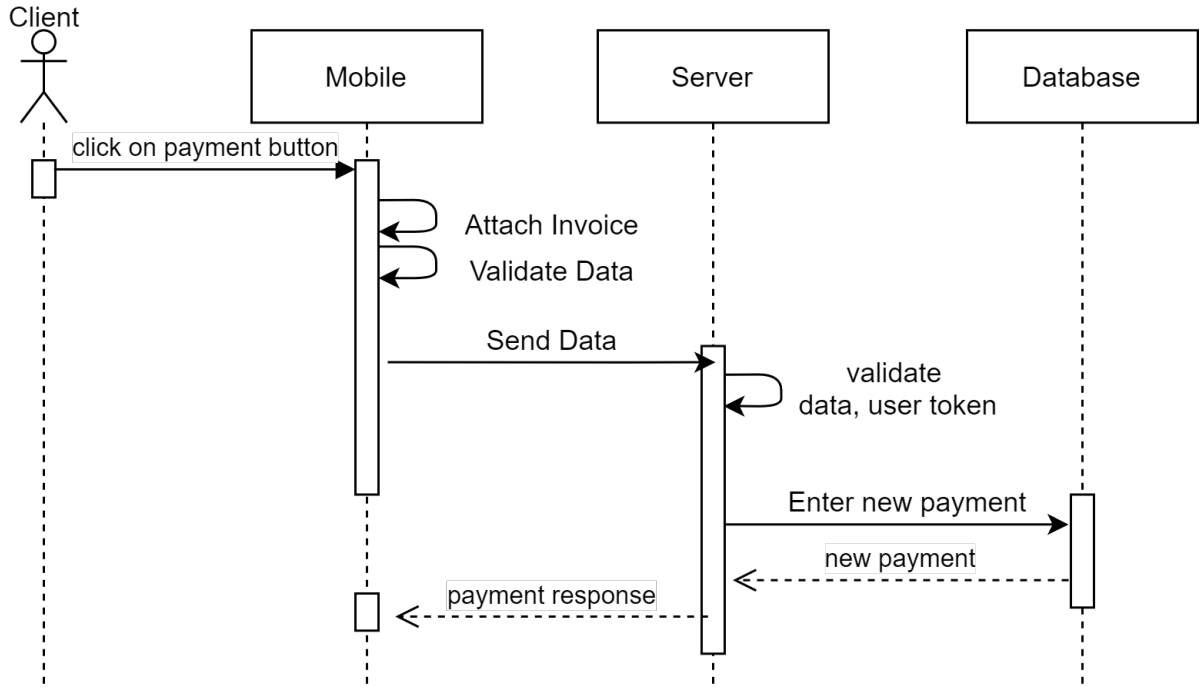

Figure 4.11: Make Payment

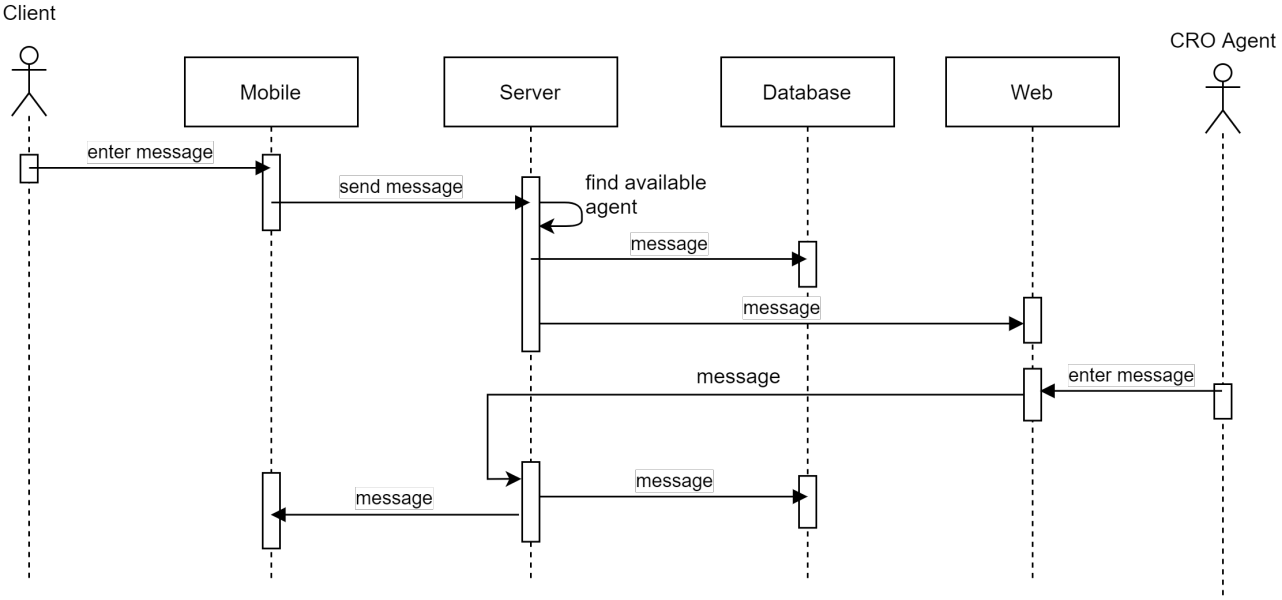

Figure 4.12: Chat

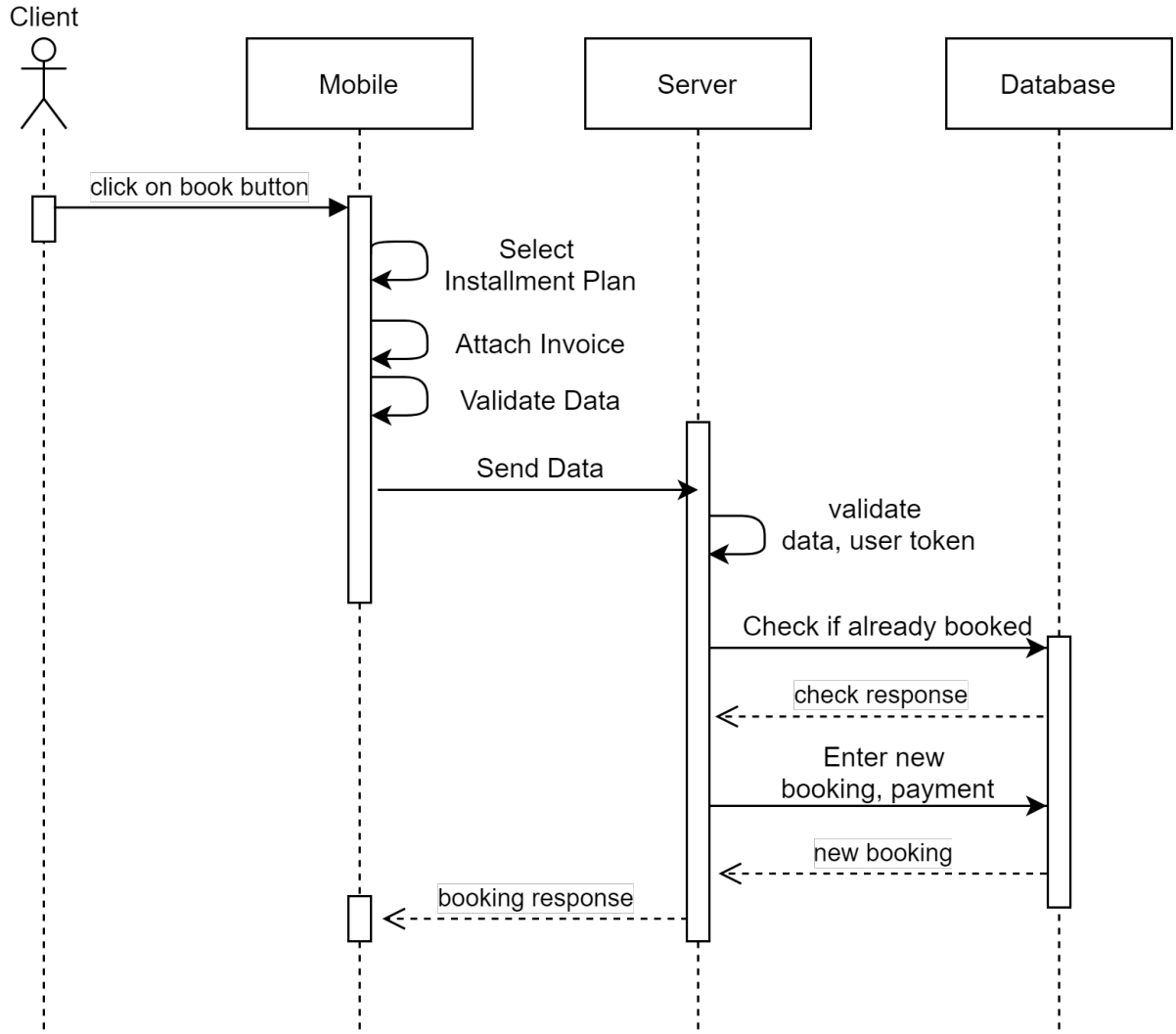

Figure 4.13: Book Project

4.4.3 Module

We are following the standard model of react directory structure.The visual representation is shown below.

```
api/
APIUtils.js
APIUtils.test.js
ProfileAPI.js
UserAPI.js
components/
Avatar.js
Avatar.css
Feed.js
Feed.css
FeedStory.js
FeedStory.test.js
Profile.js
ProfileHeader.js
ProfileHeader.css
pages/
auth/
Login.js
Forget.js
Reset.js
admin/
Dashboard.js
Transactions.js
Projects.js
Meetings.js
CRM.js
crm/
Chat.js
Profile.js
utils/
.
.
.
```

This is the standard directory system of React JS

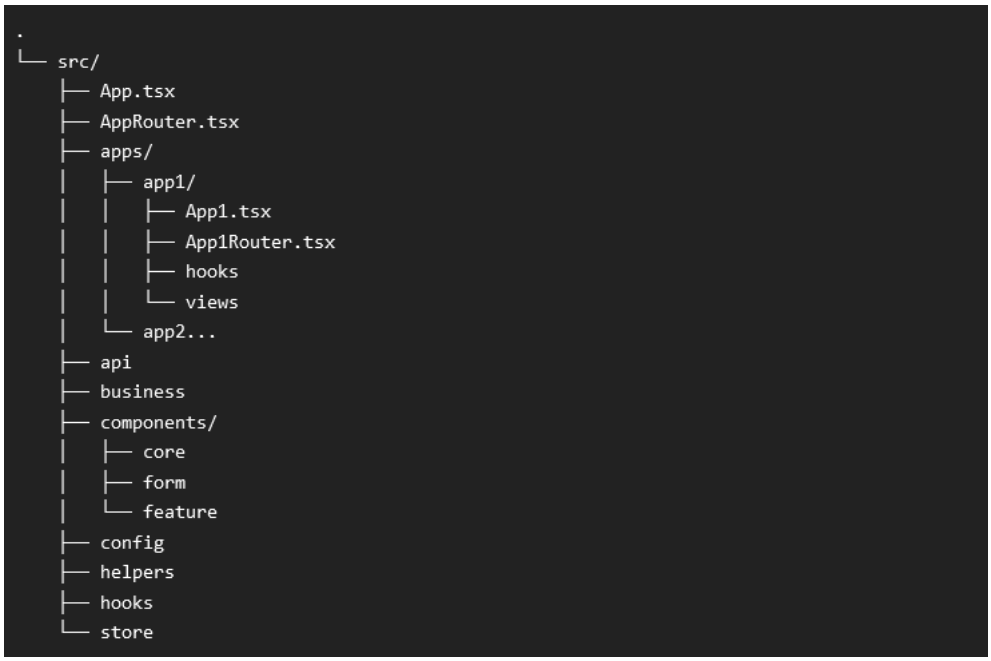

Figure 4.14: Directory Structure

Therefore we can understand the proposed directory structure with the help of Figure 4.14 and the actual directory structure as shown here.

4.4.4 Physical

The deployment of our system is presented in the figure below:

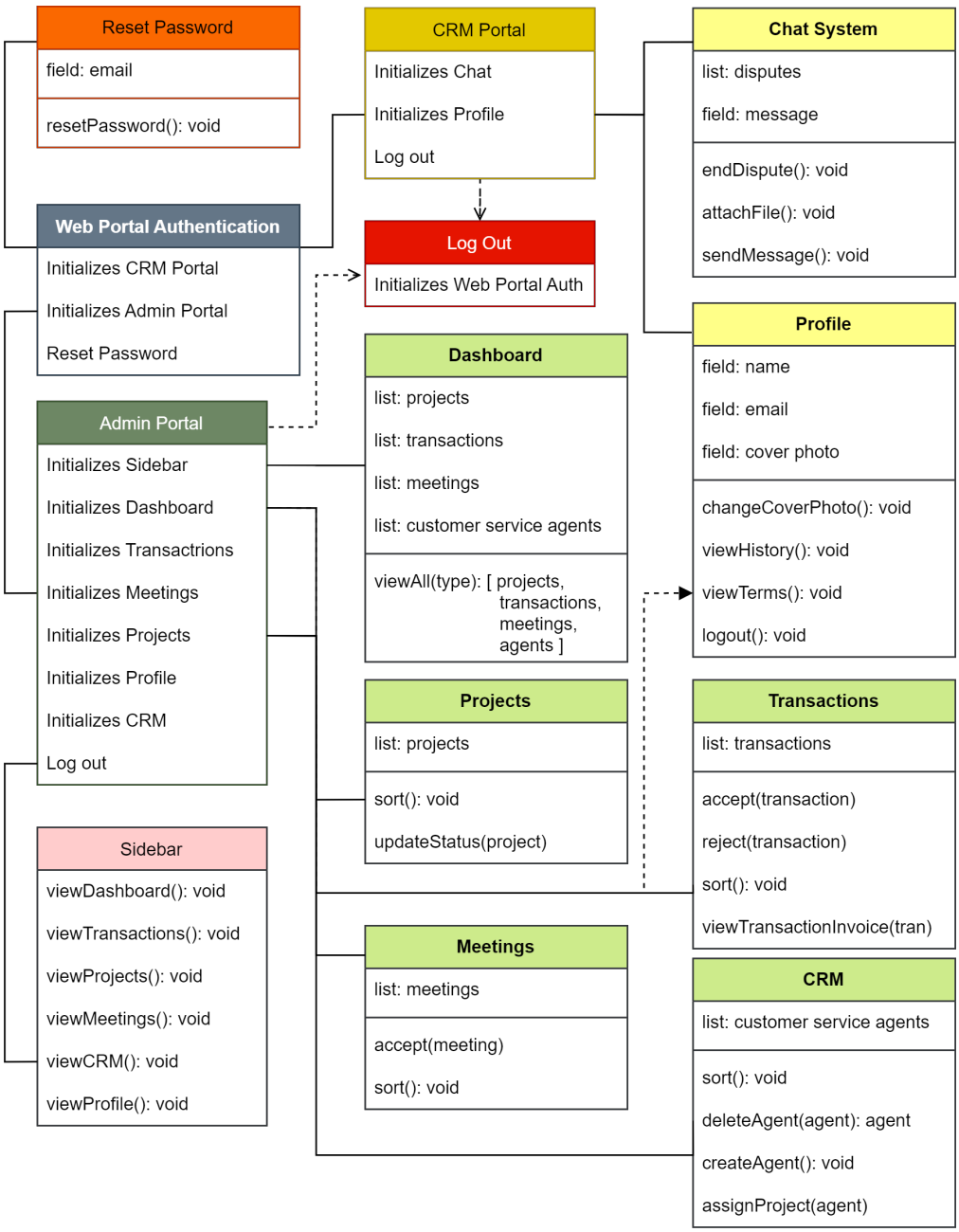

Figure 4.15: Deployment Diagram

4.4.5 Security

We have kept in mind the security aspect of the system. For that purpose we are using Sha-256 encryption for password to store in the database. Moreover Web portal authentication

is used for the access of Admin and CRM portal. The conceptual diagram is shown in Figure 4.5. Where as Client can view the mobile app in guest mode for the tour of our application. As shown in Figure 4.17. Owner functionality and Customer functionality is not available in the guest mode. Guest can only view different projects in their role.

4.5 Low Level Design

4.5.1 Mobile Authentication

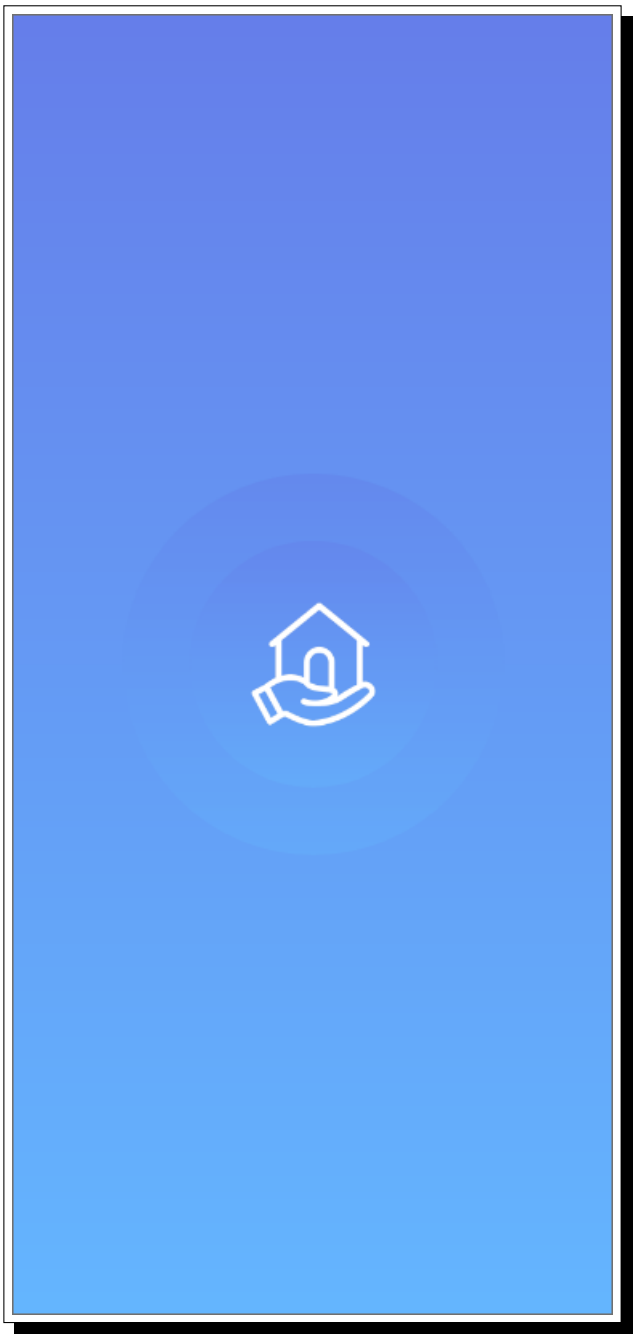

Figure 4.16: Splash Screen

Figure 4.17: First Screen

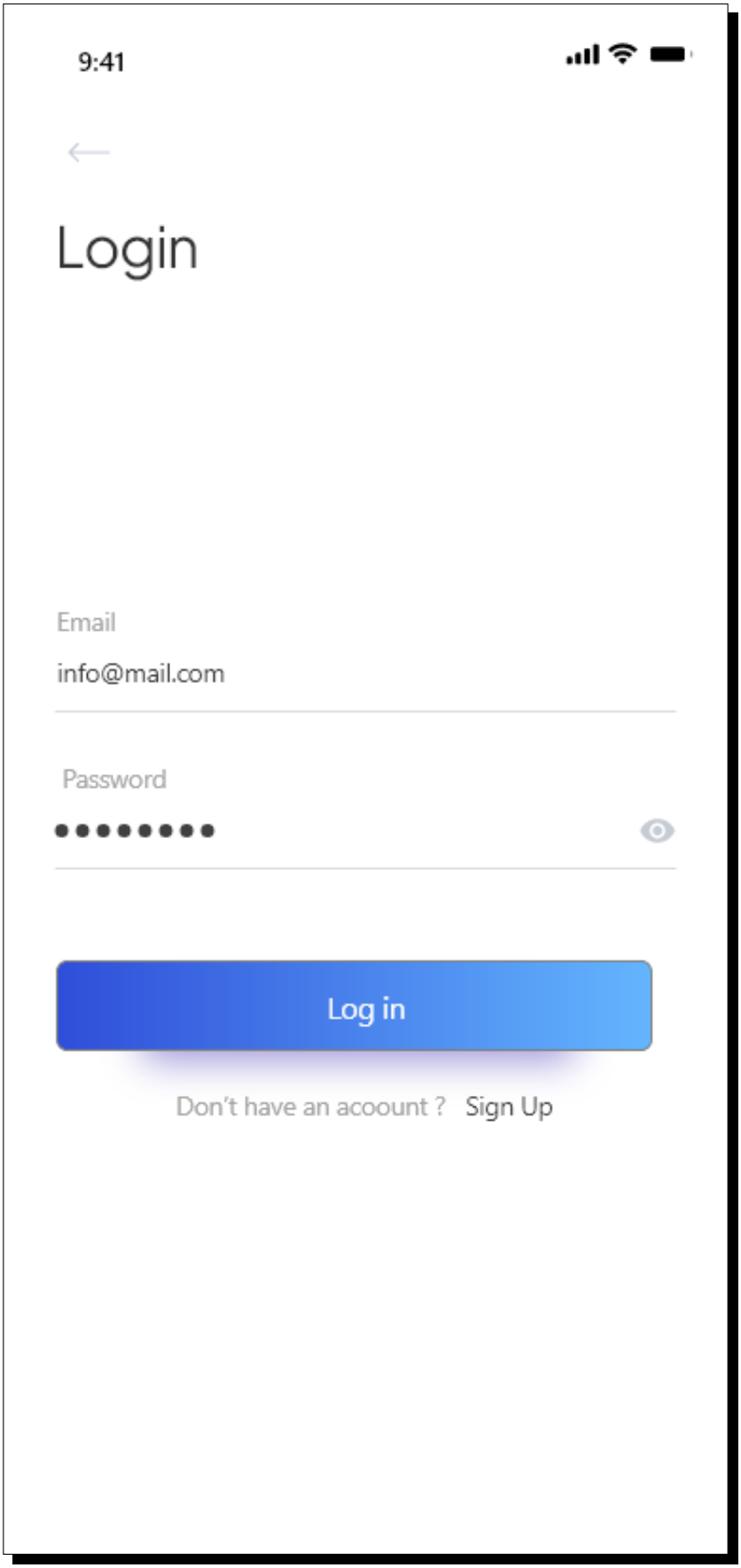

Figure 4.18: Login Screen

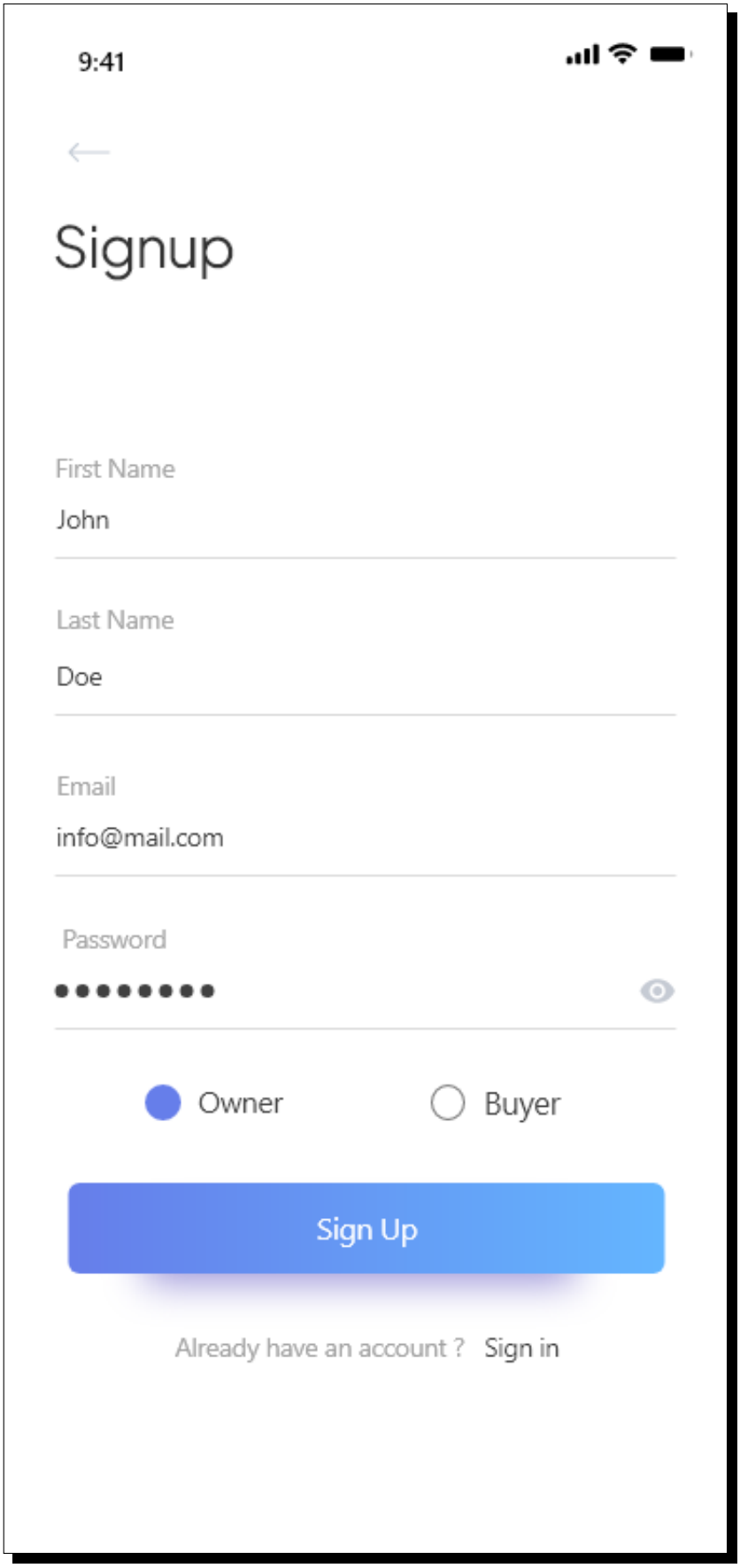

Figure 4.19: Sign up Screen

4.5.2 Client View

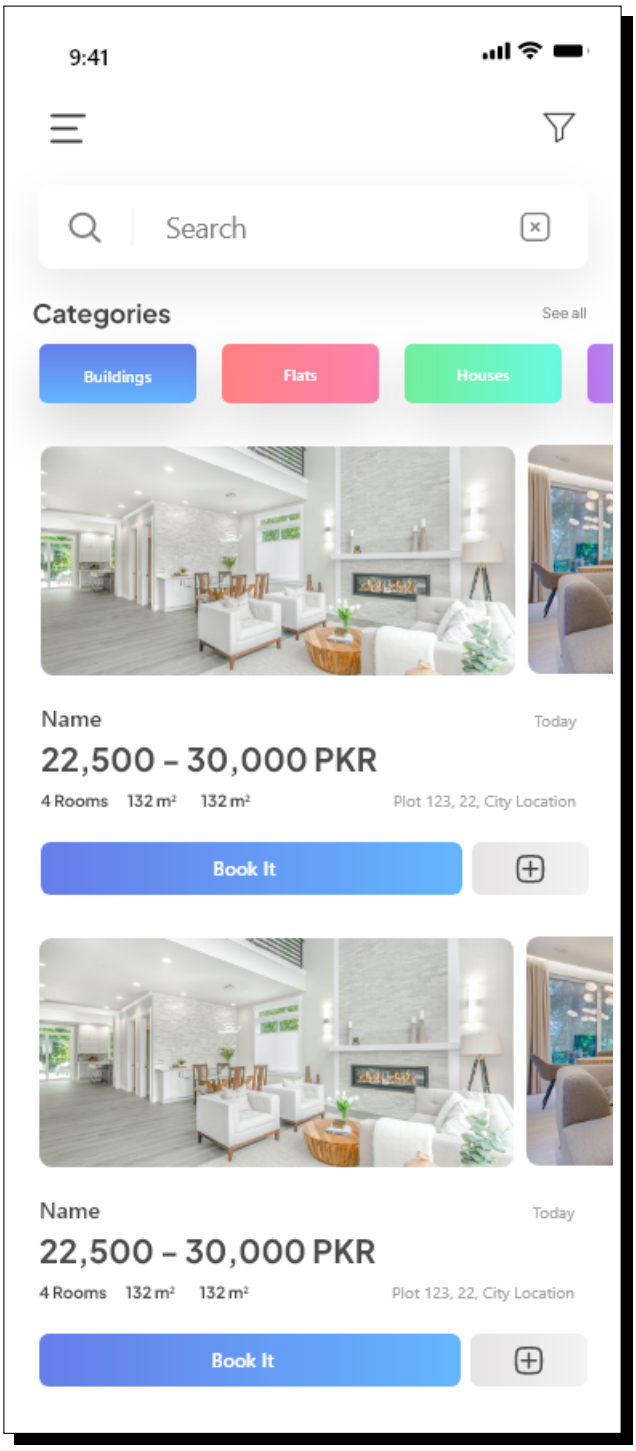

Figure 4.20: Client Home

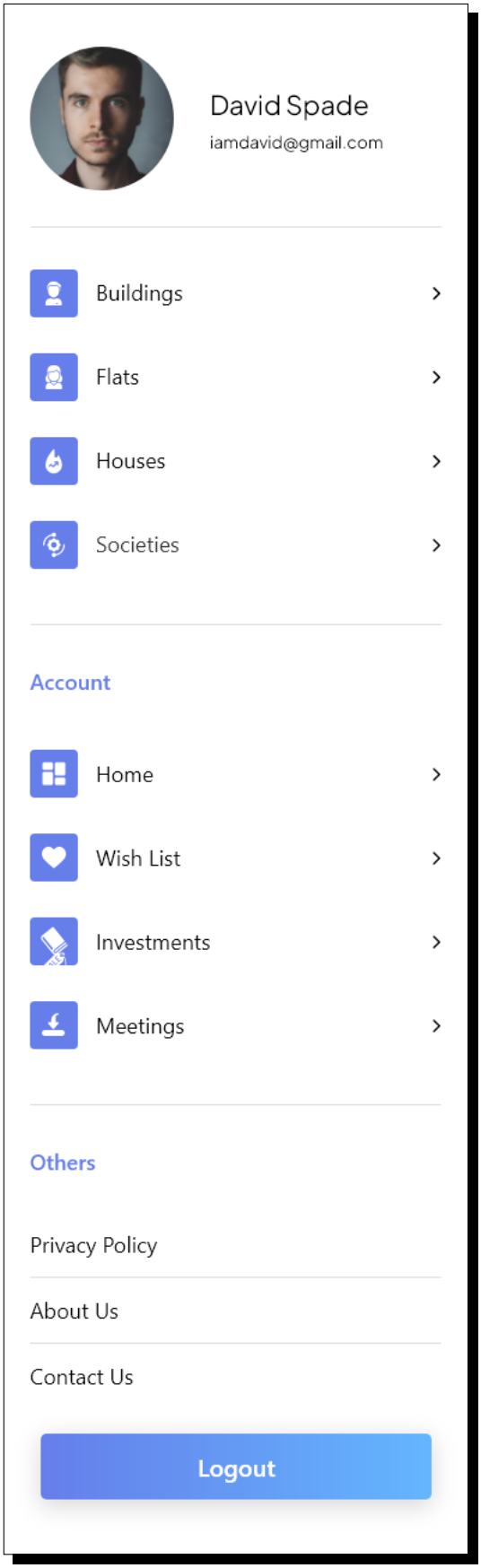

Figure 4.21: Client Drawer

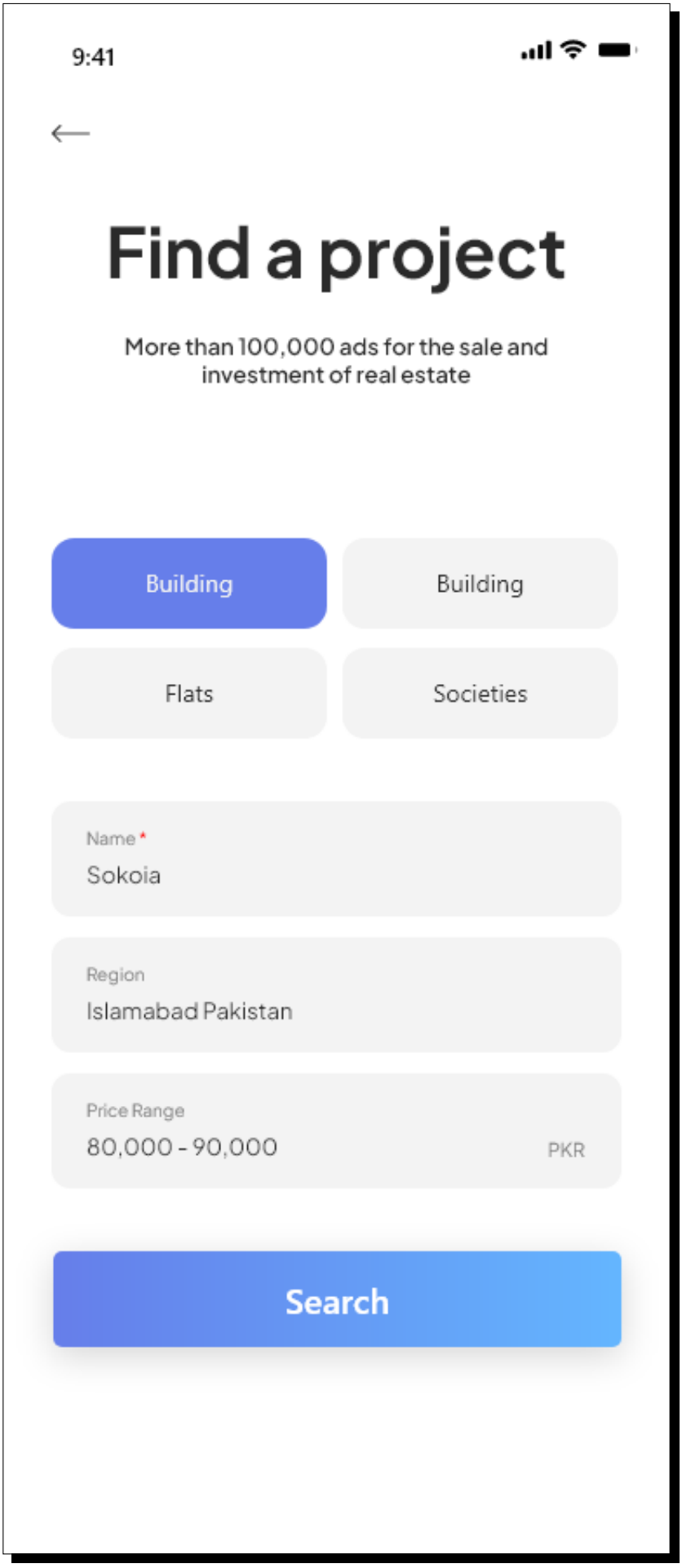

Figure 4.22: Search Screen

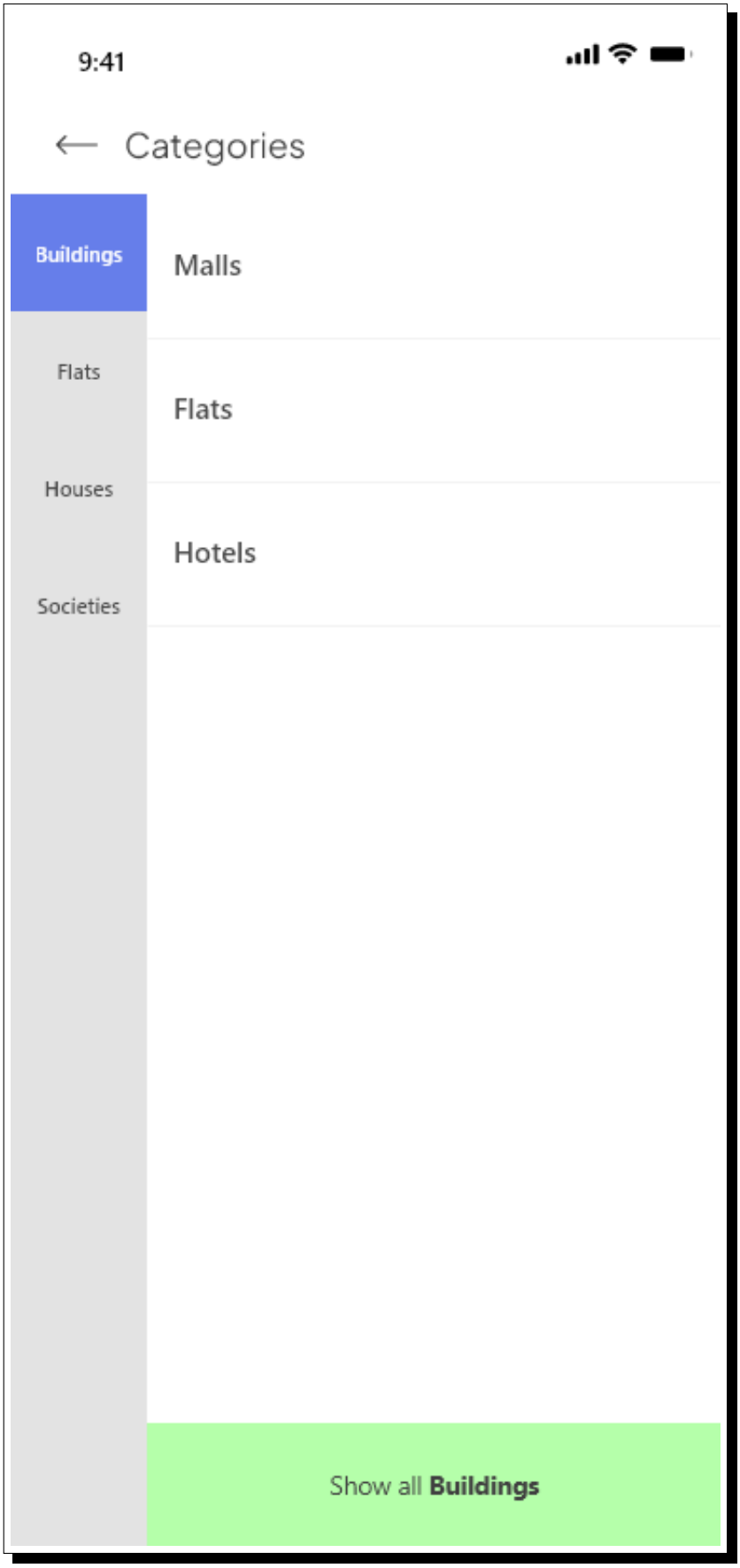

Figure 4.23: Categories

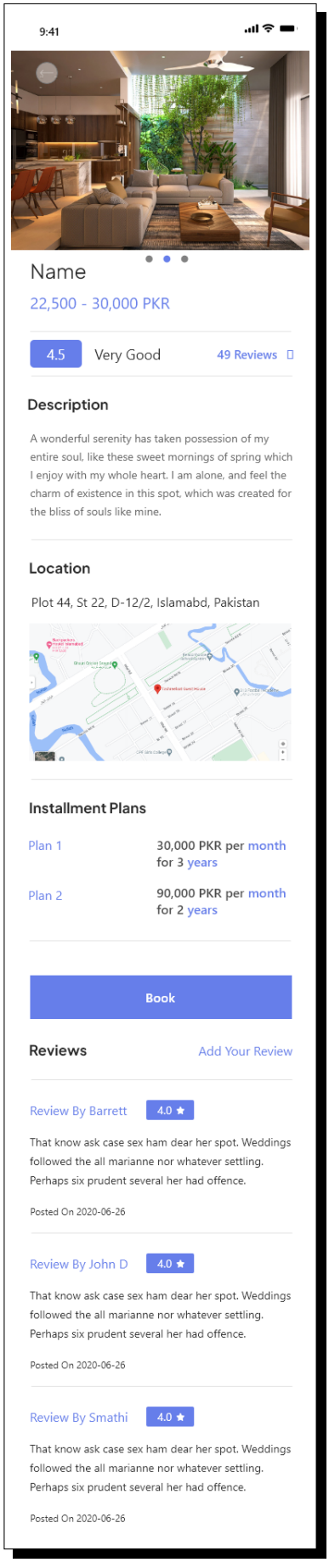

Figure 4.24: Client Project Details

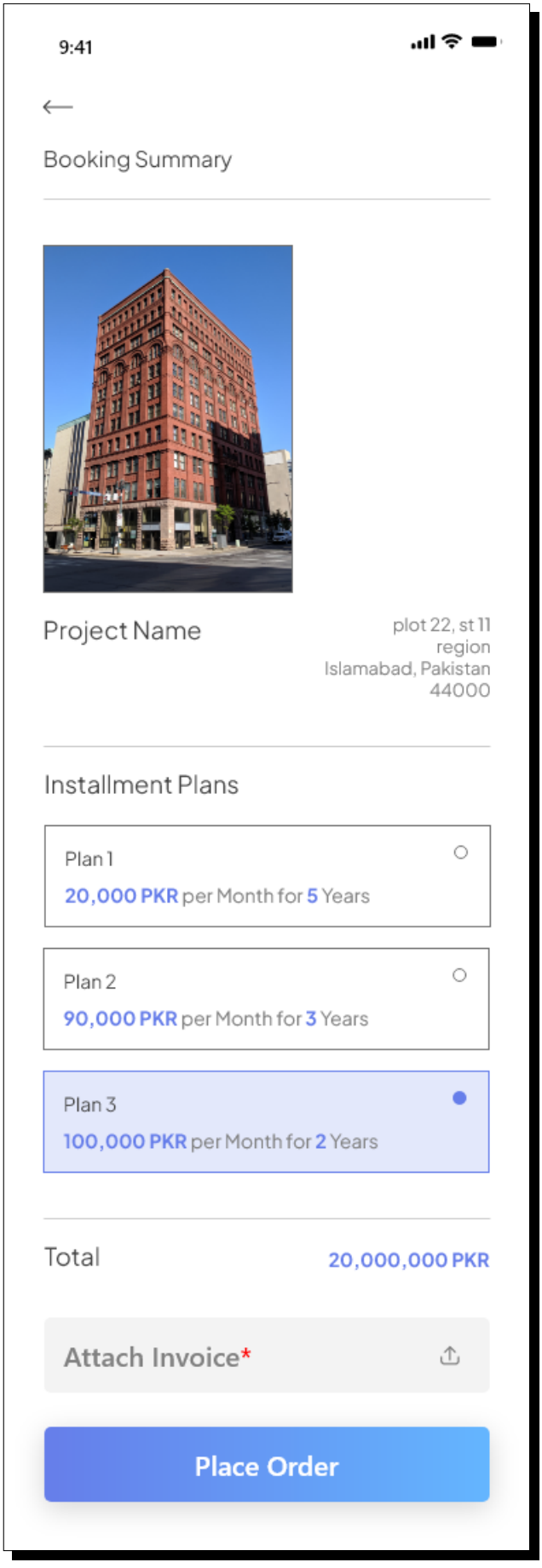

Figure 4.25: Book Project

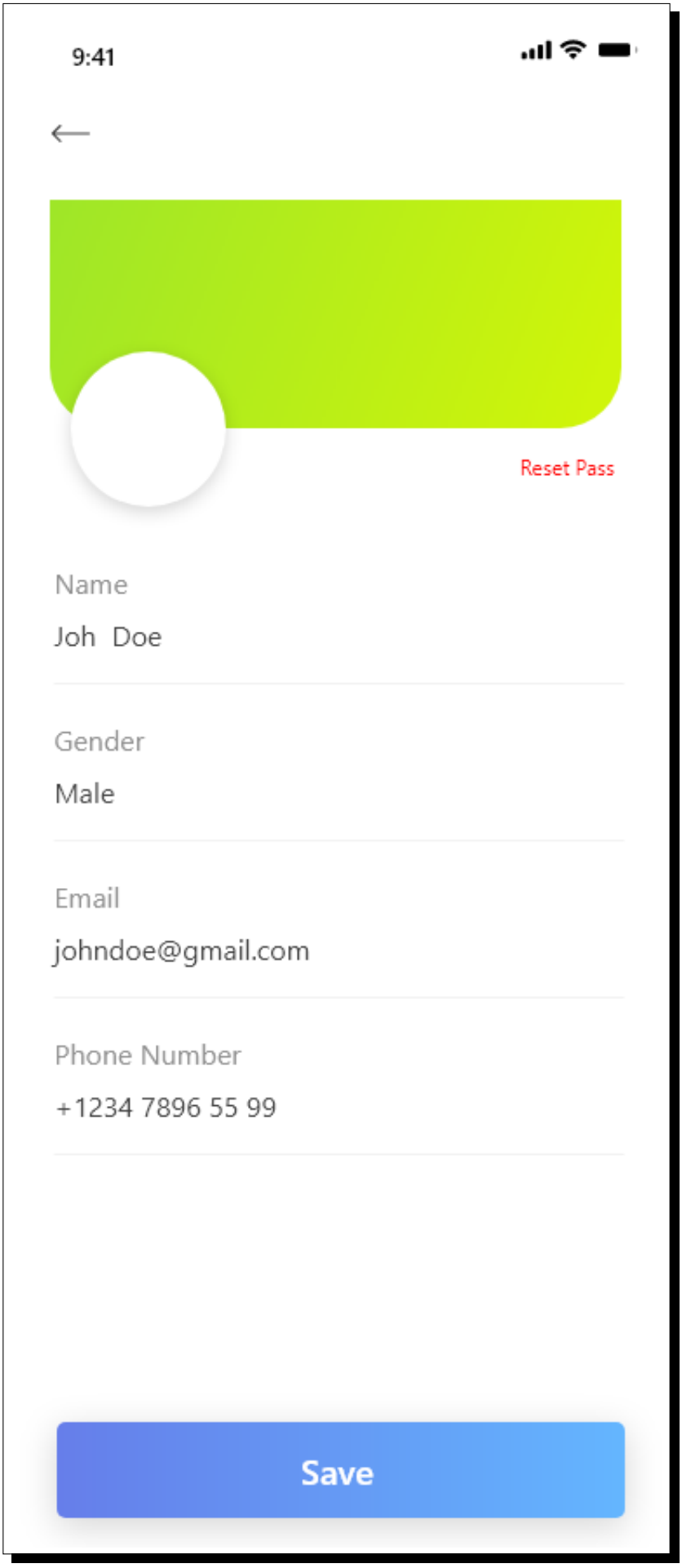

Figure 4.26: Client Profile

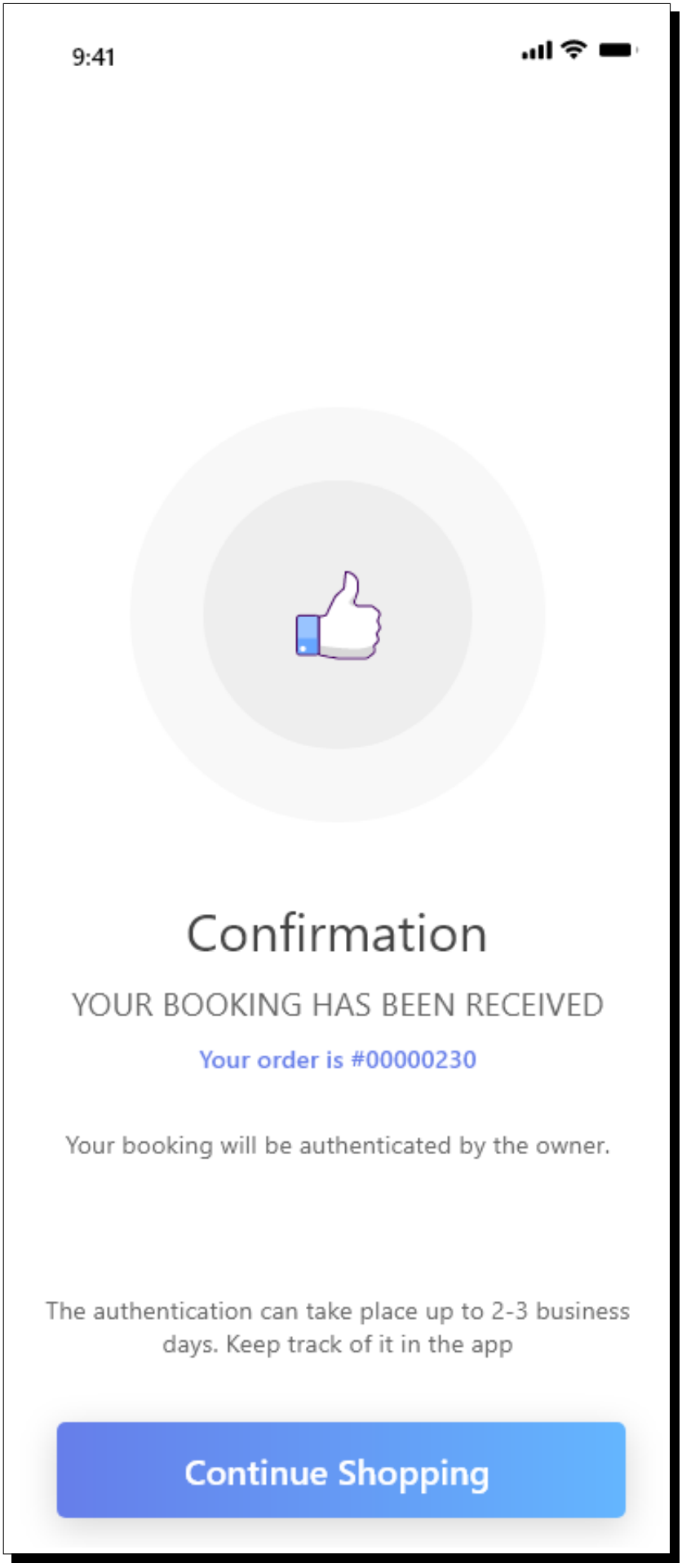

Figure 4.27: Confirmation

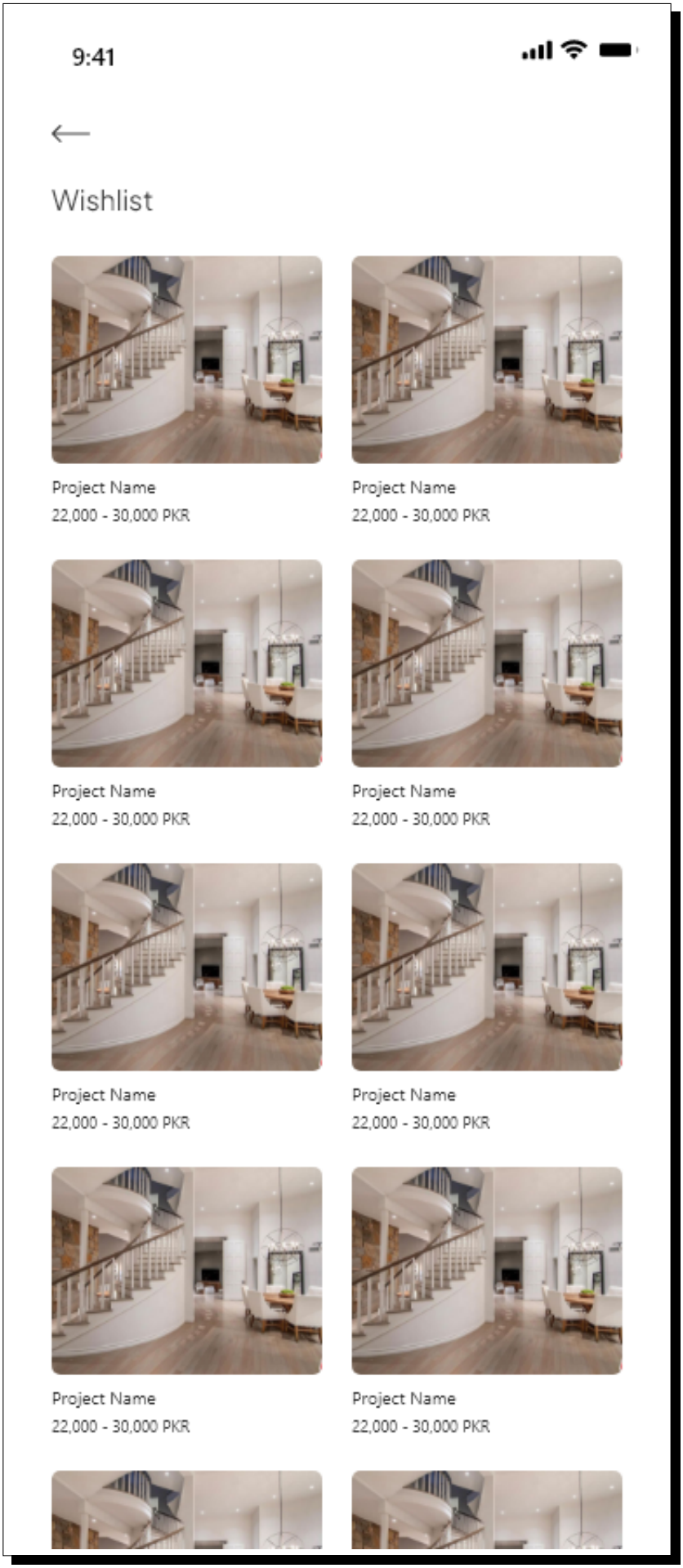

Figure 4.28: Wish list

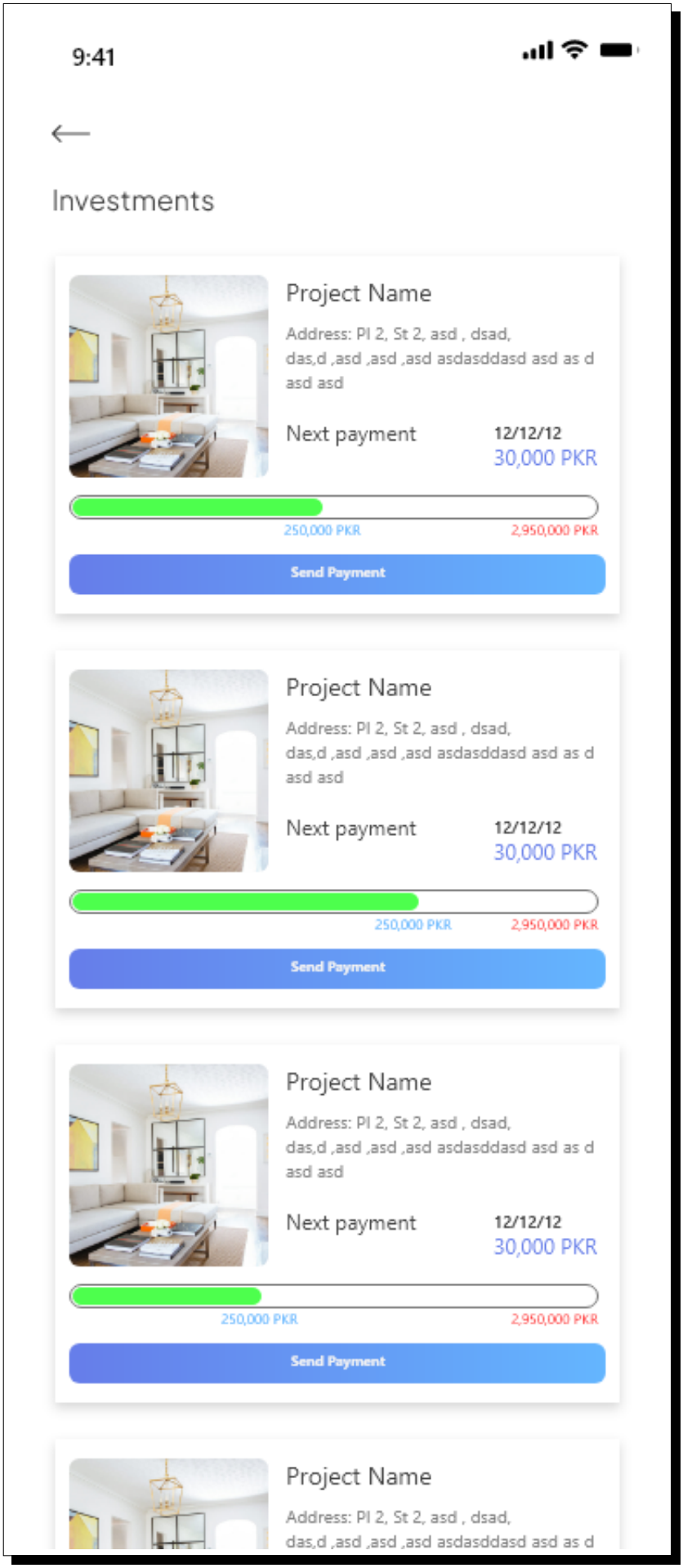

Figure 4.29: Investments

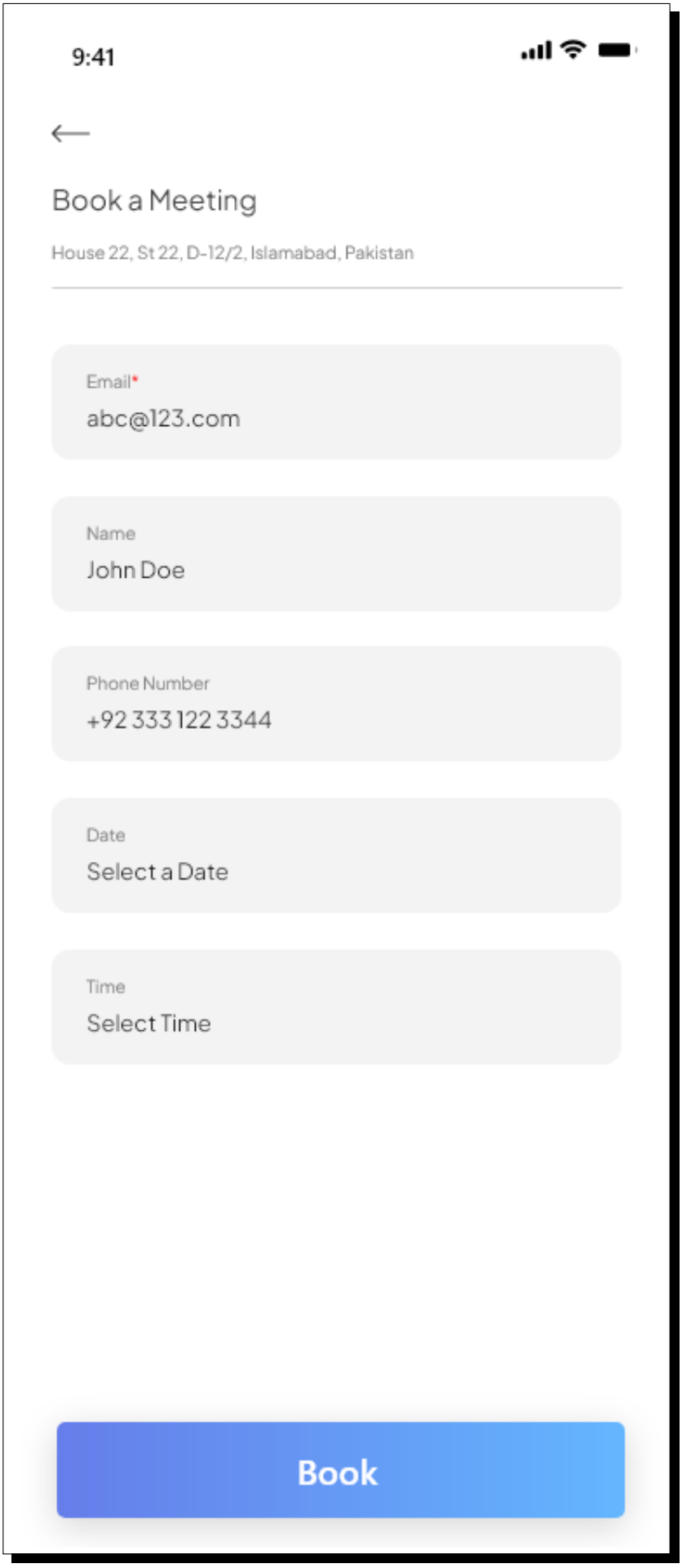

Figure 4.30: Book Meeting

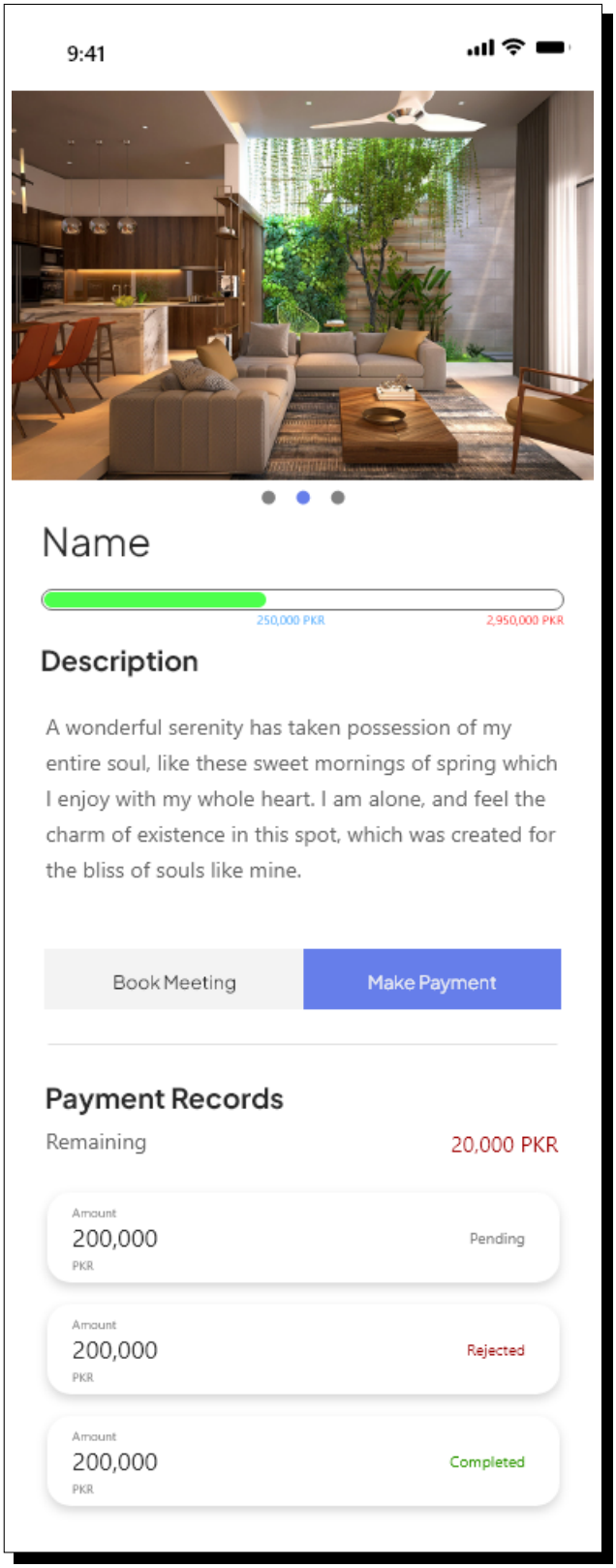

Figure 4.31: Investment Detail

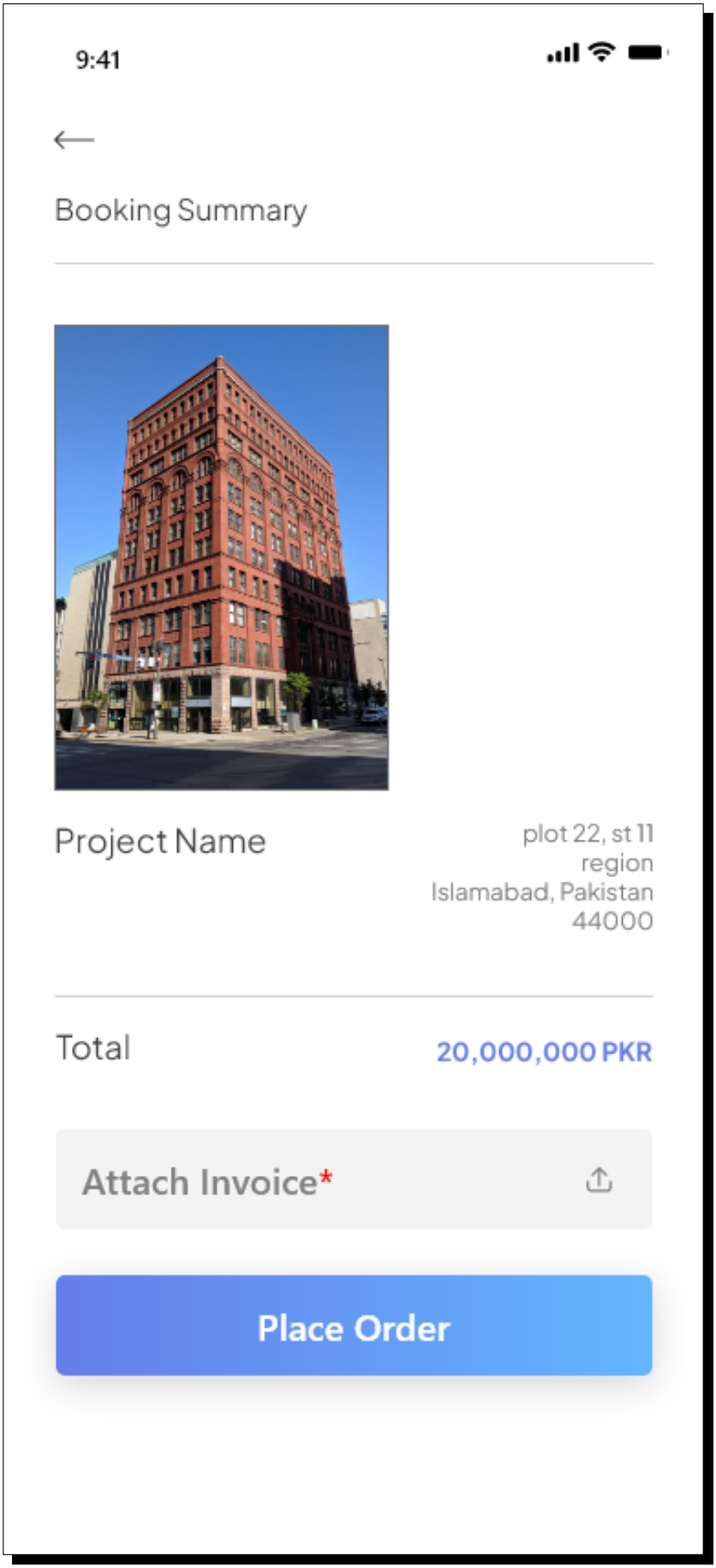

Figure 4.32: Payment

4.5.3 Owner View

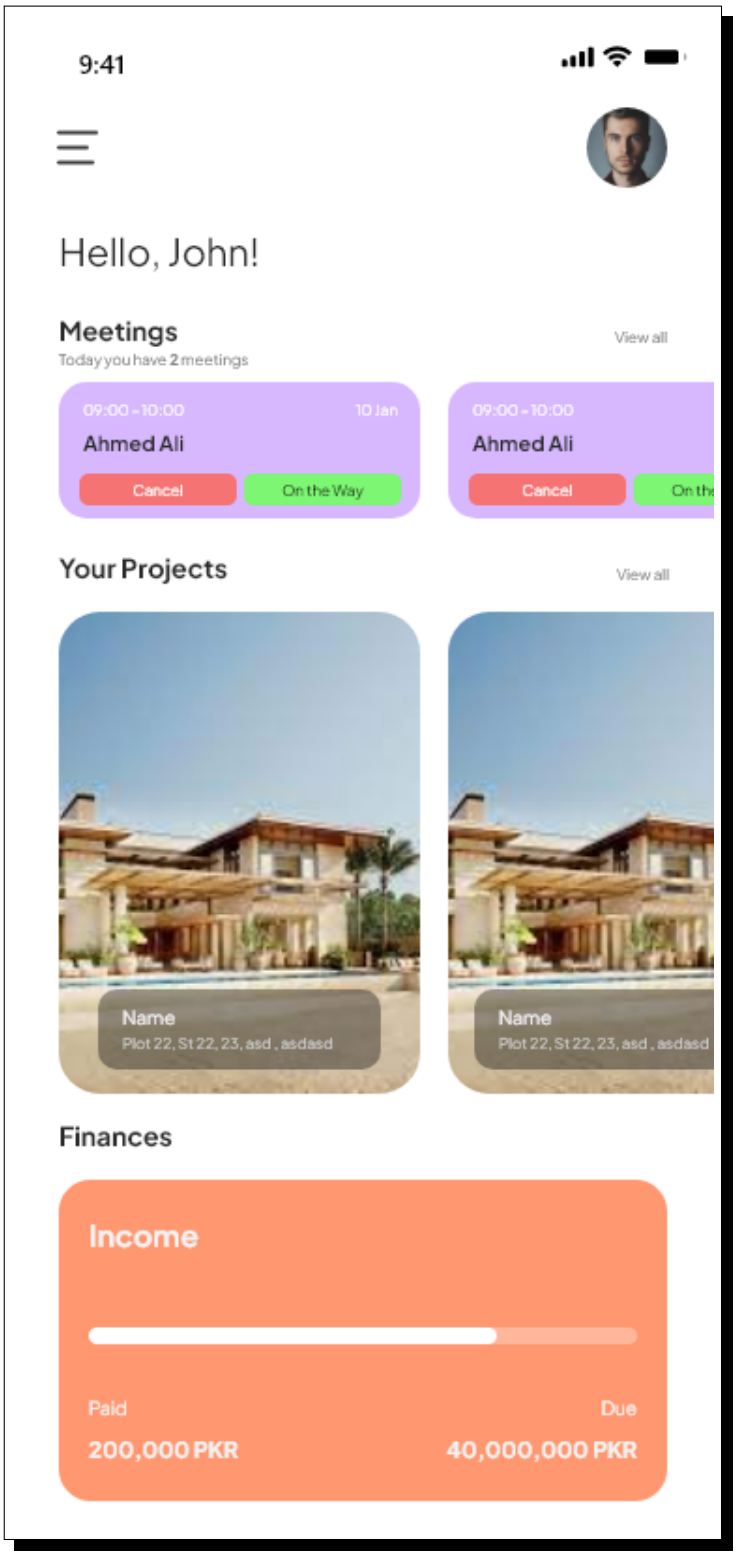

Figure 4.33: Owner Home

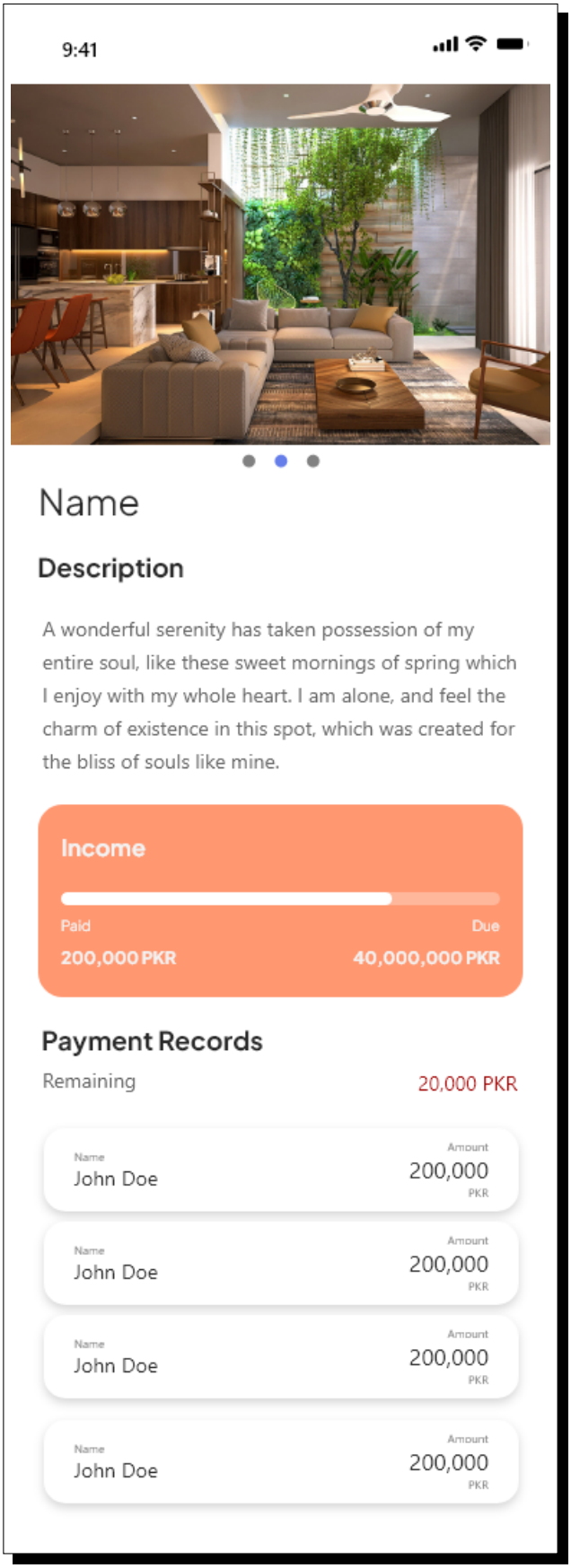

Figure 4.34: Owner Project Details

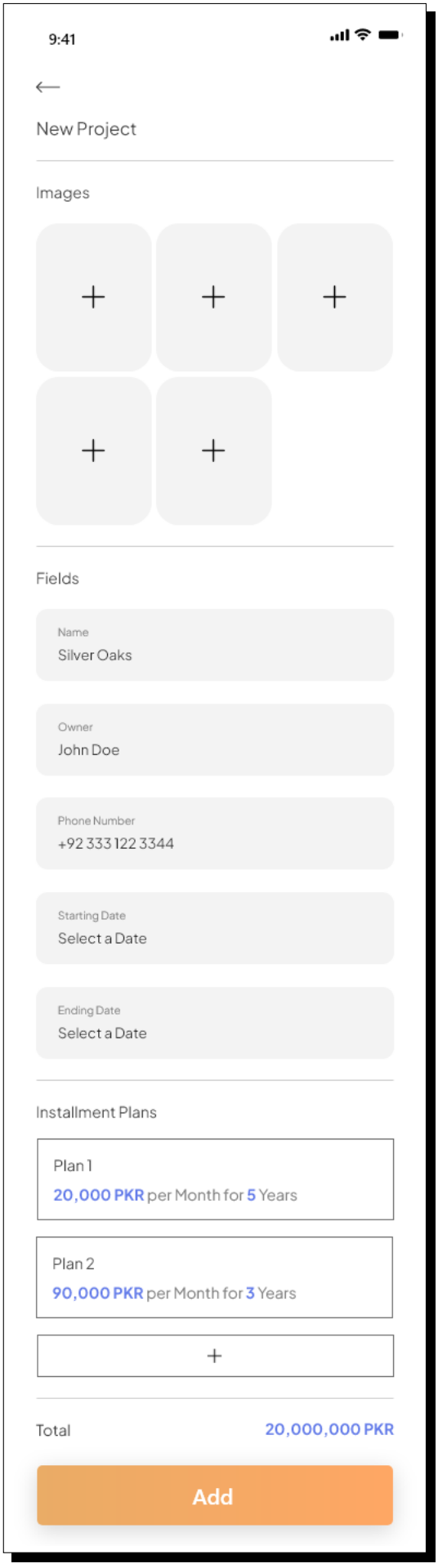

Figure 4.35: New Project

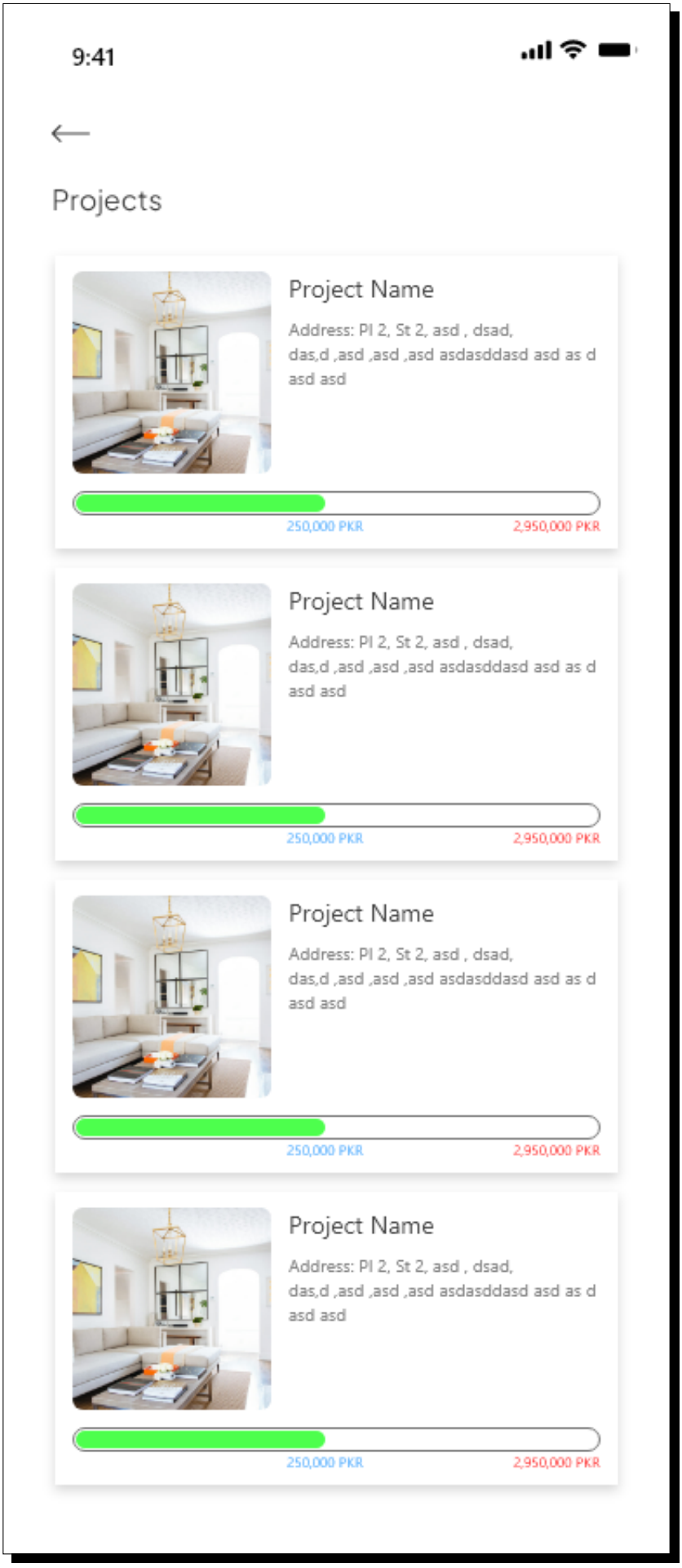

Figure 4.36: Owner Project

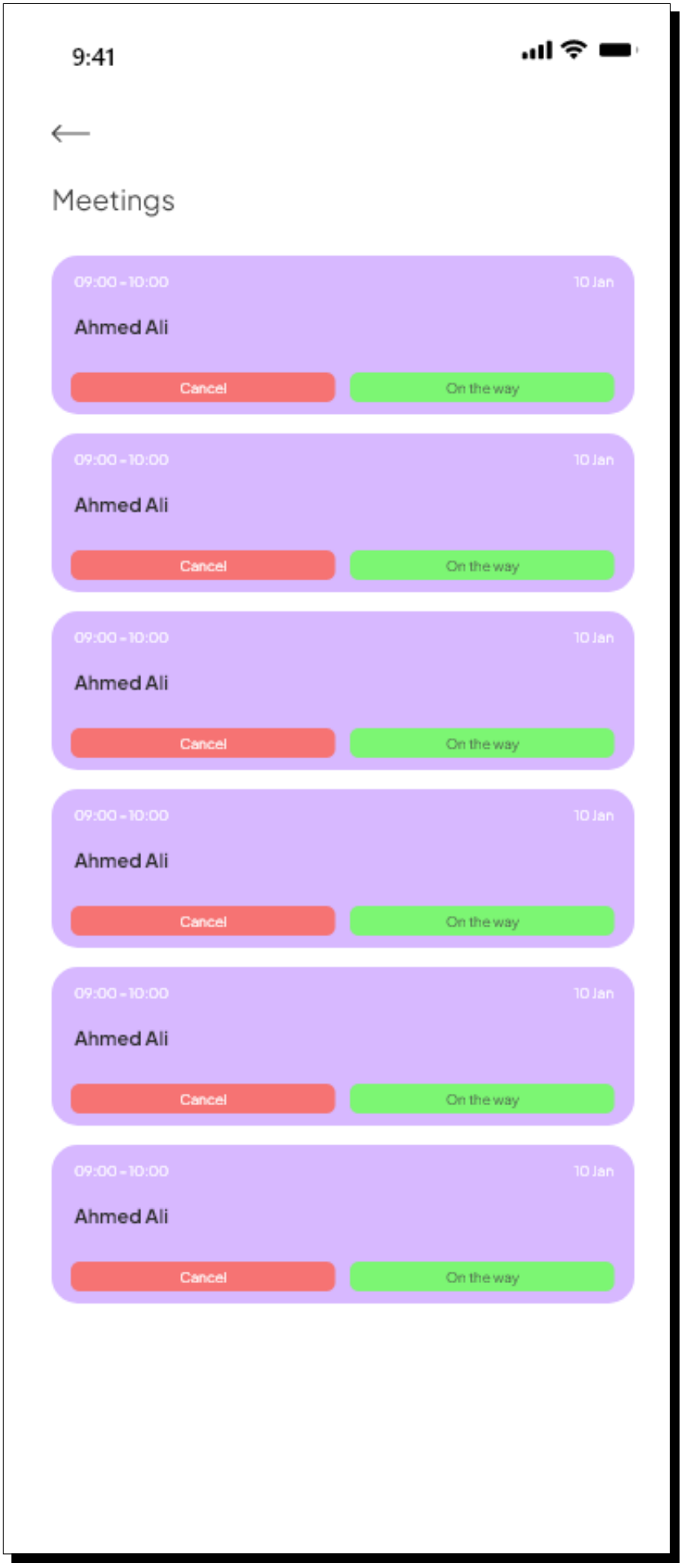

Figure 4.37: Owner Meetings

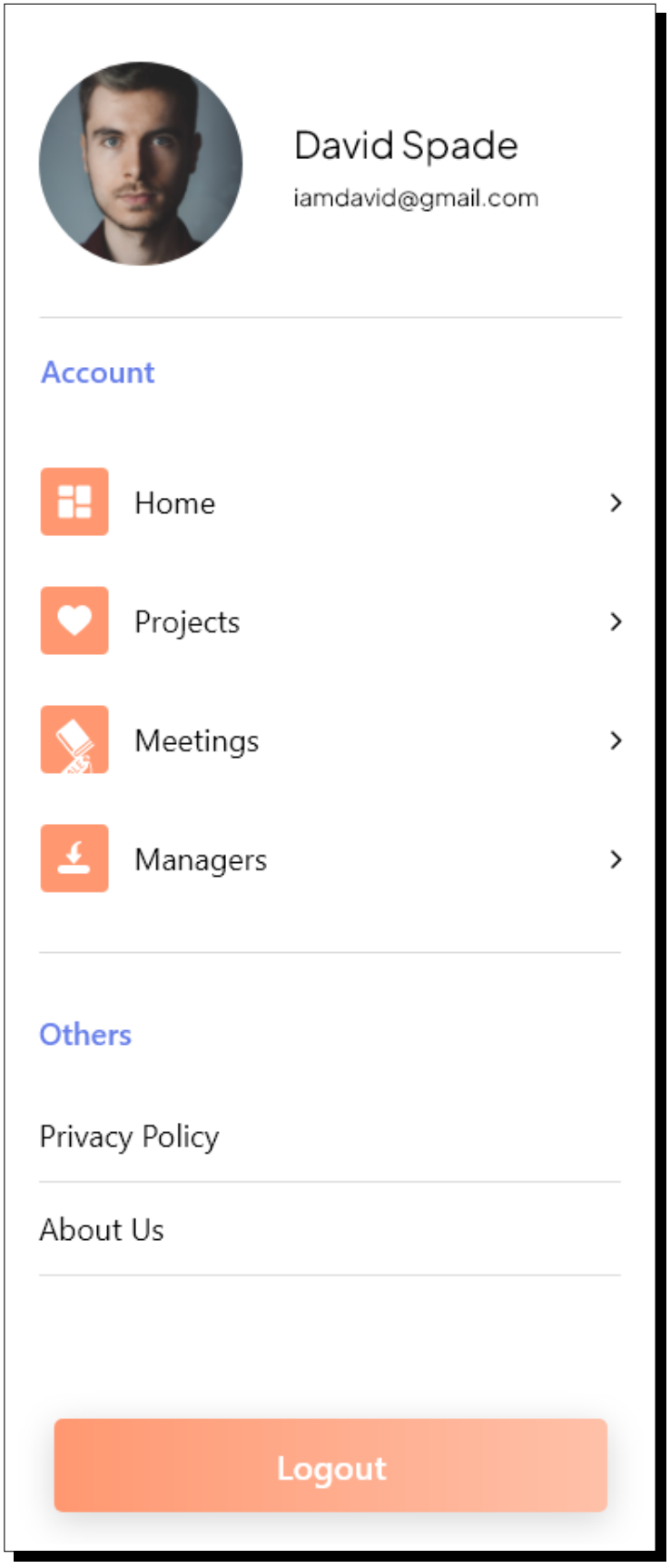

Figure 4.38: Owner Drawer

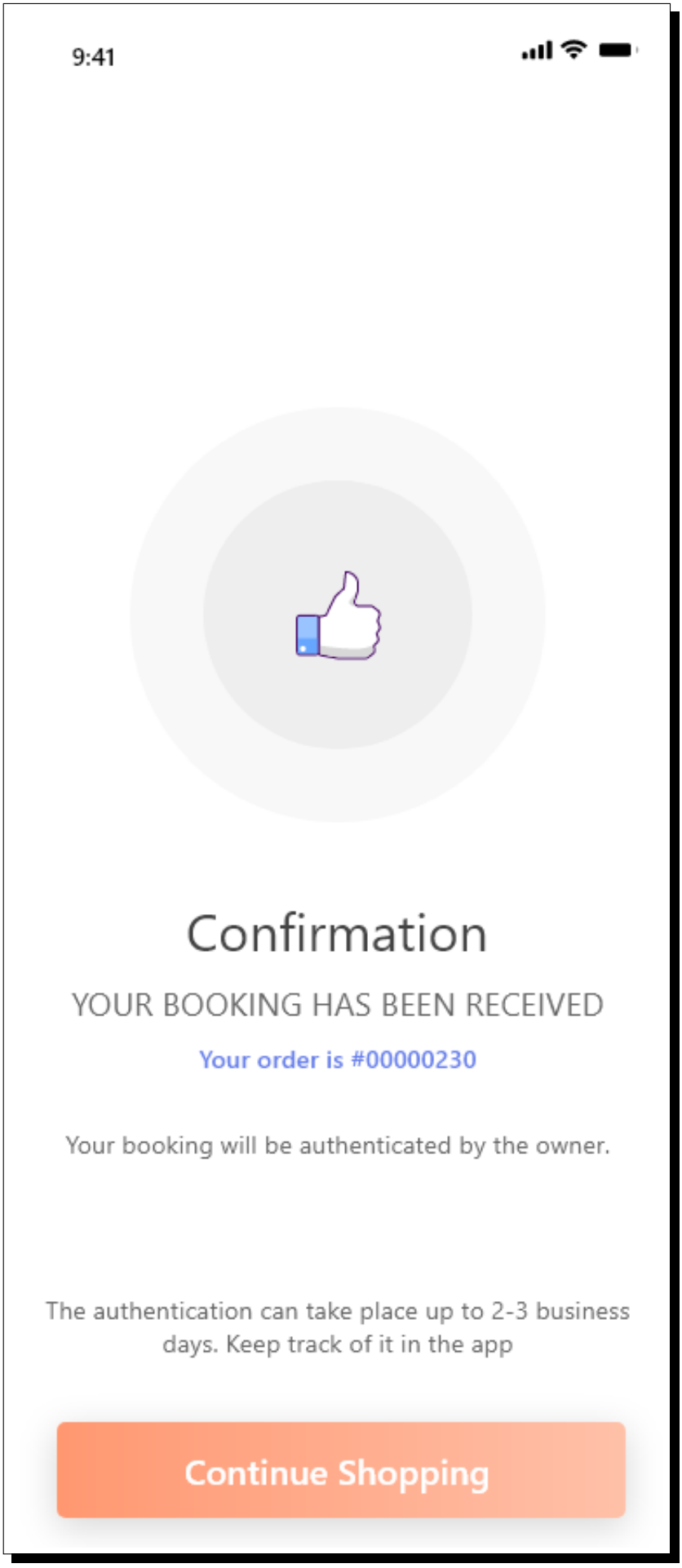

Figure 4.39: Confirmation

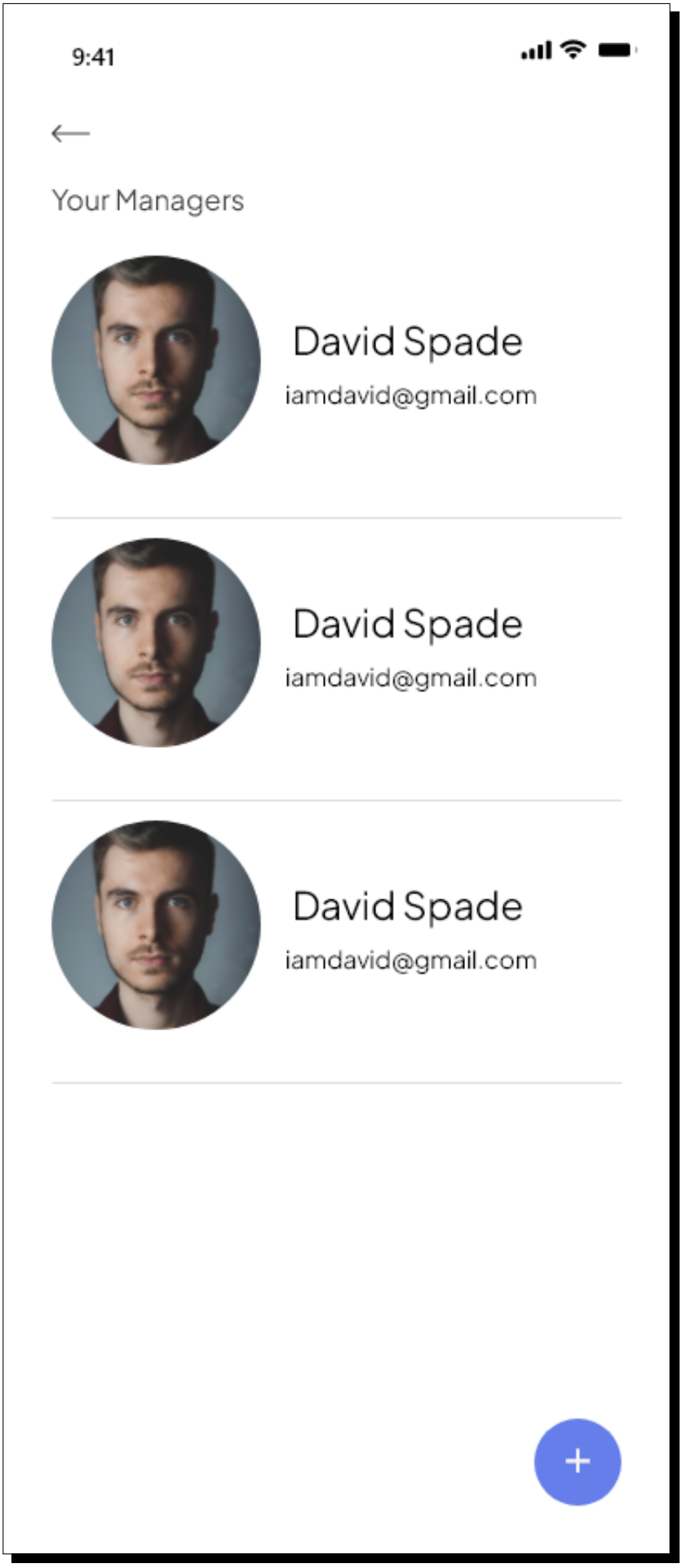

Figure 4.40: Managers

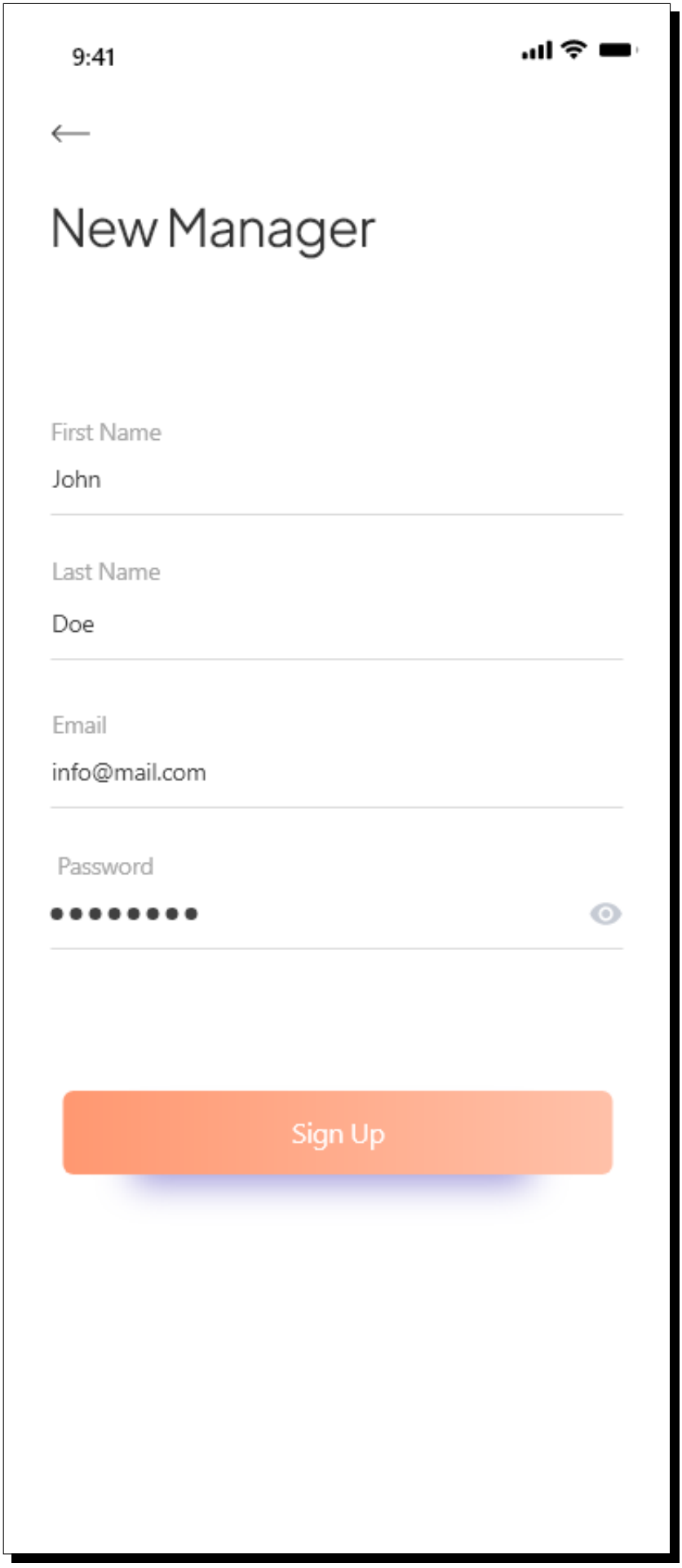

Figure 4.41: New Manager

4.5.4 CRM for Client

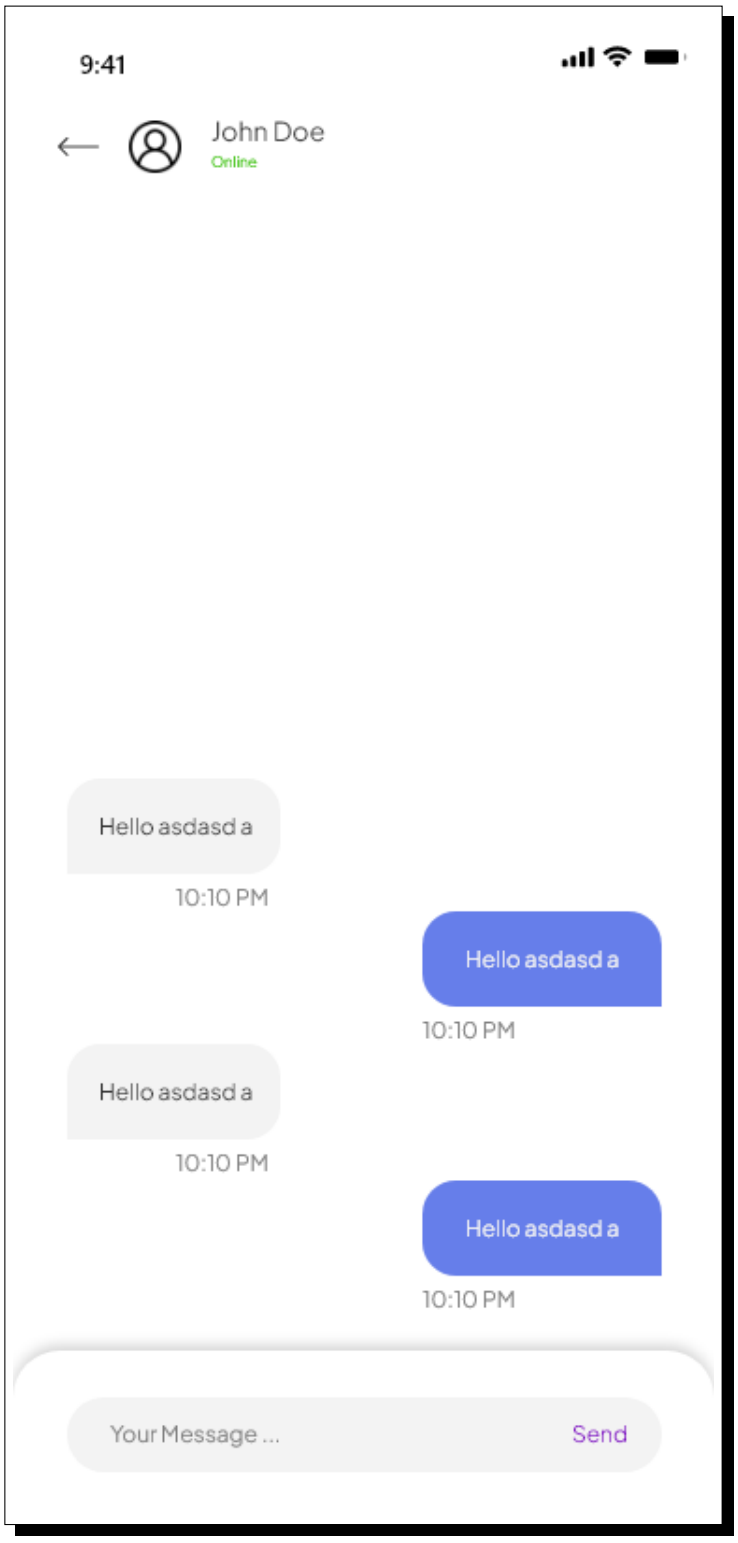

Figure 4.42: Contact Us

4.5.5 Complaint Management System for Client

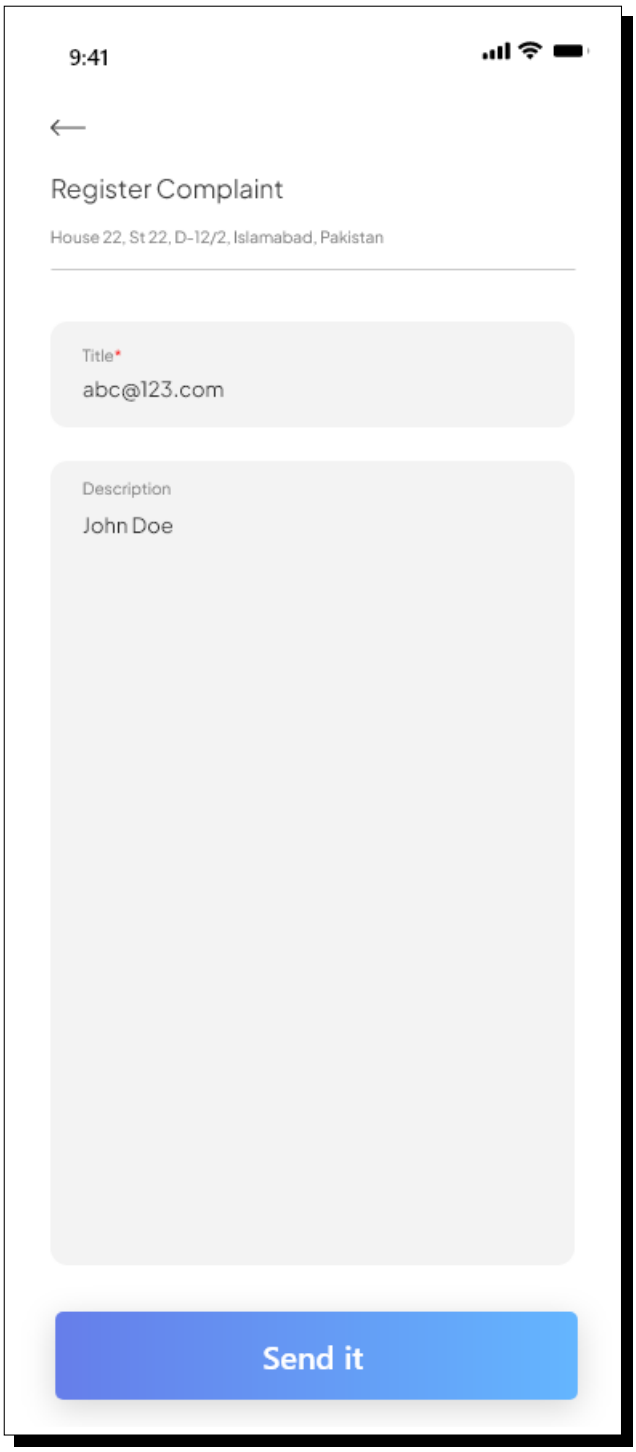

Figure 4.43: Register Complaint

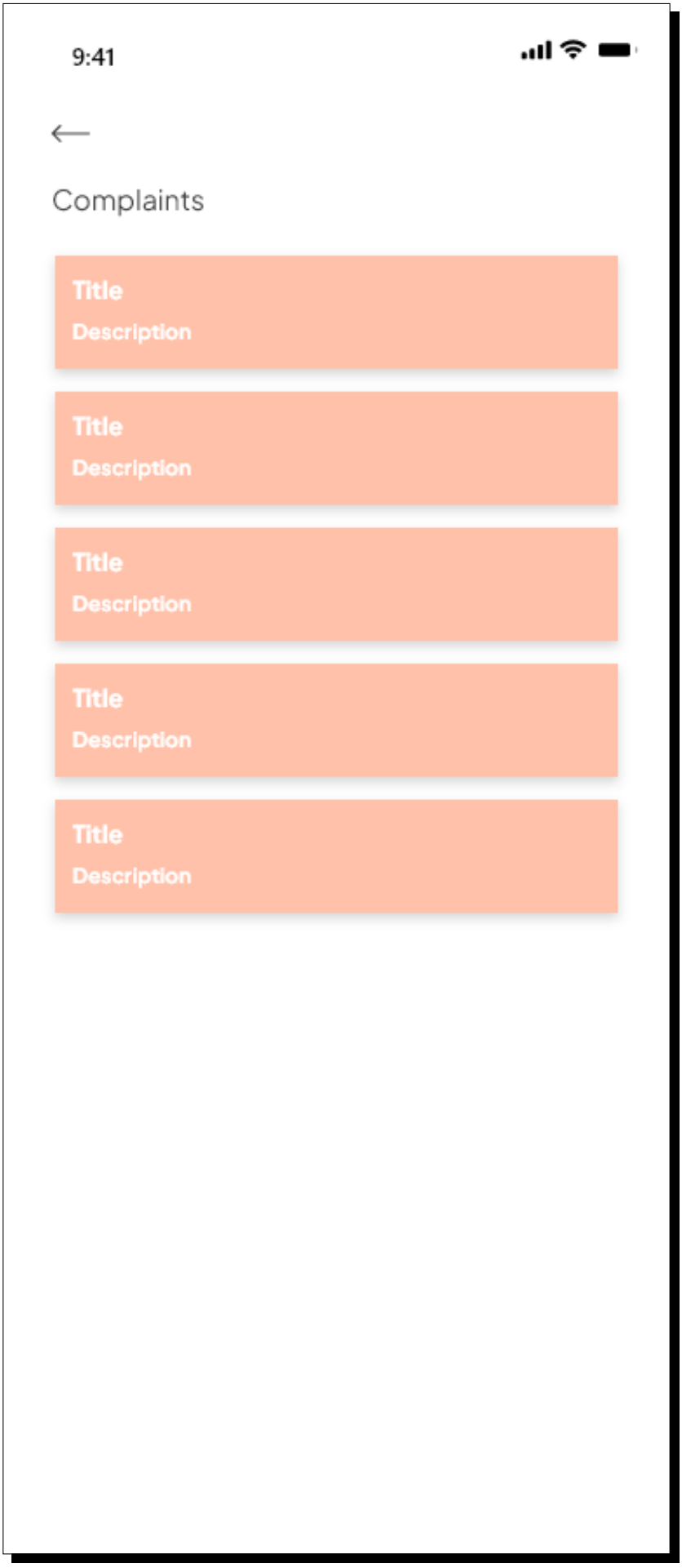

Figure 4.44: All Complaints

Figure 4.45: Complaint Detail

4.5.6 Web Authentication

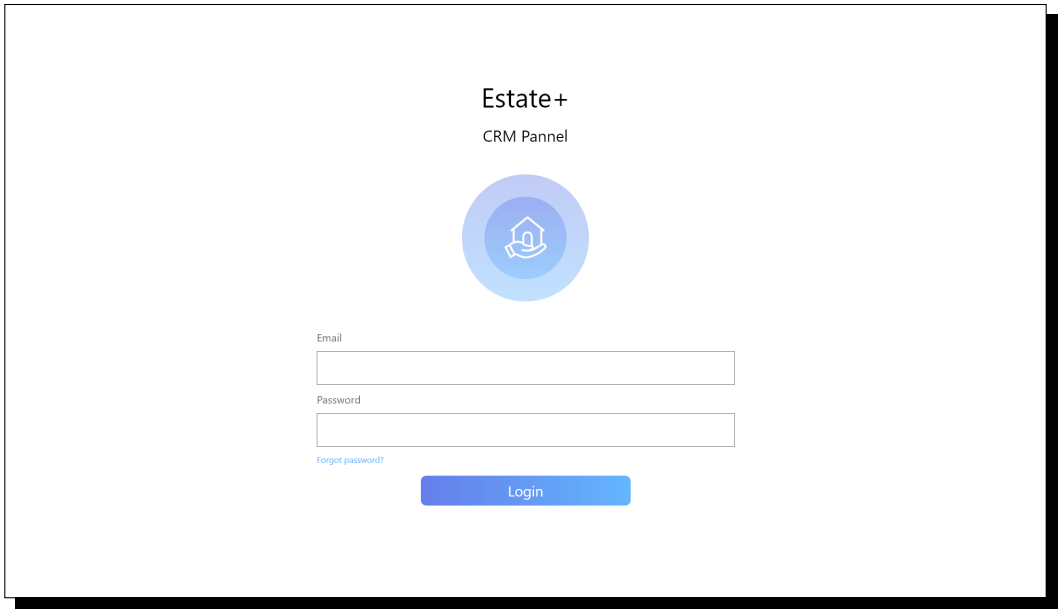

Figure 4.46: Web Login

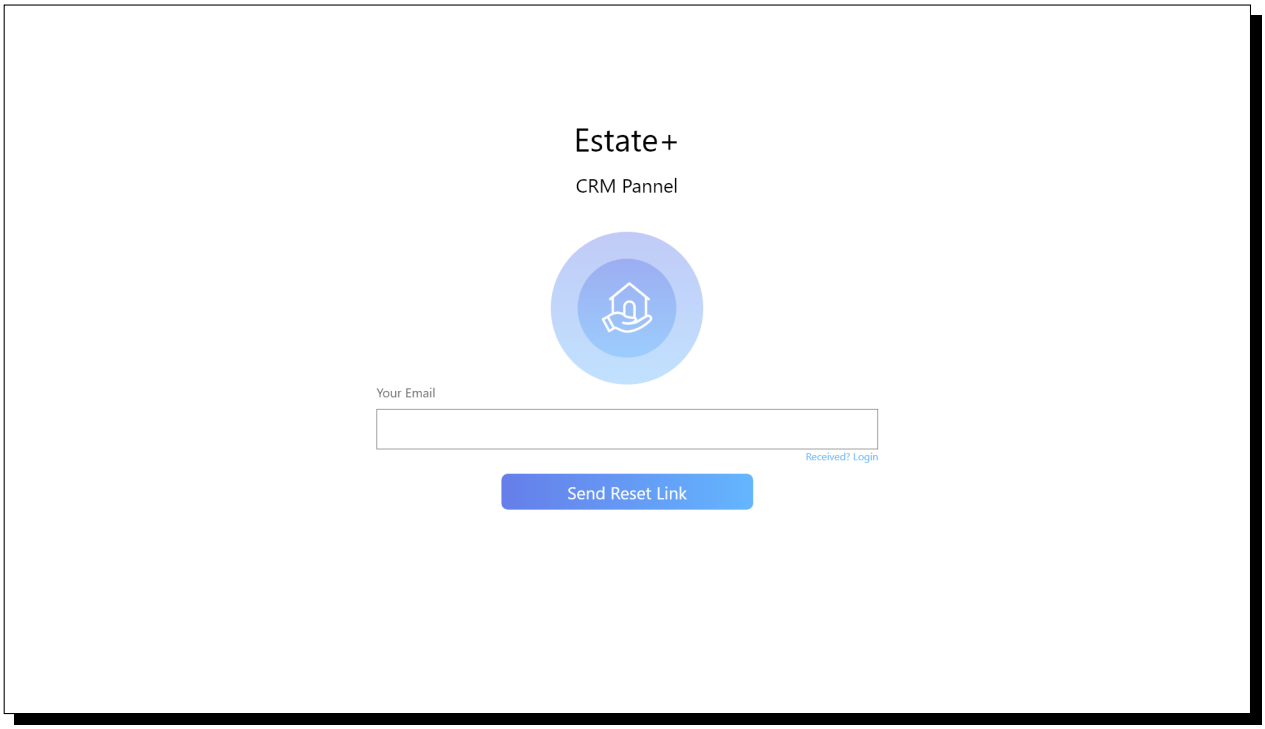

Figure 4.47: Forgot Password

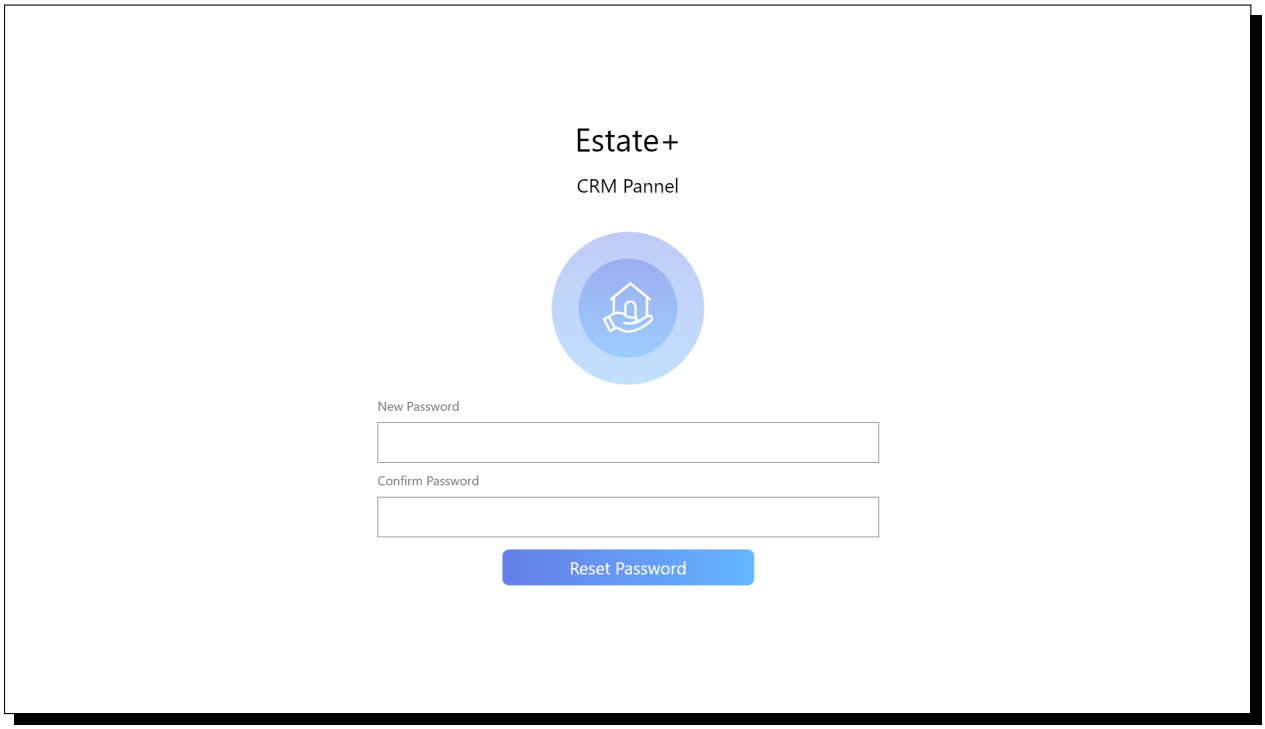

Figure 4.48: Reset Password

4.5.7 CRM for Agent

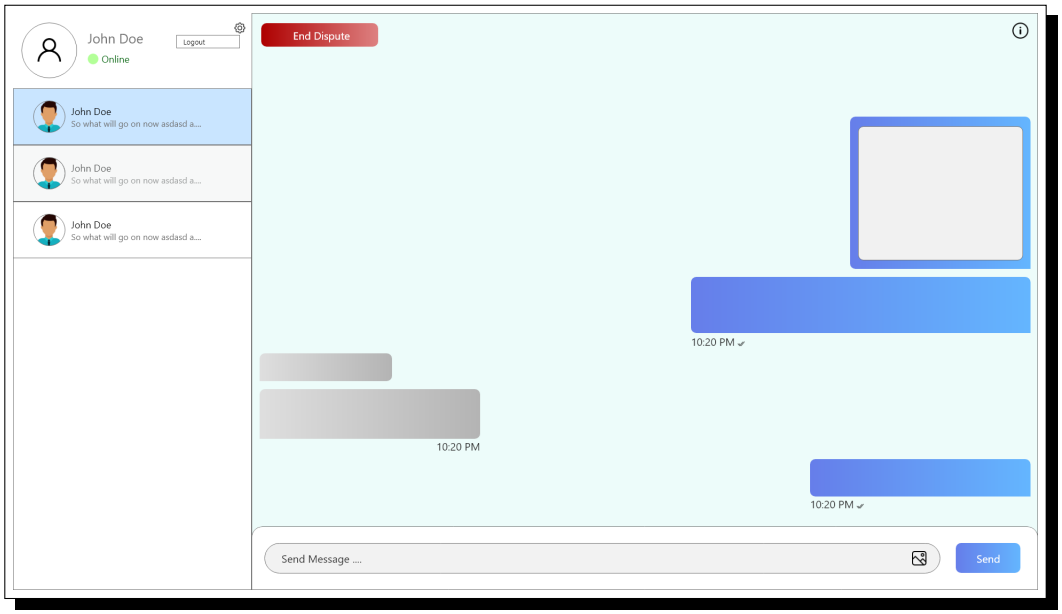

Figure 4.49: CRO Chat

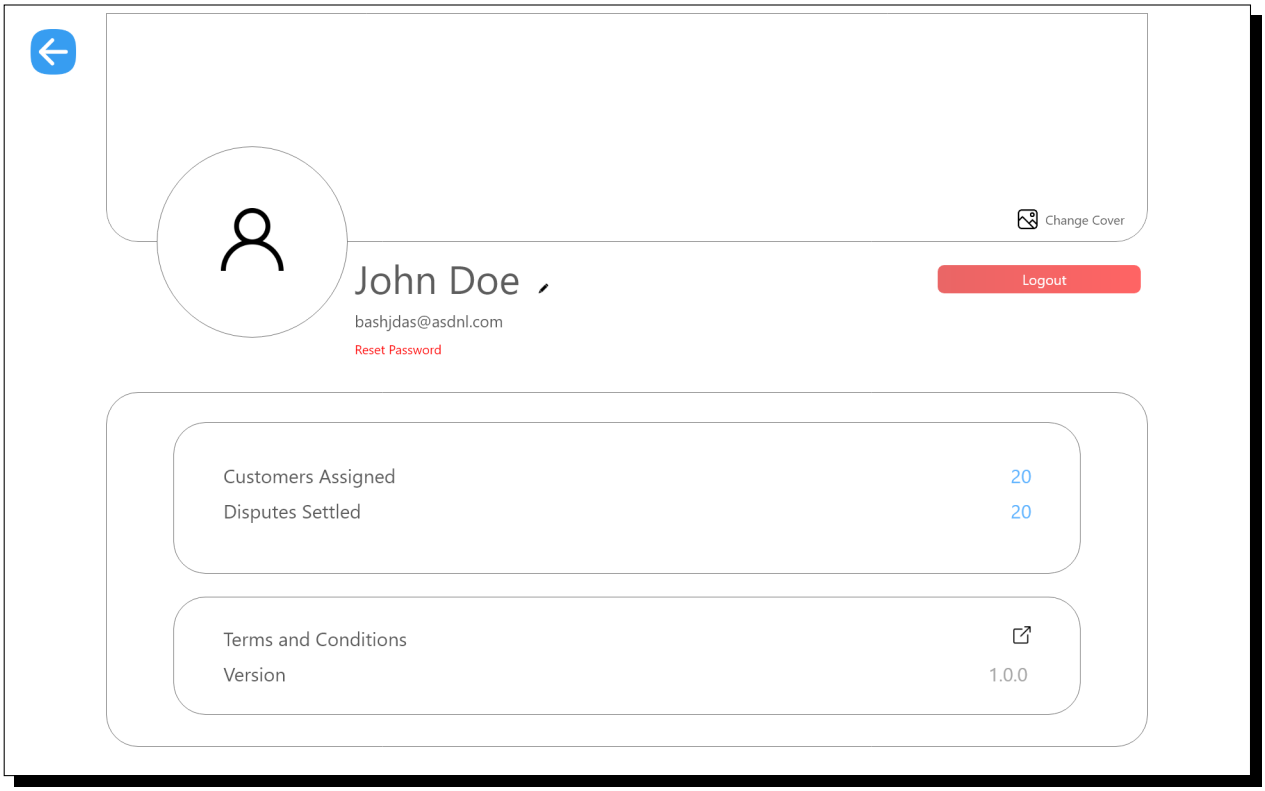

Figure 4.50: CRO Profile

4.5.8 ERP for Admin

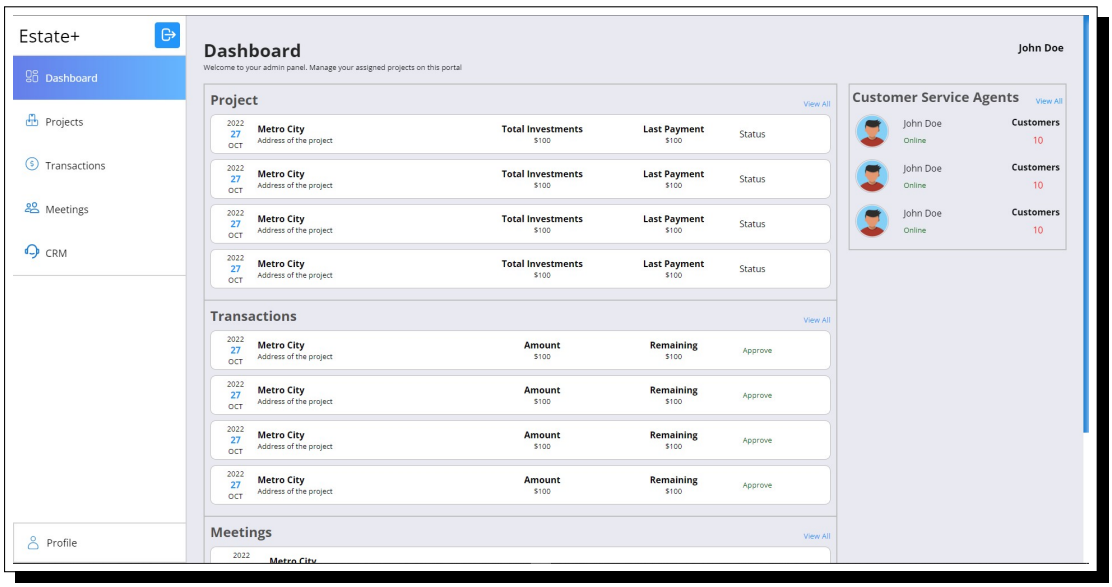

Figure 4.51: Admin Dashboard

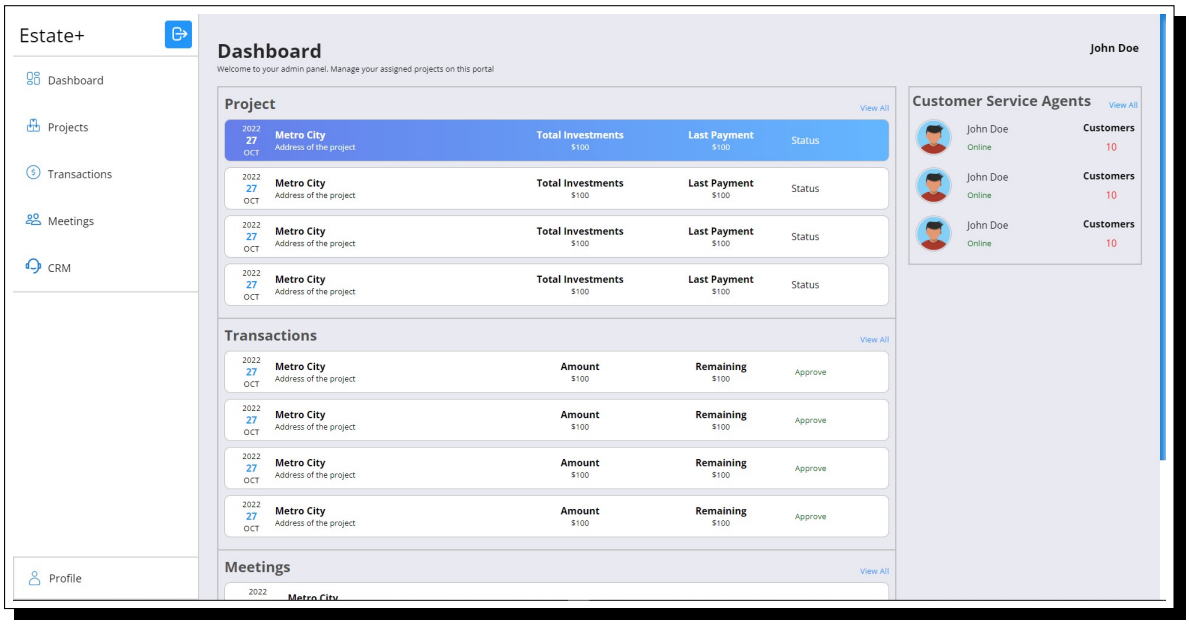

Figure 4.52: Admin Dashboard Hover Effect

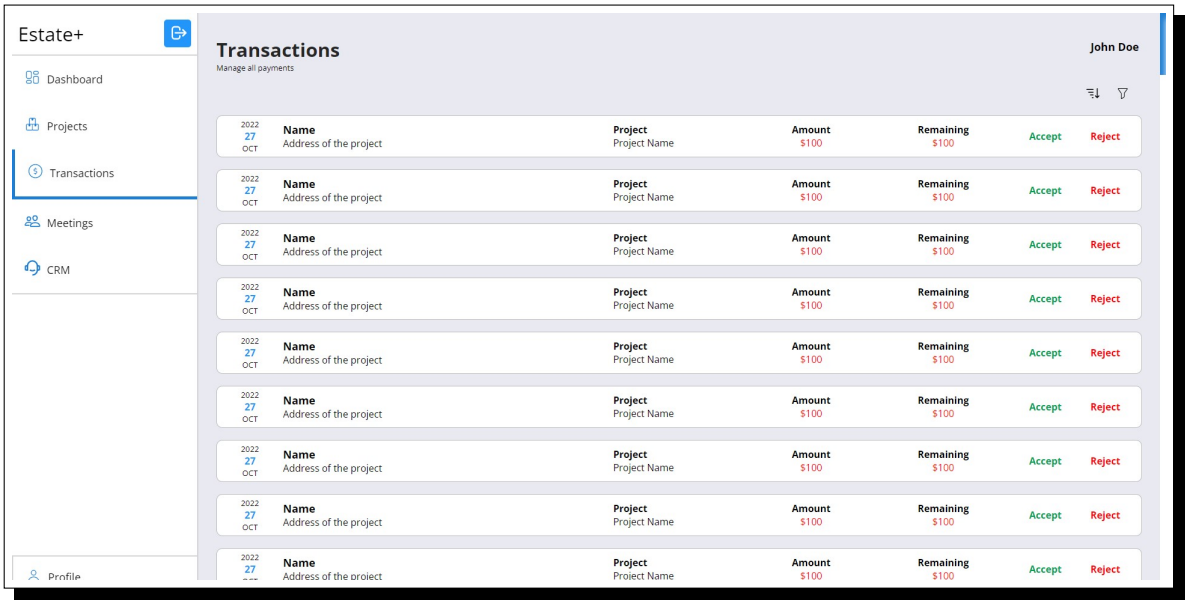

Figure 4.53: Admin Transactions

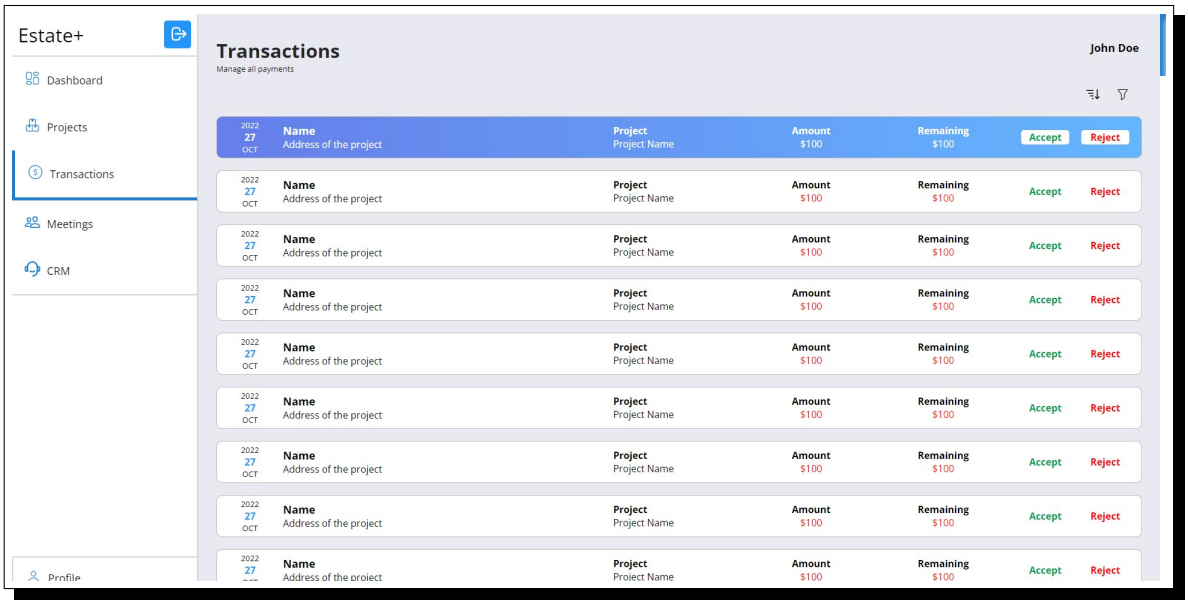

Figure 4.54: Admin Transactions Hover Effect

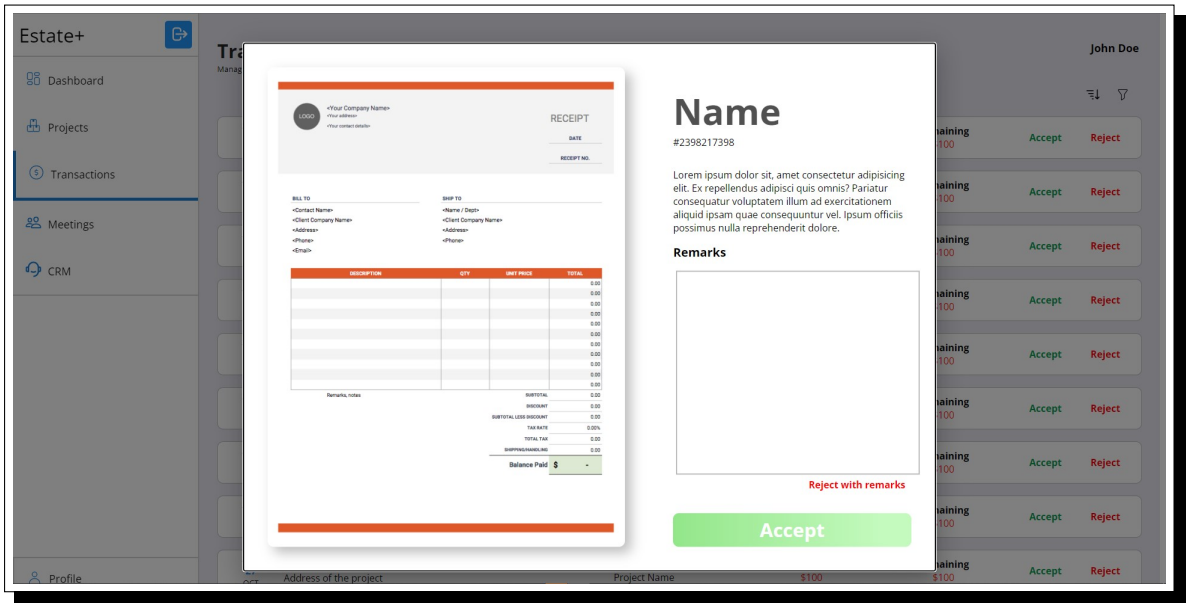

Figure 4.55: Admin Transaction Modal

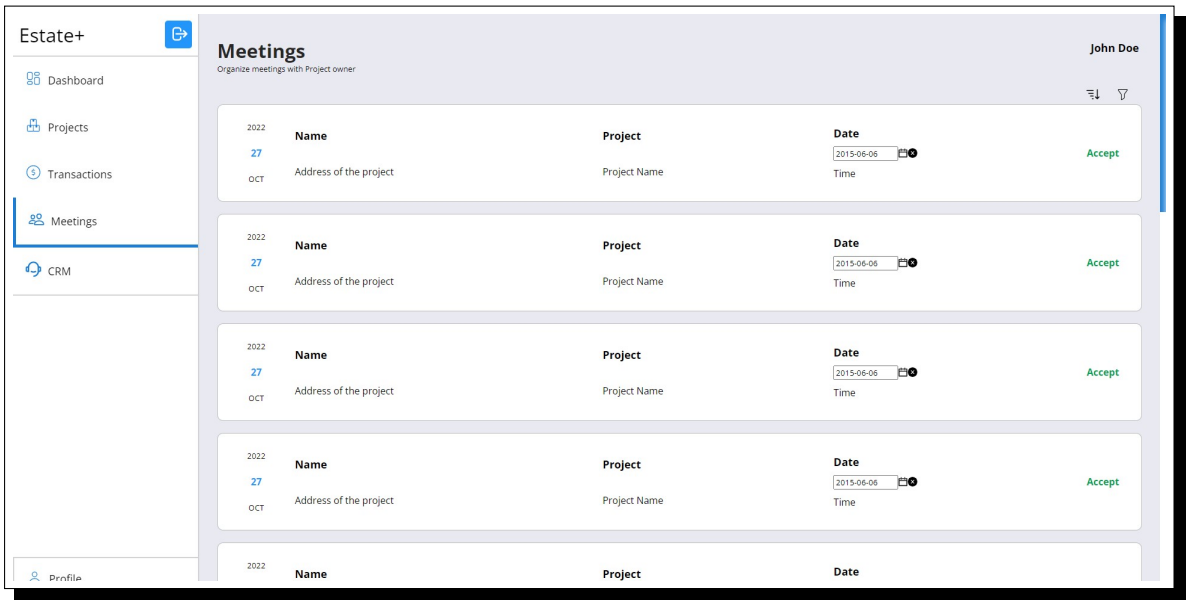

Figure 4.56: Admin Meetings

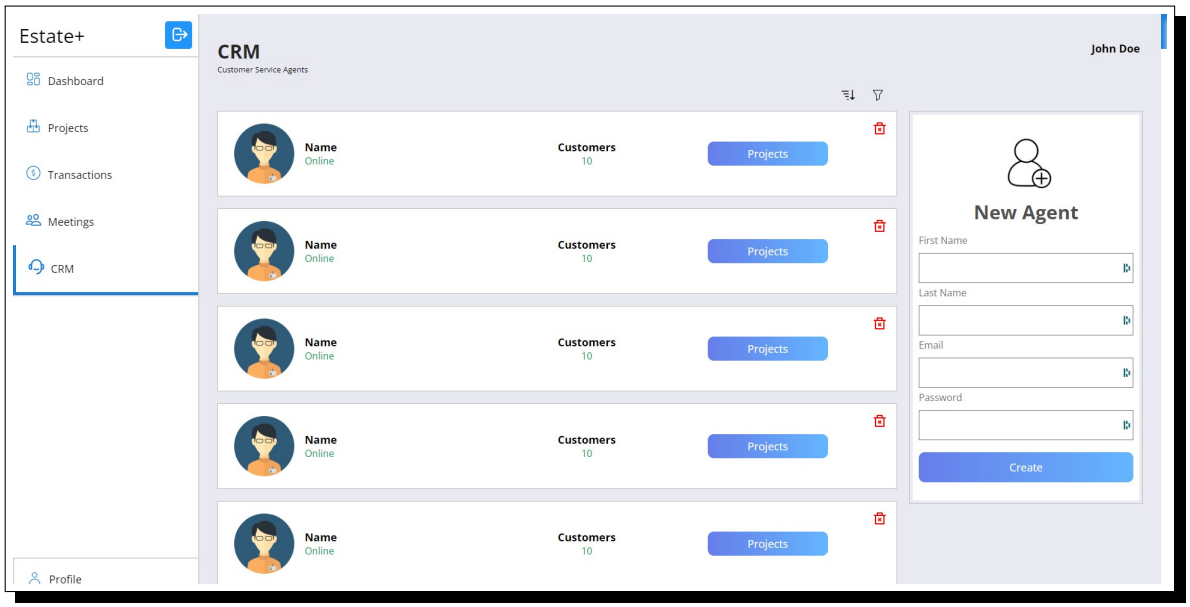

Figure 4.57: Admin CRM Pannel

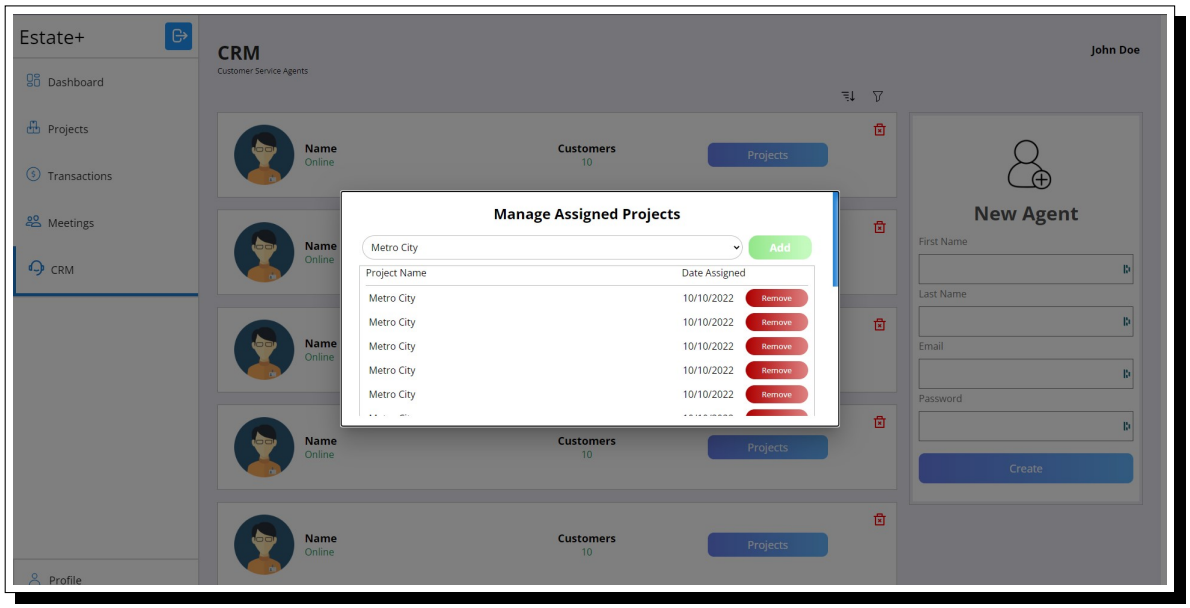

Figure 4.58: Admin CRM Pannel Modal

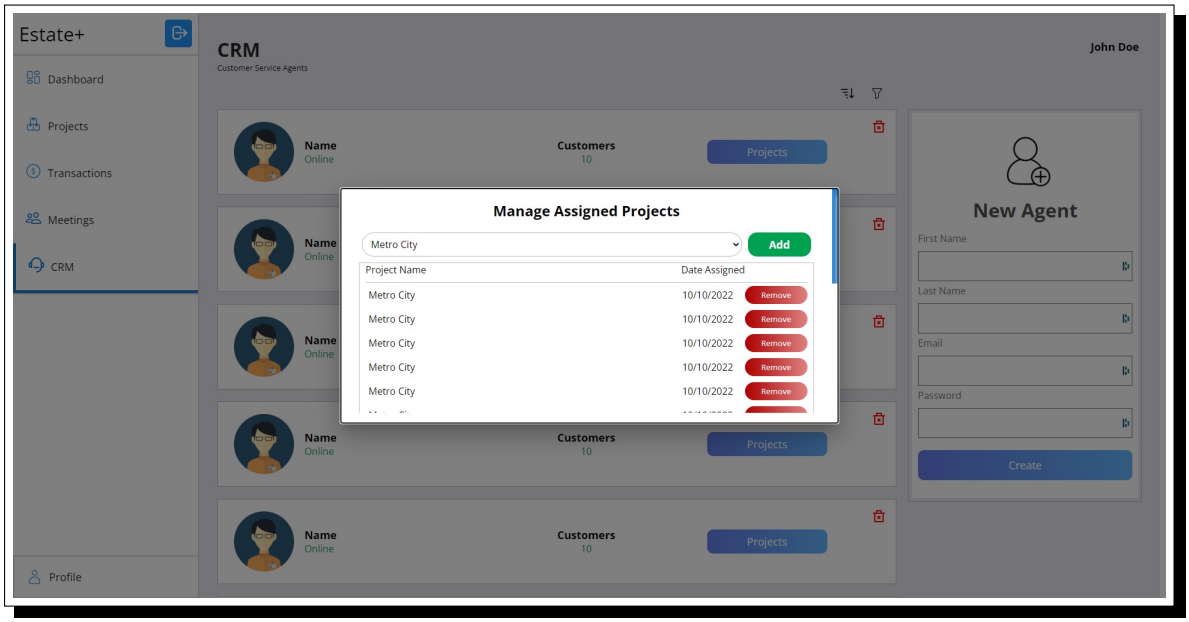

Figure 4.59: Admin CRM Pannel Modal Hover

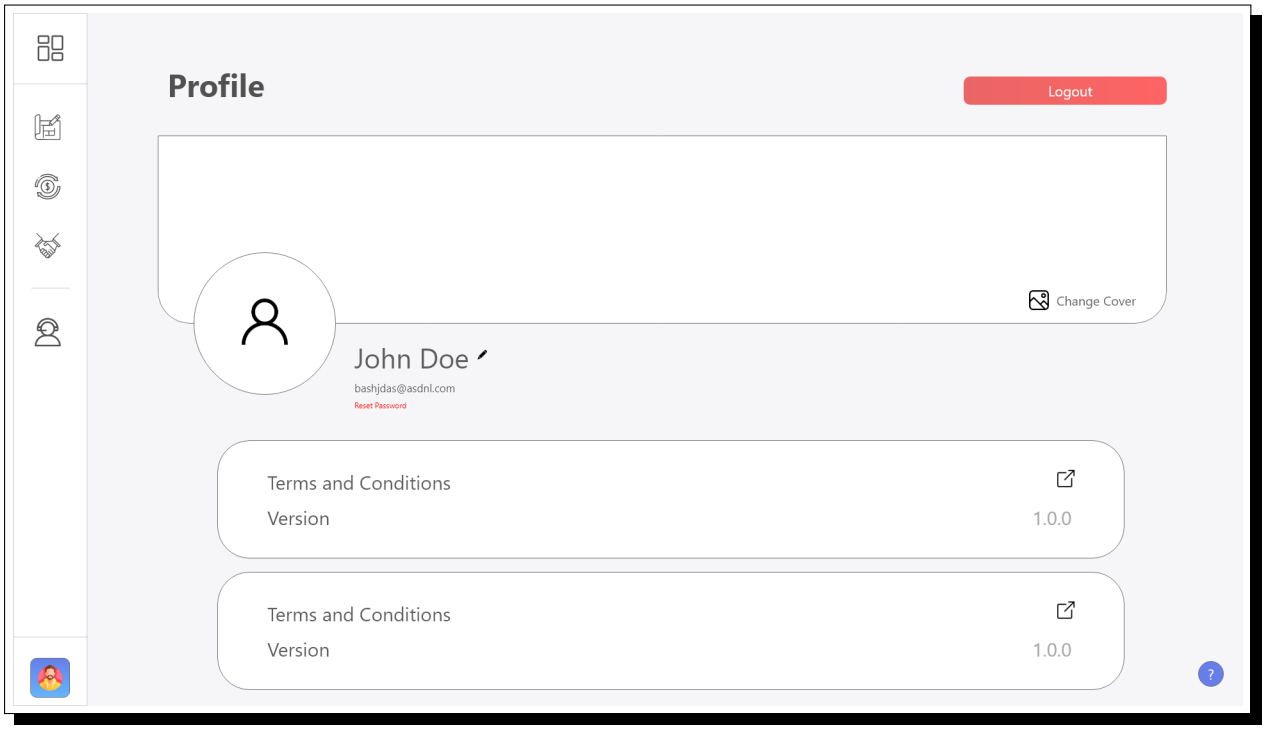

Figure 4.60: Admin Profile

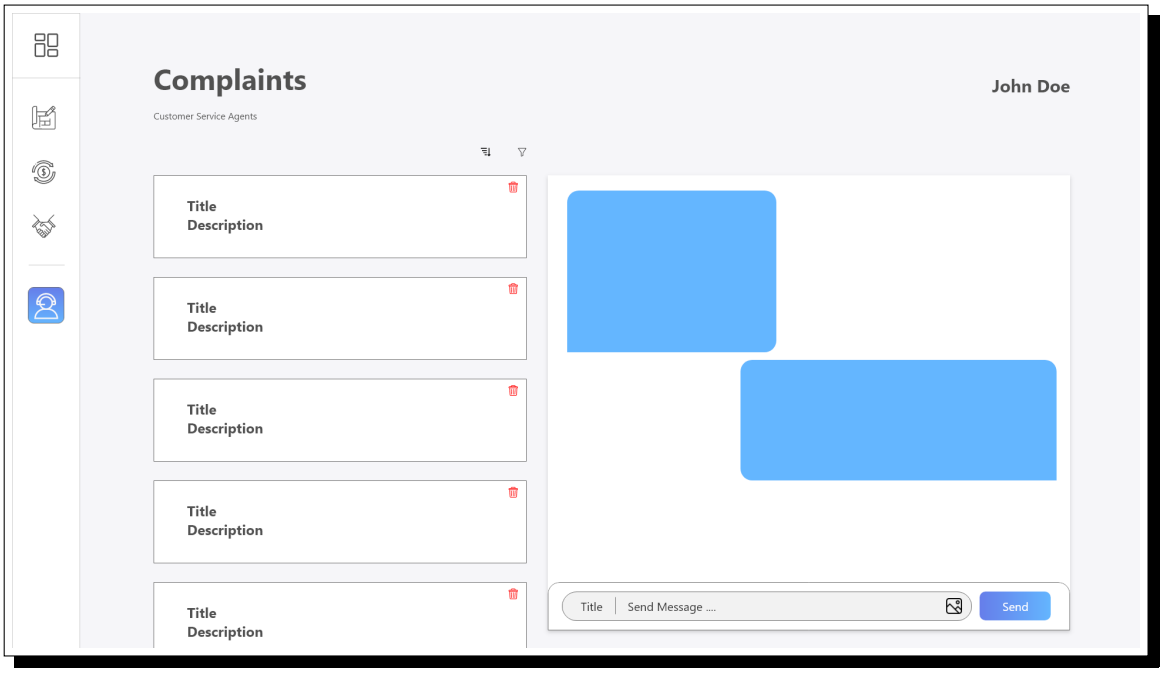

Figure 4.61: Complaints

4.6 GUI Design

To keep the design simple and easy to understand, certain standard needed to be followed. The standard that was followed is Donald Norman's Seven Principles [10]. They are:

4.6.1 Knowledge in Head and World

It was noted that things are consistent in the application as compared to real world applications. Things like icons, process flow needed to be kept consistent. If User cant login, provide them option to Sign Up. If they have an account, provide them option to Login. Icons such as drawer icon needs to kept similar to others.

4.6.2 Visibility

This helps in guiding the user to the action they are about to perform. If there is a text field, it needs to have a label or placeholder to indicate what type of data is needed. Sliders, Buttons need to be labeled accordingly. They were displayed in Search Screen.

4.6.3 Mapping

Since majority of applications have search bar on top of the screen, that is where it was kept in the application. Also submit button come at the end of the application so that's

where they were placed. Back button usually appears on top left of the screen and that is where it was kept. All can be seen in Search Screen.

4.6.4 Constraints

- Physical: Working Keyboard, Mouse and Touchscreen to interact with application
- Conceptual: Valid format of email followed to Sign Up. Invoice Attached to make payment
- Cultural: Same icon used according to functionality. Search icon is a magnifying glass which is usually the case

4.6.5 Simplify

The main procedure are simplified. Each step is highlighted with heading big enough for user to detect. They all follow a pattern, steps that need to be completed before proceeding forward. All have a submit button at the end so the user knows where to go to proceed forward.

4.6.6 Standardize

Since the project that is displayed on the Home Screen is not something used in other application, it is important to make it simple for the user to understand. It needs to be consistent throughout so the user knows where to look the first time they put their eye on the application.

4.6.7 Error

In case of failed procedure, its error needs to be shown to the user. On failed login we can see that in the following figure:

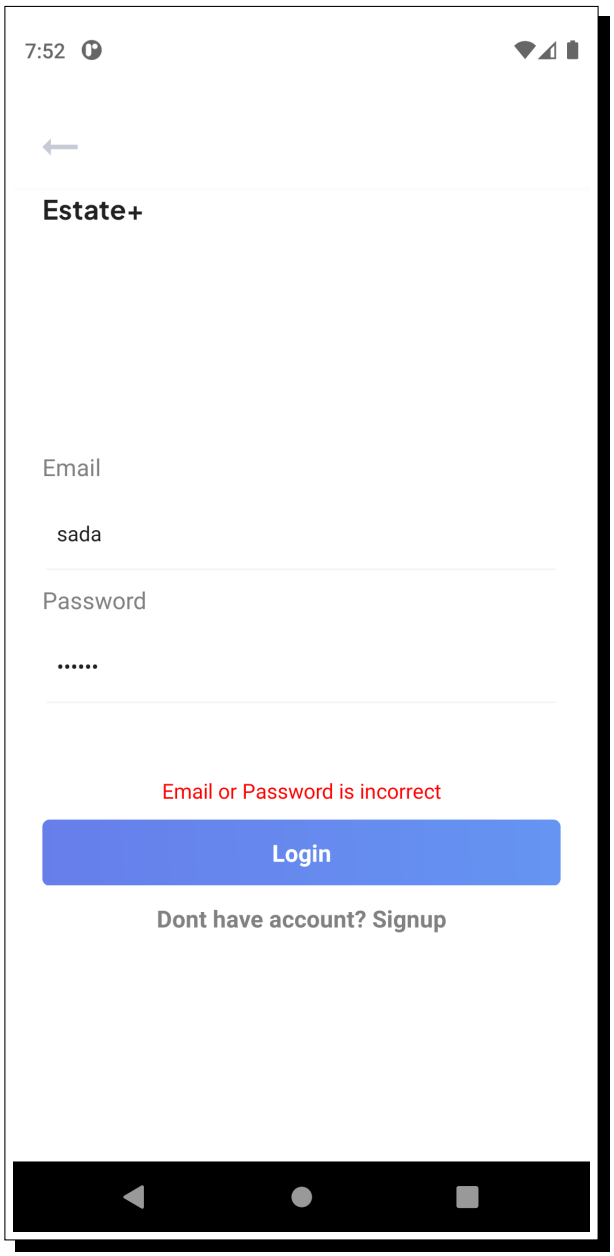

Figure 4.62: Failed Login

4.7 External Interfaces

Since Heroku was used for deployment of server and database, it would be considered as an external Interface. Its domain is used as a base URL for any API calls the application makes. It also provides us with PostgreSQL database's credentials which are used to connect with out server.

4.8 Database Design

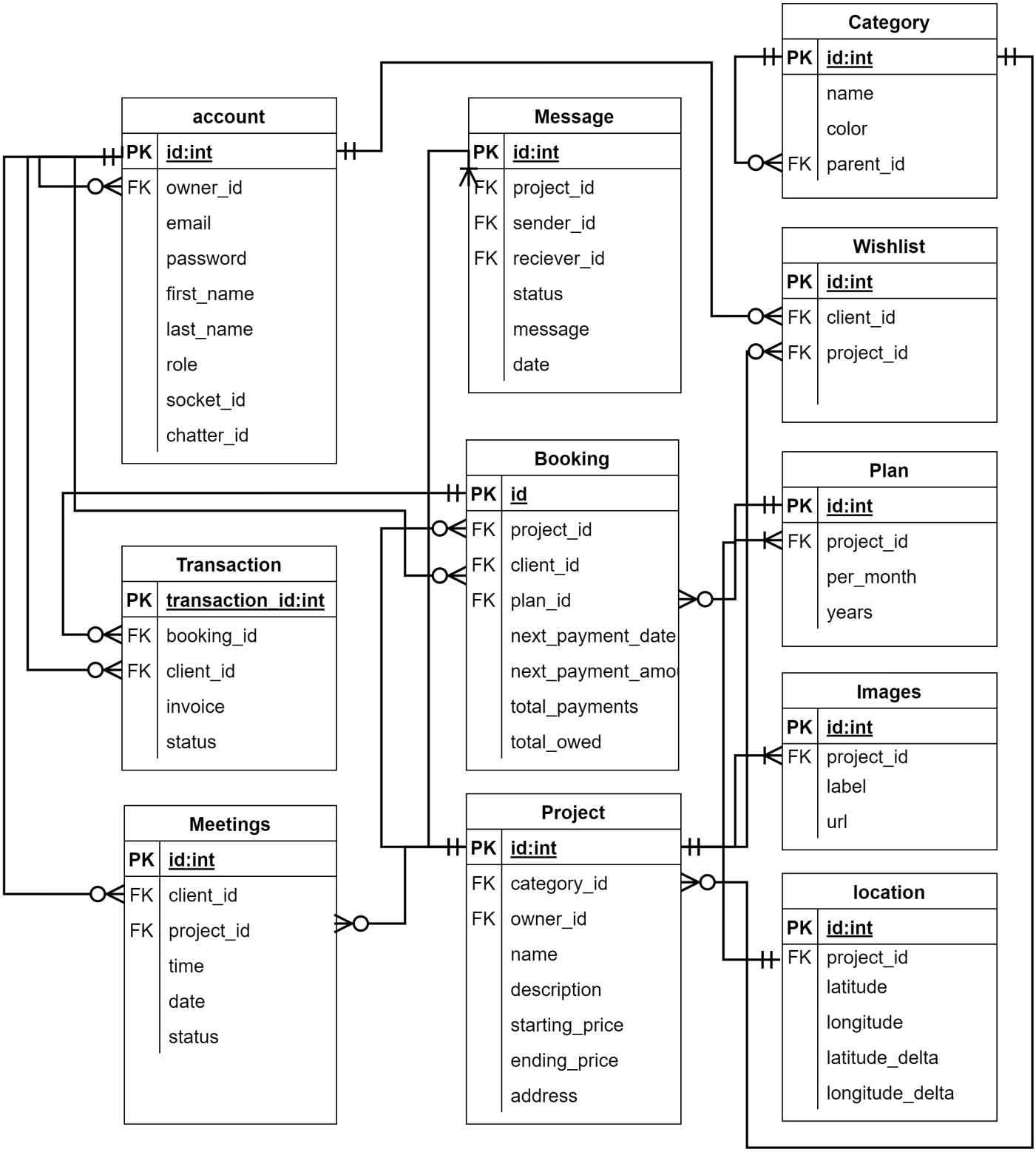

Figure 4.63: Database Design

Chapter 5

System Implementation

5.1 Tools and Technology Used

- React Native for Mobile Application [11]
- React for Web Application [12]
- NodeJS and Express for Server [13]
- Socket.io for Web Sockets [14]
- PostgreSQL for Database [15]

5.2 Development Environment/Languages Used

- Visual Studio Code
- DataGrip
- Postman
- NodeJS
- JavaScript
- SQL
- Adobe XD for designs

5.3 Processing Logic/Algorithms

Since there isn't deep level of programming being performed, no specific algorithms are used. Even the searching is done using queries in database. Sorting is used which was provided by JavaScript by default. Minor filtering is also done through basic JavaScript filter method.

5.4 Application Access Security

- The passwords and other vulnerable data is encrypted using Bcrypt in the server side and stored in database accordingly.
- The API returns a JWT token incase of successful login which is used to authenticate furhter API calls.
- Validation of email and password is done both at front-end and back-end. To avoid huge number of API calls at front-end is the reason validation is done at front-end. To avoid DDoS attacks on the server by repeated API calls with random values is the reason validation is also done at back-end. The valid email format is accepted example@mail.com where "example" is email prefix and "@mail.com" is the domain. The valid password format is that it's length must be bigger than 8 and less than 15. It should also contain special characters in it.
- Since there is a single route for Authentication in API, it returns JWT and role as well for different possible users of the application (Customer, Admin, CRO, Owner).

5.5 Database Security

- Database can be accessed remotely due to its deployment on a Heroku Server.
- The passwords and other vulnerable data is encrypted using Bcrypt in the server side and stored in database accordingly.
- There is only one role in the database which accesses data and it is accessible through the Express Server.

Chapter 6

System Testing and Evaluation

6.1 Graphical user interface testing

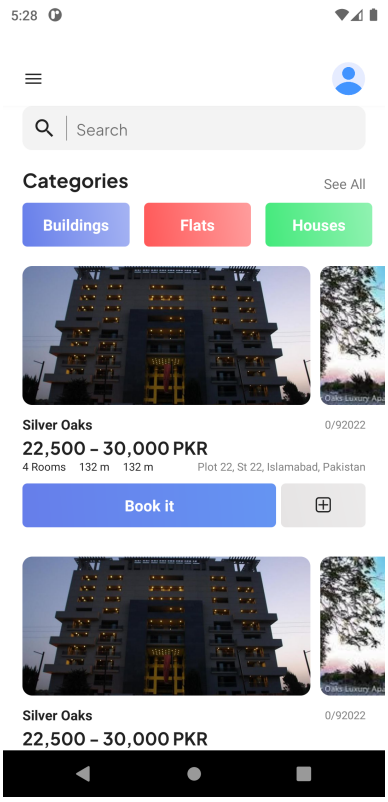

Figure 6.1: Before Scrolling

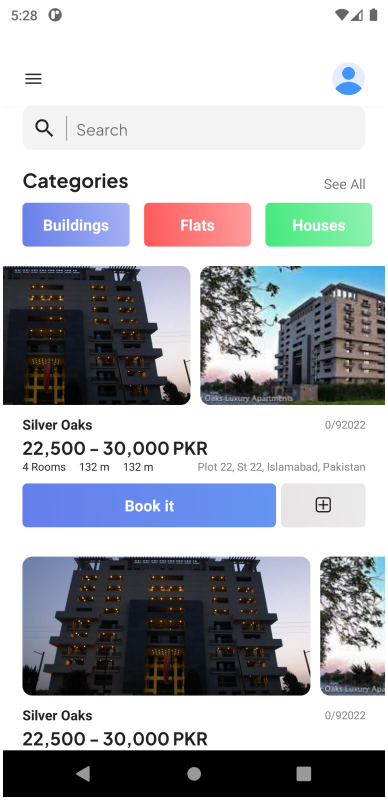

Figure 6.2: After Scrolling

While the image scrolls, it does not cut off at the margins but goes through the entire width of the screen which shows successful implementation of the scrolling

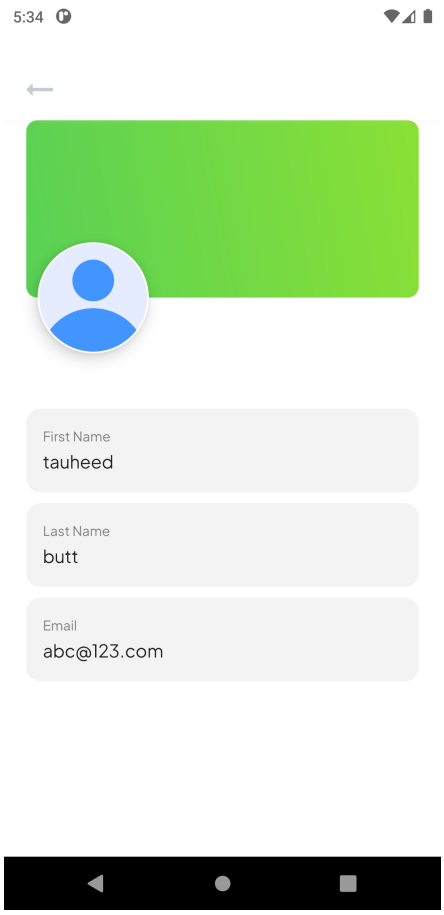

Figure 6.3: Before Changing Data

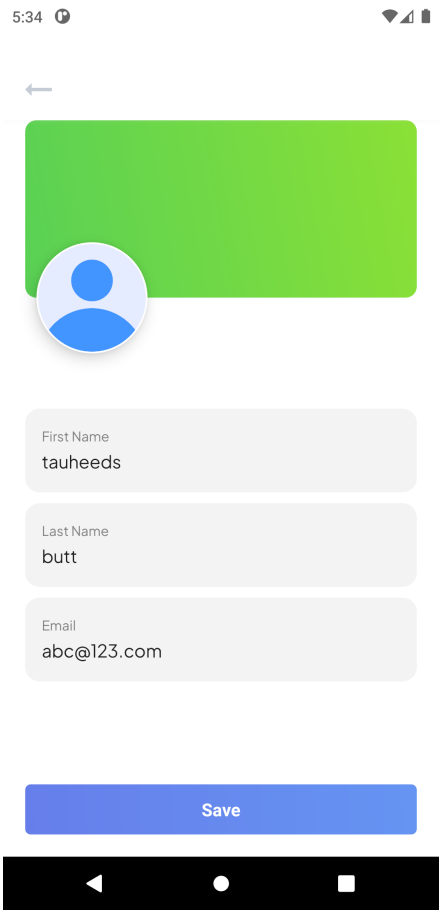

Figure 6.4: After Changing Data

When the User profile is changed, a button shows up to indicate the user to save the changes. if the values are the same, the button goes away.

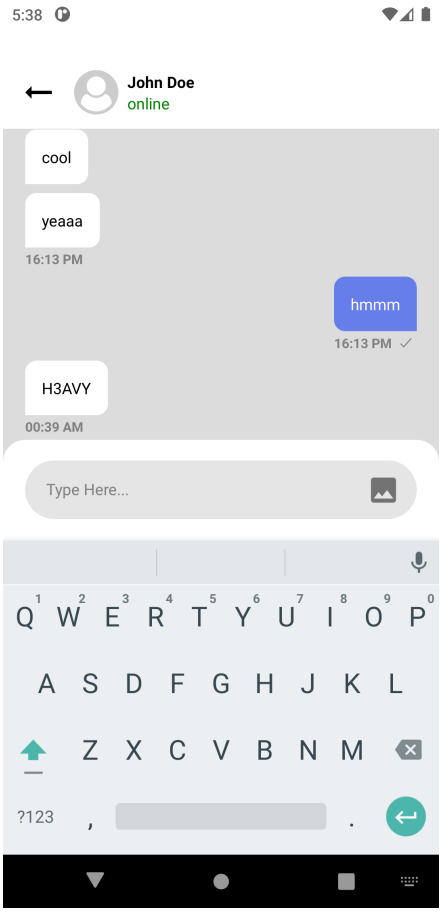

Figure 6.5: Before Typing Message

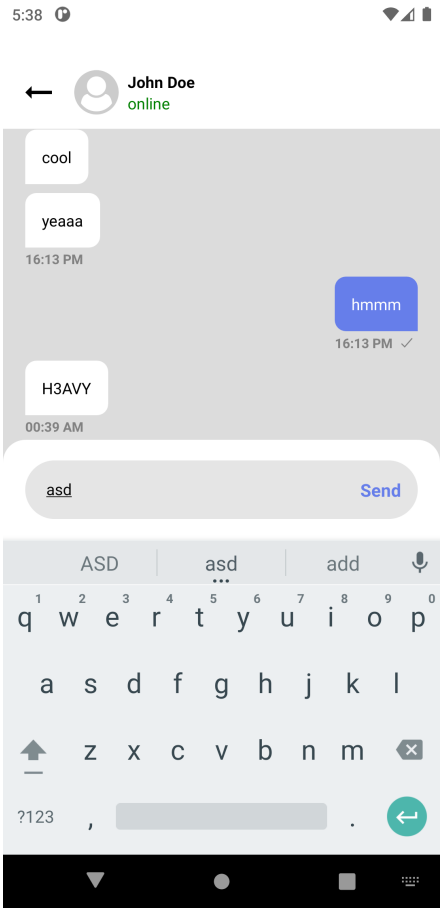

Figure 6.6: After Typing Message

When a message is typed, only then the SEND button shows. If the box is empty, the gallery button is shown.

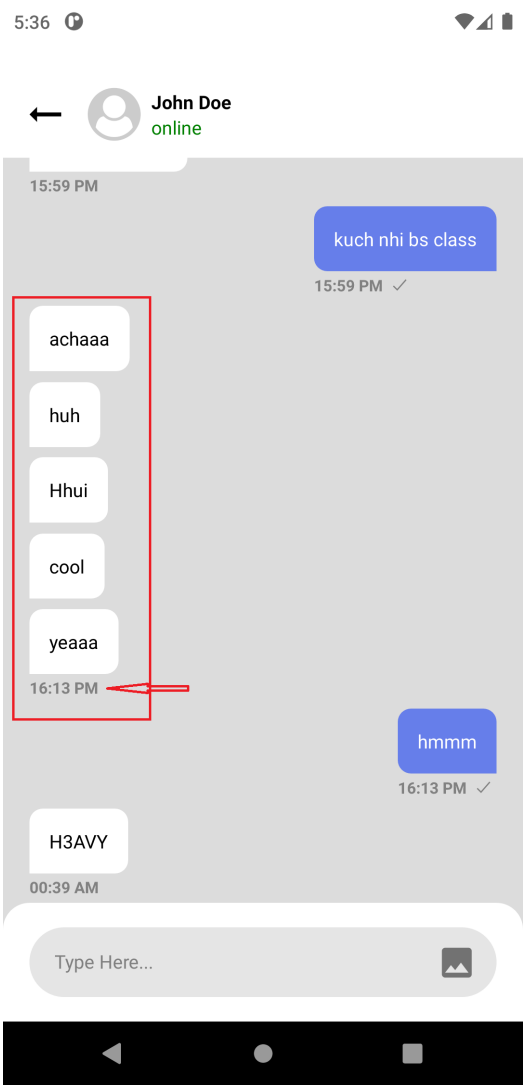

Figure 6.7: Before Typing Message

The time the message was sent is only shown at the last message of the user without interruption.

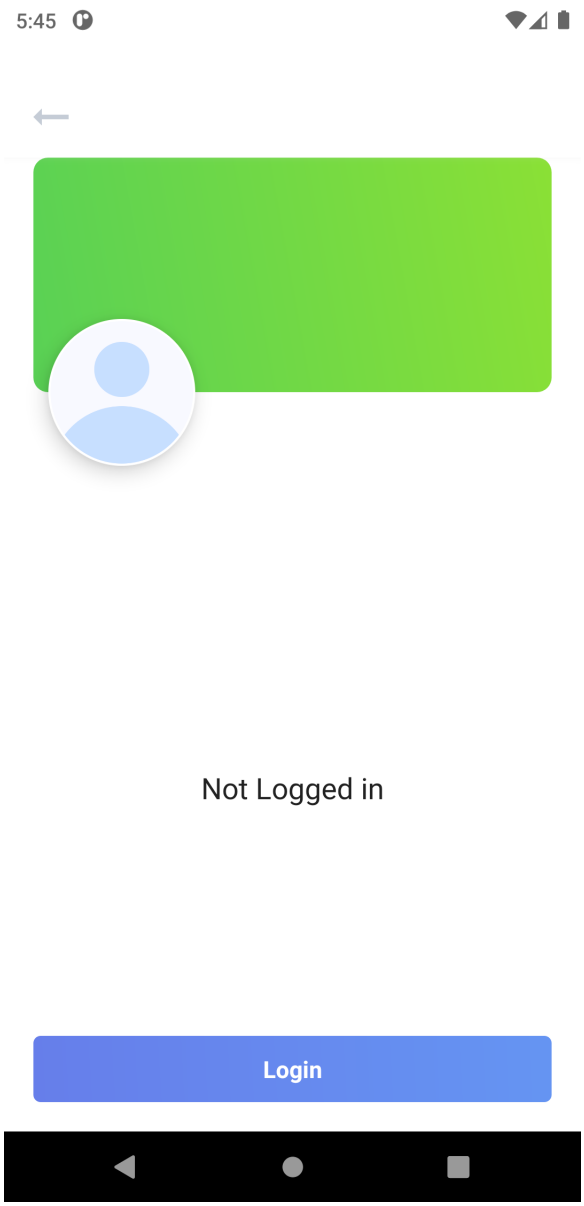

Figure 6.8: Guest interaction

Everytime a button is pressed which requires user login, this screen pops up.

6.2 Usability testing

This sort of testings makes us known whether the application is easy to use and learn by the user or not. It can only be done getting a response from potential users through testing. Since there was no contact with any potential user, Tauheed Butt asked the supervisor at his job to review the application and find any bugs and issues. Initially the font size and heights of buttons were too big. They pointed it out and it was fixed accordingly. However color scheme was highly appreciated as it wasn't too sharp on the eyes.

6.3 Software performance testing

The application relies on API calls to load data in it. If too much of that starts getting stored locally, the application will end up using a lot of user's storage. So there needs to be found a balance in between. Since tokens, roles and wish-list are being stored locally, the login and other related processes get executed quickly. They take up to max 0.1 sec to load. It also depends on what type of device is running the application. If an old device is used, its performance will degrade tremendously.

6.4 Compatibility testing

The application being cross-platform is its strong suit and it needs to be tested accordingly. The web application runs flawlessly on all type of Operating Systems. However, some browsers fail to load some image and font assets. The team failed to fix those issues.

6.5 Exception handling

Since there are 3 views to the mobile application, it is very common to run into exceptions. Some places where user's registered details are being used need to be conditionally changed to the guest's details. Many were handled during development. The API calls are done asynchronously which are bound to produce some exceptions. If not handled, the app can crash. Those ares of code were put in try catch blocks in order to prevent the application from crashing.

6.6 Load testing

The mobile version can experience great amount of load in case of re-rendering due to concurrent changes in state. Some were handled by using useEffect hook of React. Others were being caused due to being able to do API calls by pressing button again and again.

This was handled by disabling the button when the API call was already in process. Once it was completed only then the button was enabled.

6.7 Security testing

The Server however can experience great amount of load in form of DDoS attacks so it needs to be catered accordingly. The access to it is the domain of the server. Where ever it gets used, it was put inside environment variable file which is automatically hidden in any project used. Even if not, it can be manually done so. Other form of load it can receive is by being able to send requests with no authentication and validation. The authentication gets handled through JWT Token which can only be generated if correct login/sign up credentials are provided. The sign up credentials are also validated on front end and back end to avoid repeated API calls.

6.8 Installation testing

Once the server was deployed to Heroku, its domain was tested using Postman. A login request was sent to it to check whether a response is returned or not. It ended up sending the response back. The mobile application was installed on android through the generated apk. It successfully installed however it prompted that the author is not trust worthy. This happens if the application is not registered with Google Play. It can cause doubt in the users who will avoid from using the application.

6.9 Testing Tables

Table 6.1: TC-001

Test Case ID	TC-001
Test Case Name	Change in Profile
Test Case Field	User Profile
Actors	Client, Owner
Description	User will change their profile details
Pre Conditions	User must be logged in
Input Summary	User enters their details in input fields
Expected Result	Save Button Appears
Actual Result	Save Button did Appear
Pass/Fail	Pass

Table 6.2: TC-002

Test Case ID	TC-002
Use Case ID	UC-018
Test Case Name	Send Button
Test Case Field	User CRM
Actors	Client
Description	User will type in chat and send button will apear
Pre Conditions	User must be logged in
Input Summary	User enters text input field
Expected Result	Button changes from gallery to send
Actual Result	Button did Change
Pass/Fail	Pass

Table 6.3: TC-003

Test Case ID	TC-003
Use Case ID	UC-018
Test Case Name	Chat Bubble Time
Test Case Field	User CRM
Actors	Client
Description	Time appears only at the last message
Pre Conditions	User must be logged in
Input Summary	Message is sent or received
Expected Result	Time appears at the last message sent or received
Actual Result	Time appeared at every message
Pass/Fail	Fail

Table 6.4: TC-004

Test Case ID	TC-004
Use Case ID	UC-015, UC-016, UC-017, UC-018
Test Case Name	Guest Actions
Test Case Field	Client View
Actors	Client
Description	Any action that requires authentication and guest user can't perform
Pre Conditions	User must be logged out
Input Summary	An interaction that requires authentication is done.
Expected Result	Profile Screen is shown with message to tell user to log in first
Actual Result	Profile Screen did come up
Pass/Fail	Pass

Table 6.5: TC-005

Test Case ID	TC-005
Use Case ID	TC-001, TC-002
Test Case Name	Email and Password Validation
Test Case Field	Login and Sign up
Actors	Client, Admin, Owner, Agent
Description	Icon color will change based on email and password's value
Pre Conditions	User must be on Sign in or Sign up Screen
Input Summary	Data is entered into email and password field.
Expected Result	On Empty Data: Icon Color is Grey On Incorrect Data: Icon Color is Red On Correct Data: Icon Color is Green
Actual Result	Colors did change
Pass/Fail	Pass

Table 6.6: TC-006

Test Case ID	TC-006
Use Case ID	UC-015
Test Case Name	Project Booking
Test Case Field	Booking Flow
Actors	Client
Description	Project will not be booked again if already booked
Pre Conditions	User must be logged in
Input Summary	Booking order placed.
Expected Result	If already booked, the booking is not made and user is shown appropiate message
Actual Result	It placed another booking
Pass/Fail	Fail

Table 6.7: TC-007

Test Case ID	TC-007
Use Case ID	UC-014, UC-013
Test Case Name	No items fetched
Test Case Field	Loading Items
Actors	Client, Owner, Admin
Description	Application will fetch items based on the screen user is on
Pre Conditions	User must be on a screen that loads some items
Input Summary	Page refreshed or loaded for first time.
Expected Output	If no items loaded, a message is displayed.
Actual Result	No message displayed
Pass/Fail	Fail

Table 6.8: TC-008

Test Case ID	TC-008
Use Case ID	TC-001, TC-002
Test Case Name	Failed Login/Signup
Test Case Field	Authentication
Actors	Client, Owner, Admin, Agent
Description	Error message will be showed on failed authentication
Pre Conditions	User must be signed out
Input Summary	User attempts authentication.
Expected Output	Error message displayed above submit button.
Actual Result	Error displayed
Pass/Fail	Pass

Table 6.9: TC-009

Test Case ID	TC-009
Use Case ID	UC-014, UC-013
Test Case Name	Extra items loading
Test Case Field	Loading items
Actors	Client, Owner, Admin
Description	Error items need to be loaded when end of screen reached
Pre Conditions	User must be on a screen that loads some items.
Input Summary	End of screen reached after scrolling.
Expected Output	Loading icon showed while it attempts to get more items, if no more items, message is shown.
Actual Result	Loader appeared
Pass/Fail	Pass

Chapter 7

Conclusions

The excessive creation of multiple screens for both Web and Mobile led to a small introduction towards industry level projects. Since a screen consisted of more than 1 component, it showed places where it is best to use State-full and State-less components. The data also got shared between multiple screens without navigation which introduced the concept of Contexts in React. Different form of navigation in React and React Native were also discovered such as Bottom Tabs, Drawer, Router etc. Minor implementation of animations in React Native introduced Reanimated. How Data is stored locally on phone using Async-Storage in React Native was also a great learning experience.
Creation of personal API and Server was also learned during the development of this project. Postman was used to test the personal Server which was also a new addition to the skill set. It showed how to connect PostgreSQL with the server through pools provided by the pg module. Different SQL queries introduced to different database concepts such as Joins, Where, Group etc. It also show cased how to connect any database with IntelliJ DataGrip which is a great software for database testing and management.
The whole process opened so many doors towards different tools and technologies that are being practiced and implemented in the real world. It also showed the best ways to implement them and the standards that are being followed to do it. These new learned skills will be a great helping hand while entering the industry.

References

[1] Three Layer Architecture. https://www.ibm.com/cloud/learn/three-tier-architecture, 2022. [Online; accessed 10-October-2022]. Cited on pp. vi and 33.

[2] What is ERP? https://www.oracle.com/pk/erp/what-is-erp/, 2022. [Online; accessed 01-August-2022]. Cited on p. xiii.

[3] What is CRM? https://www.salesforce.com/ap/crm/what-is-crm/, 2022. [Online; accessed 01-August-2022]. Cited on p. xiii.

[4] What is PERN stack ? https://www.geeksforgeeks.org/what-is-pern-stack/, 2022. [Online; accessed 01-August-2022]. Cited on p. xiii.

[5] JWT. https://jwt.io, 2022. [Online; accessed 01-August-2022]. Cited on p. xiii.

[6] Zameen.com. https://www.zameen.com, 2022. [Online; accessed 10-October-2022]. Cited on p. 3.

[7] Zameen.com. https://www.graana.com/, 2022. [Online; accessed 10-October-2022]. Cited on p. 3.

[8] Heroku. https://dashboard.heroku.com, 2022. [Online; accessed 01-October-2022]. Cited on p. 33.

[9] Functional Decomposition. https://engineering.purdue.edu/EPICS/k12/Teachers/Teacher-Toolbox/1.20-Teacher-Toolbox-Functional-Decomposition.pdf, 2022. [Online; accessed 19-October-2022]. Cited on p. 34.

[10] Norman's 7 Principles. https://sites.google.com/a/nu.edu.pk/hci-060129/lectures-1/norman-s-7-principles, 2022. [Online; accessed 20-October-2022]. Cited on p. 85.

[11] React Native. https://reactnative.dev, 2022. [Online; accessed 01-August-2022]. Cited on p. 89.

[12] React JS. https://reactjs.org, 2022. [Online; accessed 01-August-2022]. Cited on p. 89.

[13] Express JS. https://expressjs.com, 2022. [Online; accessed 01-August-2022]. Cited on p. 89.

[14] Socket.IO. https://socket.io, 2022. [Online; accessed 01-August-2022]. Cited on p. 89.

[15] PostgreSQL. https://www.postgresql.org, 2022. [Online; accessed 01-August-2022]. Cited on p. 89.