Team iOS

Abstract

The purpose of the project is to help a businessman manage his online store. The smart inventory system is a mobile application which helps a store owner to manage his online business. The Owner and the Seller can communicate through the application. The application will manage the shipping labels, product details, orders and transaction details.

SMART INVENTORY iOS

Graduated Directed Project

Software Requirements Specification Template

SMART INVENTORY – IOS

Software Requirements Specification

Version - 2.0

April 21,2019

Team Members

Bharadwaj Dasari

Vamshi Raj Jennaikode

Midhun Kumar Kurapati

Shruthi Patlolla

Manogna Sivangula

Naga Ravi Teja Sai Mohan Vummidi

Submitted in partial fulfilment

Of the requirements of

CSIS 44-691 Graduate Directed Project II

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 09/27/2018 | Version 1.1 | Manogna S | First Revision |
| 11/18/2018 | Version 1.2 | Manogna S | Added Technical Menu |
| 12/2/2018 | Version 1.3 | Manogna S | Added User Manual |
| 04/20/2019 | Version 2.0 | Manogna S | Documentation Completed |

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
| Zhengrui Qin | Dr. Zhengrui Qin | Client | 09/29/2018 |
| Vamshi Raj | VamshiRaj Jennaikode | Lead Developer | 09.30/2018 |
| Shruthi Patlolla | Shruthi Patlolla | Tech Lead & Front-end Developer | 11/18/2018 |
| Bharadwaj Dasari | Bharadwaj Dasari | Front end developer | 12/02/2018 |
| Manogna Sivangula | Manogna Sivangula | Back end developer & Technical Writer | 04/21/2019 |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Contents** | **Page no.** |
| **1** | **Introduction**   * 1. Purpose   2. Scope   3. Definitions, Acronyms, and Abbreviations   4. References   5. Overview | **5** |
| **2** | **General Description**   * 1. Product Perspective   2. Product Functions   3. User Characteristics   4. General Constraints   5. Assumptions and Dependencies | **8** |
| **3** | **Specific Requirements**   * 1. External Interface Requirements      1. User Interfaces      2. Hardware Interfaces      3. Software Interfaces      4. Communications Interface   2. Functional Requirements   3. Use Cases   4. Class/Objects   5. Non-Functional Requirements   3.5.1. Performance  3.5.2. Reliability   * + 1. Availability     2. Security     3. Portability   1. Inverse Requirements   2. Design Constraints   3. Logical Database Requirements   4. Other Requirements   5. Prototypes (for complete project)   6. Use Case Diagrams | **10** |
| **4** | **Design**  4.1 ER diagram  4.2. GUI | **21** |
| **5** | **Analysis Models**  5.1. Data Flow Diagram  5.2. Sequence Diagram | **24** |
| **6** | **Technical Manual**  6.1 Login page  6.2 Registration  6.3 Dashboards  6.4 Announcements  6.5 User profile page  6.6 Database Functionality | **26** |
| **7** | **End User Manual**  7.1 Login page  7.2 Registration  7.3 Forgot Password  7.4 Messaging  7.5 Announcements  7.6 Orders View – Admin  7.7 User Profile Page – Admin  7.8 Dashboard  7.9 Orders View – User  7.10 User Profile Page | **29** |
| **8** | **Testing** | **48** |

**UNIT 1: INTRODUCTION**

* 1. **Purpose**

The purpose of the project is to help a businessman manage his online store. The smart inventory system is a mobile application which helps a store owner to manage his online business. The Owner and the Seller can communicate through the application. The application will manage the shipping labels, product details, orders and transaction details.

* 1. **Scope**

The scope of the project is to maintain

* Stock Efficiency
* Ease of communication between Owner and seller
* Visibility of supply chain
* Cost Maintenance
* Quality Management
* Time Management
  1. **Definitions, Acronyms, and Abbreviations**

**Pods:** [Pods](https://cocoapods.org/) manages library dependencies for your XCode projects. The dependencies for your projects are specified in a single text file called “Pod file”. Pods resolves dependencies between libraries, fetches necessary code and links it together in an XCode workspace to build your project.

**App delegate:** XCode provides an app delegate class for every new project, so you do not need to define one yourself initially. When your app launches, UI Kit automatically creates an instance of the app delegate class provided by XCode and uses it to execute the first bits of custom code in your app. The app delegate is effectively the root object of your app. Like the UI Application object itself, the app delegate is a singleton object and is always present at runtime.

**Story board:** A storyboard is a visual representation of the user interface of an iOS application, showing screens of content and the connections between those screens. A storyboard is composed of a sequence of scenes, each of which represents a view controller and its views; scenes are connected by segue objects, which represent a transition between two view controllers.

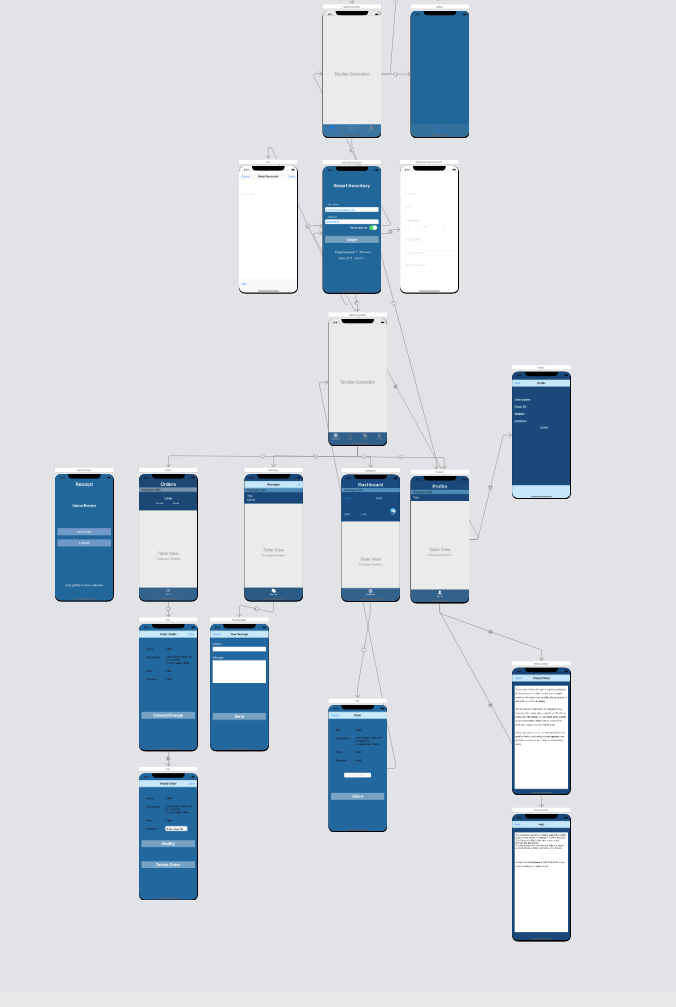
****

Fig.1: Story Board

* 1. **References**

*Apple developer documentation*. (2019). Retrieved from Apple Inc,.: https://developer.apple.com/documentation/

*Backendless SDK for iOS* . (2018). Retrieved from Backendless Corp: https://backendless.com/docs/ios/

*Swift*. (2019). Retrieved from Swift for Apple: <https://developer.apple.com/swift/>

* 1. **Overview**

The inventory system will help a businessman manage his online store. For example, suppose there is an Amazon salesman Bob who buys products on sale in physical stores such as BestBuy and Target and then re-sells them on Amazon. The problem is that the physical stores usually have restrictions on the number of products one can buy at most during sale time. So Bob recruits some people to buy these products and send them to Bob’s Amazon store. Now Bob wants an inventory system to keep the inventory and billing records. Supposes Bob wants to collect some iPhone 6s, he will announce how many he needs and how much he pays for each of them in the inventory system. When Alice buys some products, say 5 iPhone 6s, for Bob, she logins the inventory system and records 5 iPhone 6s. Then she asks Bob for a shipping label and requests a payment.   When Bob sees the request, he will upload a shipping label to the system so Alice can download it. Next, Alice will download the label, ship the package, and upload the shipping receipt into the system. When Bob receives the package, he will examine the products. If everything matches, he will make a payment to Alice.

**Unit 2: GENERAL DESCRIPTION**

* 1. **Product Perspective:**

The product perspective is to make ease of communication for store owner and users and to manage a online store by its owner. This solution is very helpful for owner and user for their activities. They can easily ship and track items, claim items, Instant messaging, Requesting and recording payments, Getting shipping label.

* 1. **Product Functions**

The product functions are divided into two tasks: store owner(admin) and users.

**Store owner or admin tasks:**

1. Login (sign up, sign in, reset passed, etc)
2. Post/edit/delete announcements
3. Send/reply messages
4. Approve/modify/reject user claim or request
5. Check/edit inventory (claimed items, shipped items, items being shipping, etc)
6. Provide and record shipping label
7. Confirm items shipped by user
8. Record payment

**User tasks:**

1. View announcement
2. Make/modify a claim
3. Send/reply messages
4. Check/modify inventory
5. Request shipping label
6. Download shipping label
7. Upload shipping receipt
8. Confirm admin's confirmation
9. Request payment

**2.3User Characteristics**

* Two types of user characteristics admin and user
* Admin has access to post, edit and delete announcement where as user can only view the announcement and request to claim products.
* Admin has to approve the claim request and upload shipping label to ship the claim products.
* User can’t ship claimed products without admins approval.
* Admin and user can directly message using messaging characteristic’s.
* User should request the payment and then admin will record payment for claimed products he receives.
  1. **General Constraints**
* Only admin has access to post or edit or delete announcement.
* User should just can view announcements and see details.
* User has full access to the app when he is logged-in as “Admin”.
* User is limited to access, he will not have all access.
  1. **Assumptions and Dependencies**

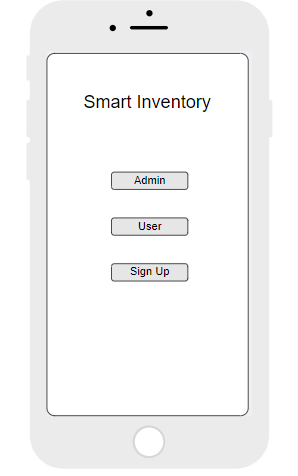
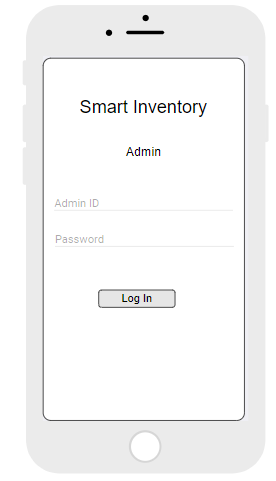
Dependencies required for this application is Swift Language, X-code IDE, Cocoa pods, Back endless Database, GitHub Version control system.

**Unit 3. SPECIFIC REQUIREMENTS**

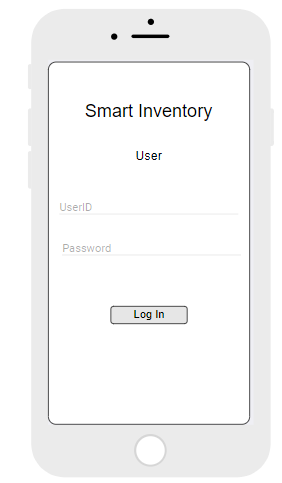
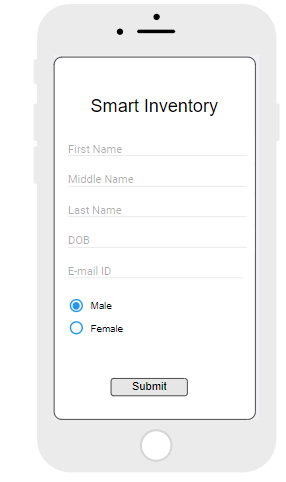
* 1. **External Interface Requirements** 
     1. **User Interfaces**

User interfaces used in this project are mentioned below, welcome screen, sign-up screen, sign-in screen of user, sign-in screen of admin.

1.Welcome screen 2. Admin login screen

** **

**3.**User login screen 4. Sign-up screen

** **

* + 1. **Hardware Interfaces**

Our hardware interfaces are iPhone, iPad, MacBook, Mac Pro. Memory should be 64 GB. Operating system using is MAC. Processor is 2.3GHz dual-core Intel Core i5.

* + 1. **Software Interfaces**

As for the programming environment, OS X is a UNIX system and as such it comes with most of the tools you need to develop. The IDE must be downloaded separately from Apple, it’s called "XCode". The current version of the IDE is 4.x but version 5.x will be released once Mavericks is out.

XCode comes with an iOS emulator, so you do not need to have an actual iPhone during development, it is however recommenced since working in a virtual environment is never the same as working on the actual hardware (Host-Target Development Model). In case you choose to use the iOS emulator, more RAM and possibly an SSD disk will speed things up.

XCode also comes with a dynamic tracing tool called Instruments. Instruments is based on the DTrace tracing framework which was originally developed by Sun Microsystems on the Solaris platform.

You need an iOS Developers account from apple, you can get one at developer.apple.com for about 100$.

**3.1.4 Communications Interface**

**Google Docs, sheets and slides:**

Google Docs, sheets and slides lends itself to collaborative projects in which multiple authors work together in real time from geographically diverse locations. All participants can see who made specific document changes and when those alterations were done. Because documents are stored online and can also be stored on users' computers, there is no risk of total data loss as a result of a localized catastrophe. However, the Internet-based nature of Google Docs, sheets and slides has given rise to concerns among some authors that their work may not be private or secure.

We use this for creating project documents, sheets and slides for presentation. It is reliable for team-work.

**Share Point:**

SharePoint is a web-based collaborative platform that integrates with Microsoft Office. Launched in 2001, SharePoint is primarily sold as a document management and storage system, but the product is highly configurable, and usage varies substantially between organizations.

We use share point to work with team on same platform as Microsoft word.

**Slack:**

**Slack** is a cloud-based set of proprietary team collaboration tools and services, founded by Stewart Butterfield. **Slack** began as an internal tool used by his company, Tiny Speck, in the development of Glitch, a now defunct online game. The name is an acronym for "Searchable Log of All Conversation and Knowledge".

We used to share or receive any type size of files in slack.

**Outlook:**

It is Microsoft e-mail service. We use outlook to send and receive mails. Used to contact team members.

**3.2 Functional Requirements**

**3.2.1 Admin Modules**

1. Login

Admin can login into the system with some specific privileges using his credentials.

Password should be at least 8 characters, it should be a combination of letters and numbers and should contain at least on special character.

1. Add or Remove Users:

Admin can add or remove the users from system.

1. Add or Delete Items

Admin should have an option to add or delete items in the inventory.

1. Post updates on the dashboard

Admin can post the updates on dashboard, so that all the users can view the data.

1. One to One communication

Admin can contact the user on personal chat.

1. Send Shipping labels to user

Admin should be able to send the hipping labels to the users to send the products.

1. Confirmation up on receiving the product

Admin should send a confirmation note to user up on receiving the item.

1. Send amount to user.

Admin can able to send the money to the user through PayPal.

1. View transaction history

Can view the transaction history.

* + 1. **User Modules**

1. Sign up

New user should be able to sign up for inventory system.

1. Login

Users can able to login to the system with their credentials.

Username should be unique.

Password should be at least 8 characters, it should be a combination of letters and numbers and should contain at least on special character.

1. View Admin posts

User can view the admin posts on the dashboard.

1. Contact Admin

User can contact admin about the items he has requested and details.

1. Send request for selling items

User can send a request to admin about the availability of items.

1. Send request for shipping labels

User can request for shipping labels.

1. Send an invoice for payments

User should be able to send invoice about the products they have shipped.

1. View Transaction records

User should be able to check the transaction history.

1. Vendor rating

User can provide rating to vendor about the process.

**3.3 Use Cases**

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. It consists of a group of elements (for example, classes and interfaces) that can be used together in a way that will have an effect larger than the sum of the separate elements combined. The use case should contain all system activities that have significance to the users. A use case can be thought of as a collection of possible scenarios related to a particular goal, indeed, the use case and goal are sometimes considered to be synonymous.

A use case (or set of use cases) has these characteristics:

* Organizes functional requirements
* Models the goals of system/actor (user) interactions
* Records paths (called *scenarios*) from trigger events to goals
* Describes one main flow of events (also called a basic course of action), and possibly other ones, called *exceptional* flows of events (also called alternate courses of action)
* Is multi-level, so that one use case can use the functionality of another one.

In our project, our use cases are products, invoice, transaction, orders etc., You can see all use cases in Use case diagram section (3.11)

**3.4 Class/Objects**

* User:

To store the user details like name, email, password, type of user.

* Product:

To store the product details like product name, number, quantity.

* Order:

To store the orders details like order number, items in the order and quantity and also the shipments number.

* Shipment:

Shipment is used to store shipping details of each order , it includes shipment id, order id, date and address.

* Transaction:

Transaction describes about all the transaction details with the id, amount, sender and receiver.

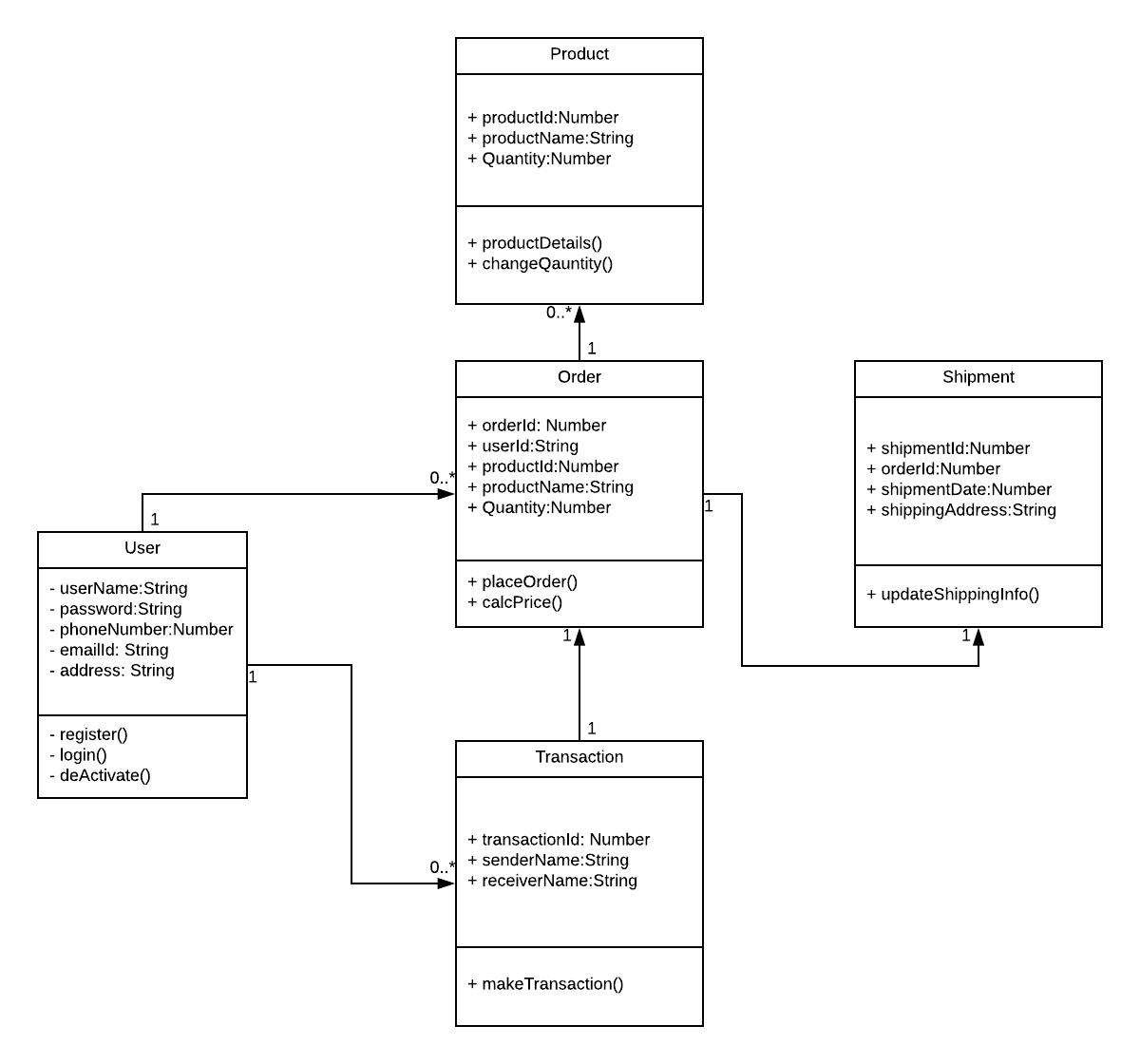


Fig.2: Story Board

**3.5 Non-Functional Requirements**

**3.5.1. Performance**

This is to check the speed of the app that means how fast the app is responding to the actions. Generally, every action should not take more than 20 seconds to respond. In this application it the user should not feel any delay in response when he performed the series of action through the app.

**3.5.2. Reliability** Reliability is the percentage of time that the app works correctly without any failures and to deliver the results properly. For our app it is the amount of time the app can run process of selecting products and make transaction of it. It should be 99 percentage in general.

**3.5.3. Availability**

Availability means the provider systems should meet the agreed availability targets like service downtime and available hours as defined in the operational level agreement. For this app we will specify the downtime periods for month and the services will be not available at that time and rest of the time the services are available.

**3.5.4. Security**

The provider should resist unauthorized access through the application and should provide access to authorized users only. So, in this app we will authenticate the users based on security mechanisms and provide security for the user’s personal information.

* + 1. **Portability**

Portability checks for whether the app works for different platforms or different OS environments. As this project is to support only the IOS platform, we will work on to support all the devices which runs on IOS operating system.

* 1. **Inverse Requirements**

Inverse requirements state the requirements what the system will not do. It describes the constraints on allowable behavior. In most of the cases, it is easier to state that certain behavior must never occur than to state requirements guaranteeing acceptable behavior in all circumstances. It depends mostly on the client.

* 1. **Design Constraints**

Most mobile products are failed because of not limiting the application design. Here are some design constraints that we are planning to use.

**1.Client-side storage**: From the statistics the average app loses more than 50% of the users within 3 days because of the storage problem that users are facing.

2.**Update in the Application**: If the update has any problems then there’s a chance of reduction in the users. So, the update must be fully tested and should be bug free.

* 1. **Logical Database Requirements**

Specify the logical requirements for any information that is to be placed into a database, like

* Types of information used by various functions
* Frequency of use
* Integrity constraints
* Data entities and their relationships
  1. **Other Requirements**

A Mac Book with minimum requirements like 8 GB RAM and 500 GB hard disk and should be able to run X-Code. An iPhone with minimum model of iPhone 6 is required for testing the application. IOS developer account is required to access more granular control over the operating system entities. A NoSQL database for storing data related to the project.

* 1. **Prototypes (for complete project)**

Prototype is an early sample, model, or arrival of an item worked to test an idea or process or to go about as a thing to be repeated or learned from.

**Categories of Prototypes:**

**Proof-of-Principle Prototype** serves to verify some key functional aspects of the intended design, but usually does not have all the functionality of the final product.

**A Working Prototype** represents all or nearly all of the functionality of the final product.

**A Visual Prototype** represents the size and appearance, but not the functionality, of the intended design. A Form Study Prototype is a preliminary type of visual prototype in which the geometric features of a design are emphasized, with less concern for color, texture, or other aspects of the final appearance.

**A User Experience Prototype** represents enough of the appearance and function of the product that it can be used for user research.

**A Functional Prototype** captures both function and appearance of the intended design, though it may be created with different techniques and even different scale from final design.

**A Paper Prototype** is a printed or hand-drawn representation of the user interface of a software product. Such prototypes are commonly used for early testing of a software design, and can be part of a software walkthrough to confirm design decisions before more costly levels of design effort are expended

**3.11. Use Case Diagrams**

A UML use case diagram is the primary form of system/software requirements for a new software program under developed. Use cases specify the expected behavior(what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (such as UML). A key concept of use case modeling is that it helps us design a system from end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

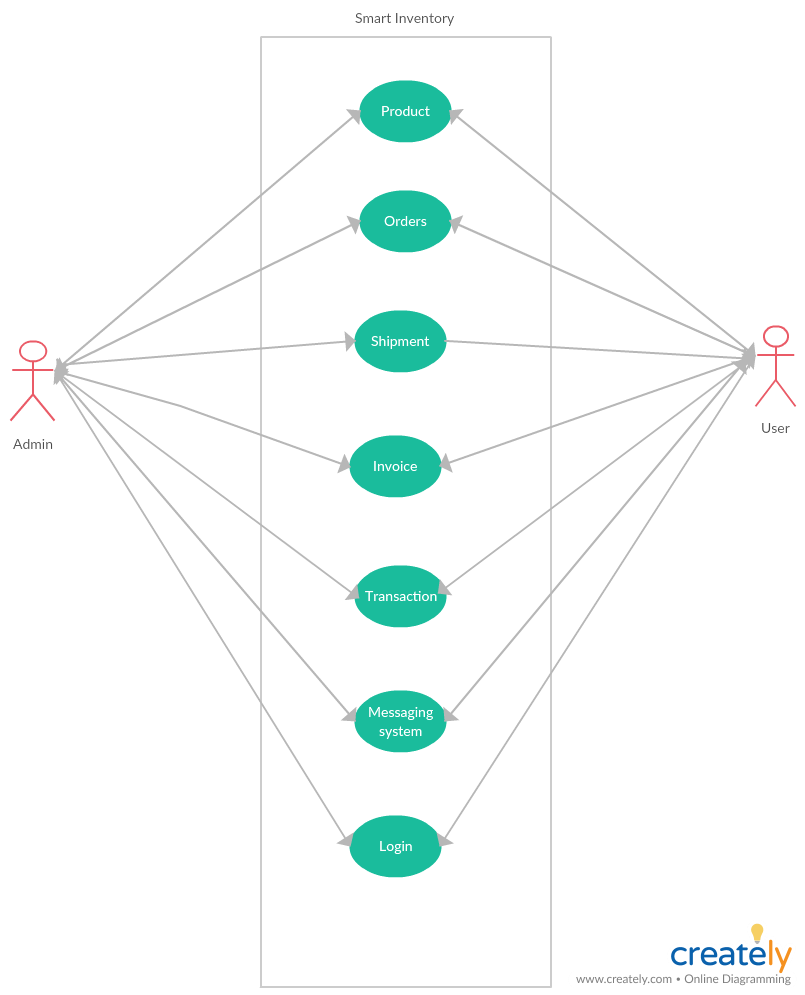


Fig 3. Use-Case Diagram

**UNIT 4. DESIGN**

**4.1. ER diagram:**

An entity-relationship model (ERM) is a theoretical and conceptual way of showing data relationships in software development. ERM is a database modeling technique that generates an abstract diagram or visual representation of a system’s data that can be helpful in designing a relational database. These diagrams are known as entity-relationship diagrams, ER diagrams or ERDs.

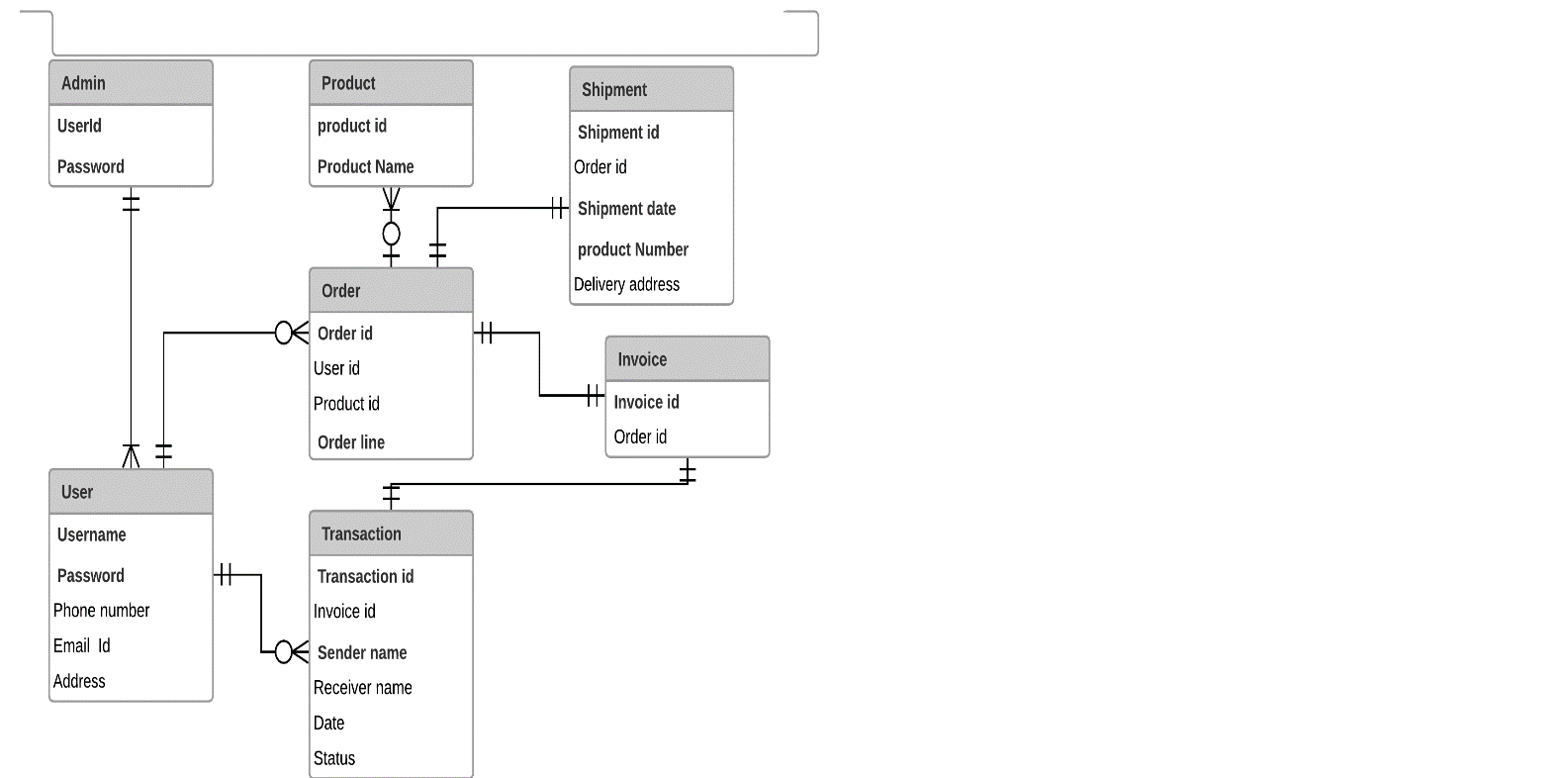


Fig 4. ER Diagrams

**4.2. GUI:**

Mockups reflect the design choices for color schemes, layouts, typography, iconography, the visuals of navigation, and the overall atmosphere of the product.

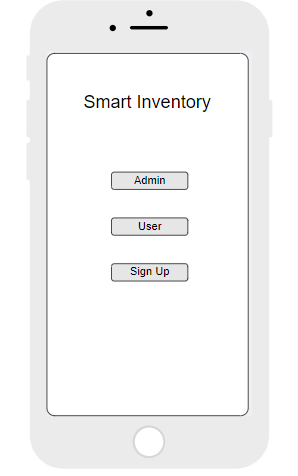
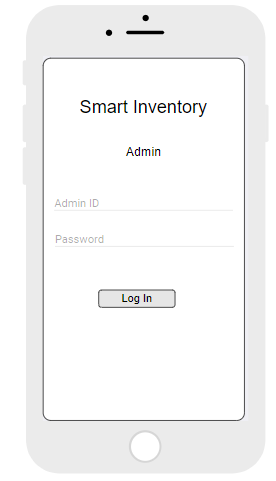
** **

Fig 5. Welcome Screen Fig 6. Admin login screen

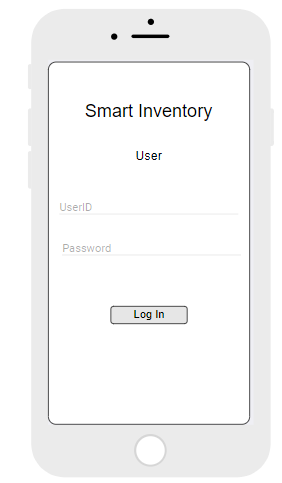
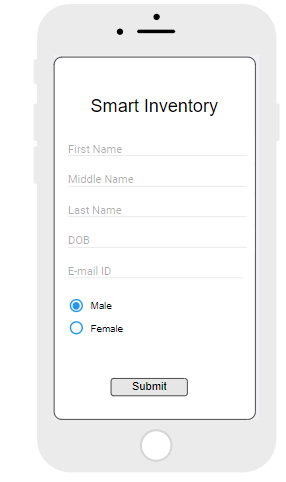
** **

Fig 7. User Login Fig 8. Signup screen

**Unit. 5: ANALYSIS MODELS**

**5.1. Data Flow Diagram**

A **data-flow diagram** (DFD) is a way of representing a flow of a data of a [process](https://en.wikipedia.org/wiki/Process) or a system The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a [flowchart](https://en.wikipedia.org/wiki/Flowchart).

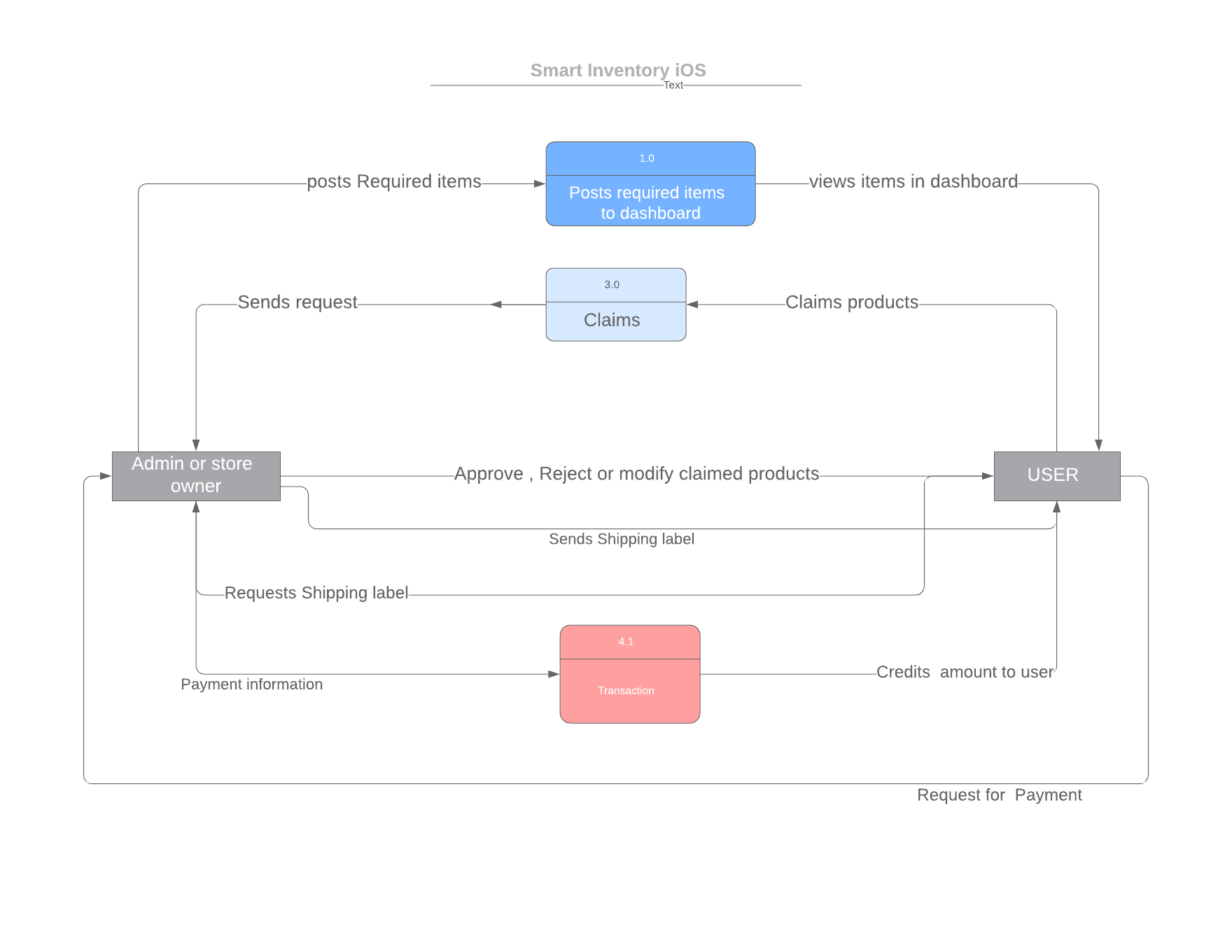
****

Fig. 9: Dataflow diagram

**5.2. Sequence Diagram**

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams** or **event scenarios**.

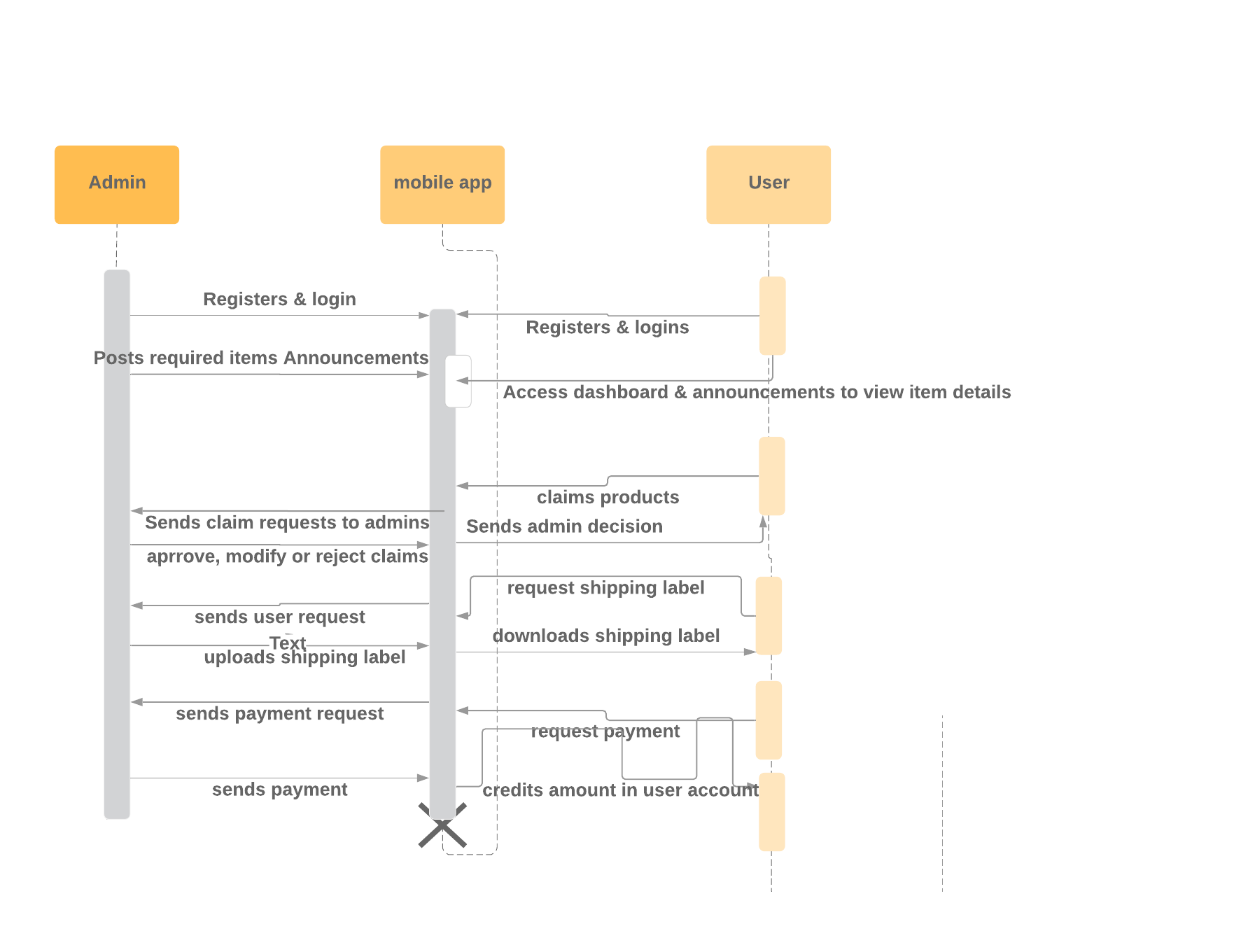
****

Fig. 10: Sequence Diagram

**UNIT 6. Technical Manual**

**6.1 Login page**

This page has username and password text fields. When user enters username and password verifies user credentials.

* The Login authorization of user is done using Back-endless userService. We have UI validations for username and password and will displays proper feedback to user.
* Forgot password functionality is implemented using Back-endless in-built e-mail services. A temporary password is sent to the user through E-mail.

**6.2 Registration**

User has to register with required details. When user clicks on Registration Button, if all the details given are correct then feedback message user is sent “Registration Successful”. If the details are wrong, an error message is sent to user.

All the Valid details are stored to Back-endless database when the user is successfully registered.

**User Functionality**

**6.3 Dashboards**

All the items and quantity required for the admin is published in the Dashboards. It is in progress, team is planning to work on it.

**6.3.1 Claim products**

**6.3.2 Update claim**

**6.4 Orders**

**6.4.1 Request shipping label**

**6.4.2 Download Shipping label**

**6.4.3 Upload shipping receipt**

**6.4.4 Request Payment**

**Admin Functionality**

**6.5 Announcements**

**6.5.1 Post Announcement:**

**6.5.2 Update Announcement:**

**6.5.3 Delete Announcement:**

**6.6 Orders**

**6.6.1 Upload shipping label**

**6.6.2 Confirm Order**

**6.6.3 Order Received**

**6.6.4 Record Payment**

**6.7 User or Admin Profile Page**

In user profile page displays personal details of user.

When user login’s, we will store all the details of user details using user struct (Like web sessions in Web). Using current user id, set all the details of the user.

**6.8 Database Functionality**

Database used for this application is “Back-endless”.

**Connection to Back-endless:**

* Register & create an app in Back-endless. After created every app has API-Key in Back-endless (can see API-Key in “Settings”).
* Connect app to back-endless using API-Key (embed this API-key in APP DELIGATE file).
* Create a back-endless instance, using this back-endless perform CRUD operations.

Using back-endless user service, we have implemented login, registration and forgot password functionalities.

**6.9 Messaging:**

**UNIT - 7 END USER MANUAL**

**7.1 Login page:**

Every user needs to login to access all the functionalities. If he/she is a new user, they need to sign-up for new account. Every user is recommended to have an account. If user forgets his password, can reset his password using “Forgot password” functionality.

Login screen consists of two fields, one is username and other is password. user needs to give his correct username and password in respective fields. If the given credentials are valid, user can log-in. If credentials are invalid, they are acknowledged with a feedback message “Invalid username and password”.

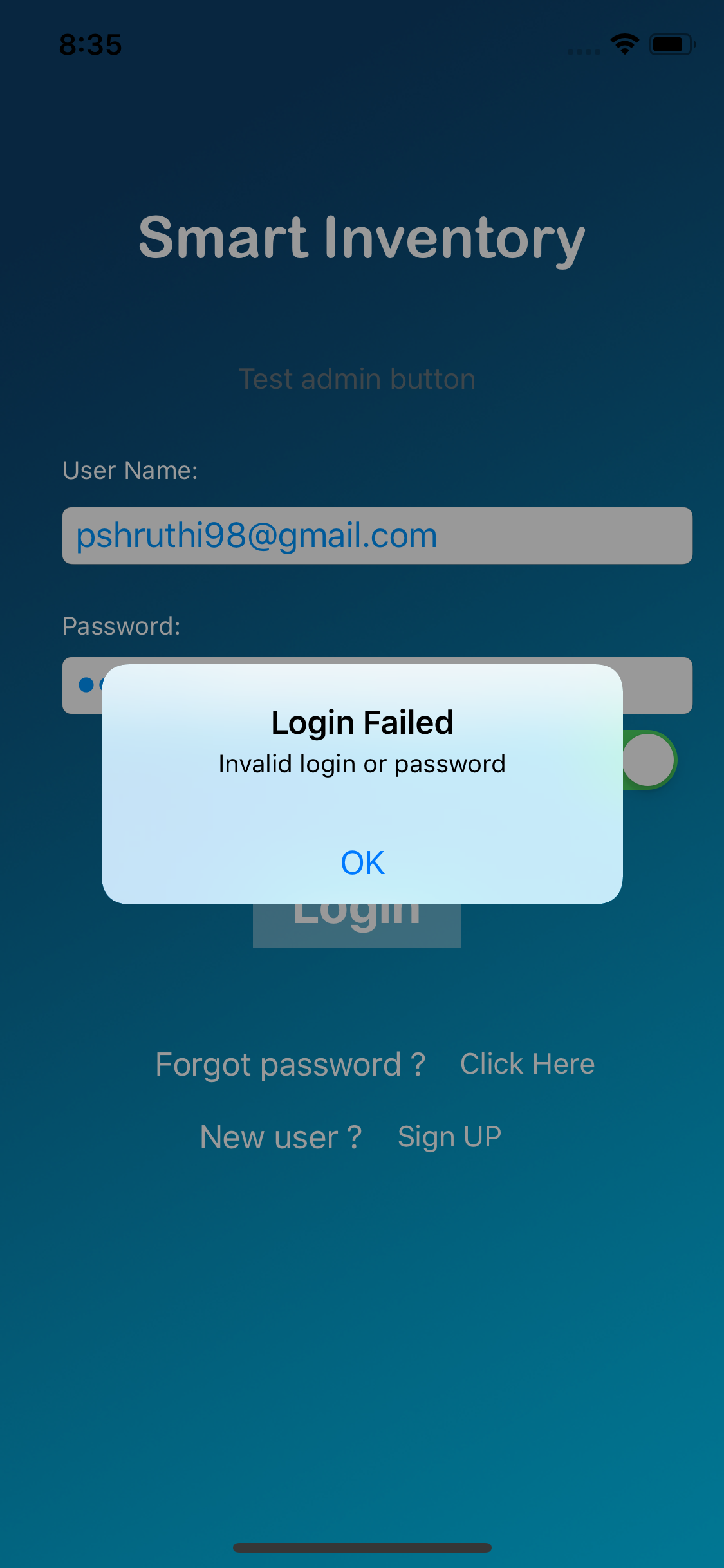
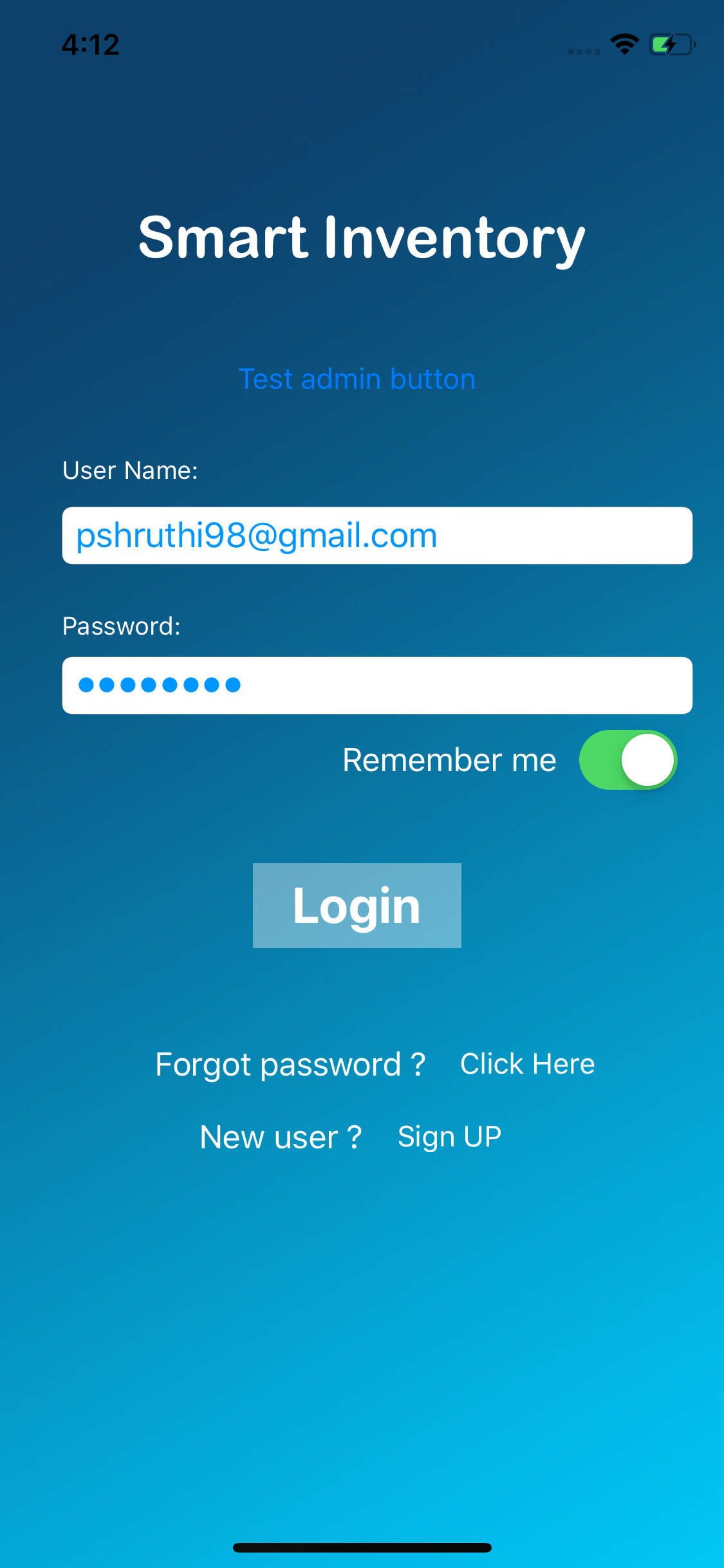


Fig.11: Login page Fig.12: Login authentication failed

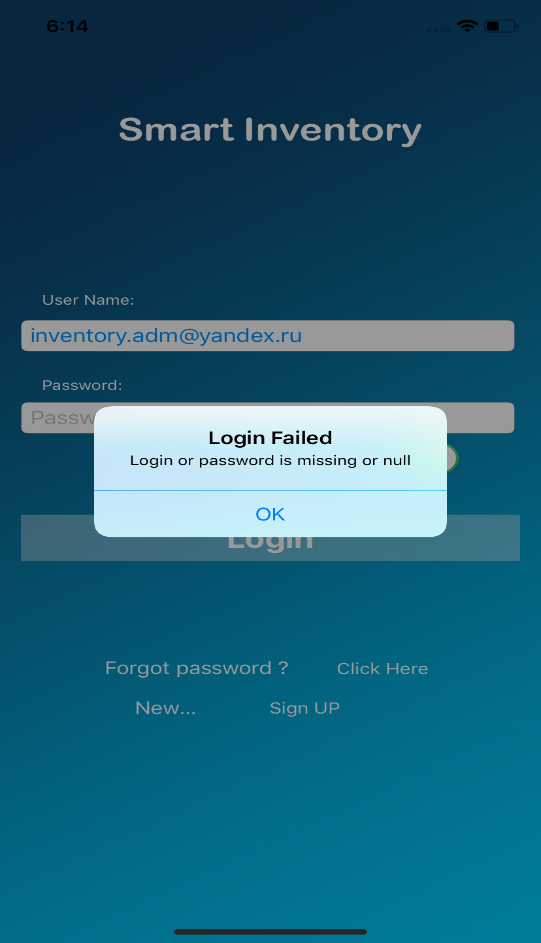


Fig.13: Login Failed Alert

**7.2 Registration:**

When user clicks on “Sign-up” in log-in screen, user is redirected to registration page. In registration page user needs to fill all required fields for successful registration.

User needs to fill Name, e-mail address, phone number, password to get registered. User E-mail address is used as username for the account. Password should have at least one capital letter, one special character, one number, and minimum eight characters in total.

If the required fields are filled and valid, user is registered successfully. Otherwise, user receives a feedback message that he/she is not registered successfully.

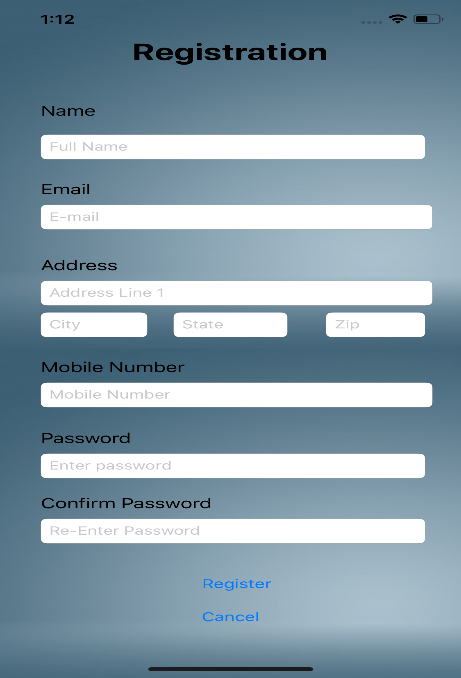
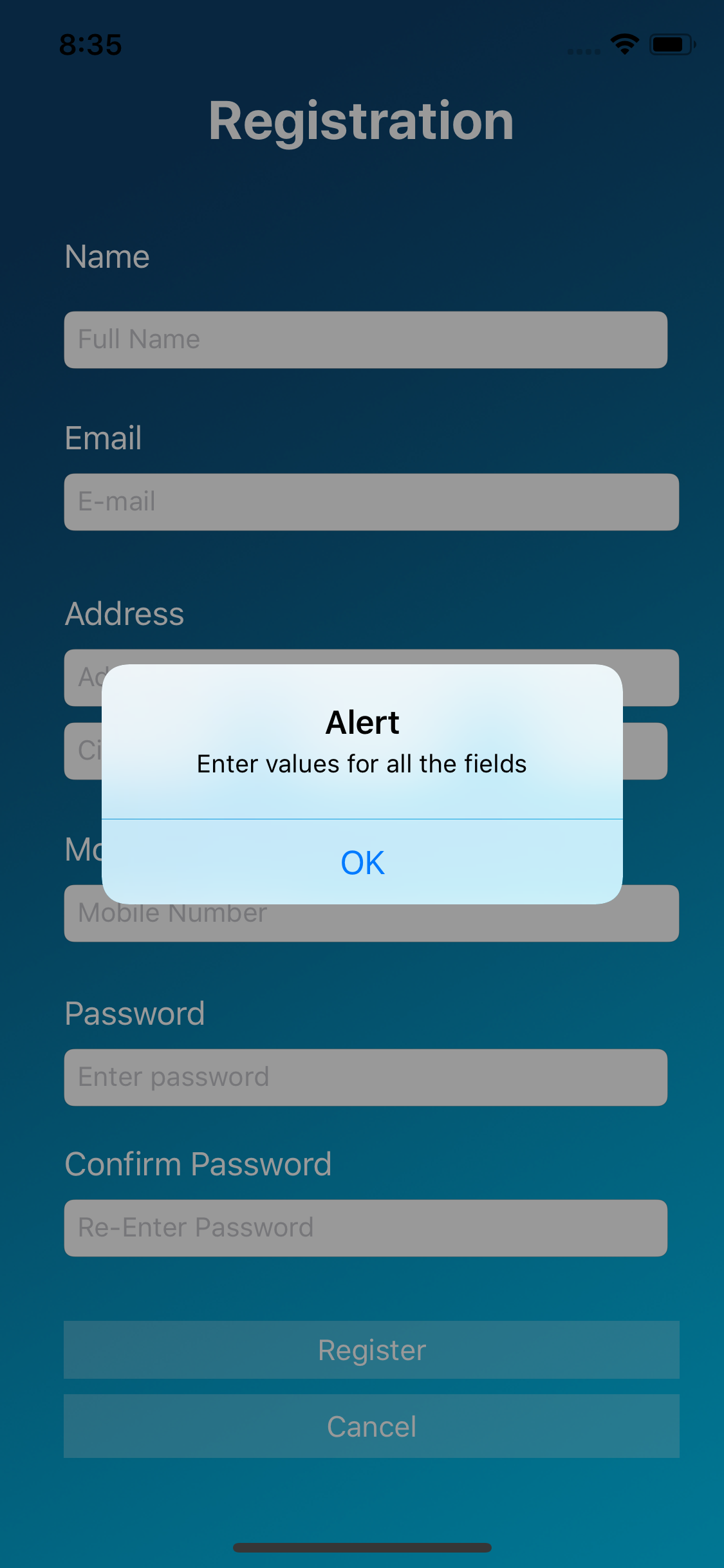
 

Fig.14: Registration page Fig.15: Registration Alert

**7.3 Forgot Password:**

When user clicks on forgot password he/she is redirected to forgot password screen. In forgot password user needs to give his registered e-mail address in requested field. If the e-mail is registered e-mail, user was sent a link to reset password. By clicking on that e-mail, user can reset his password.

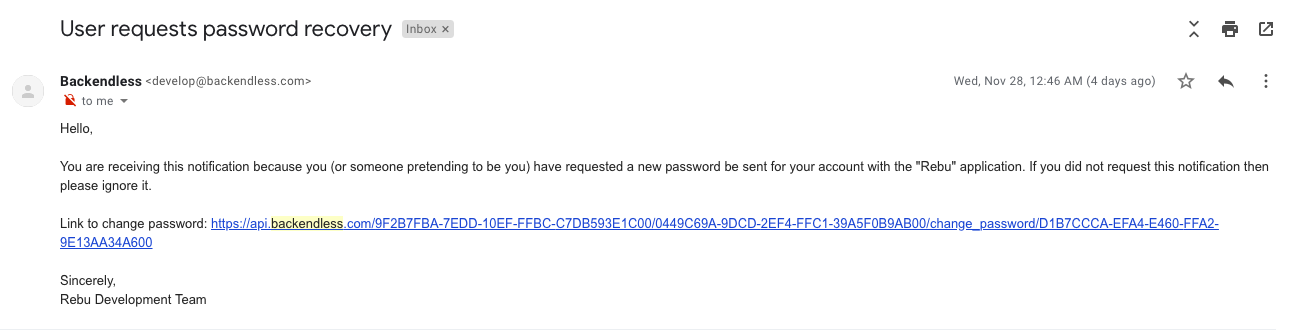


Fig.16: E-mail notification to reset password



Fig.17: Forgot password

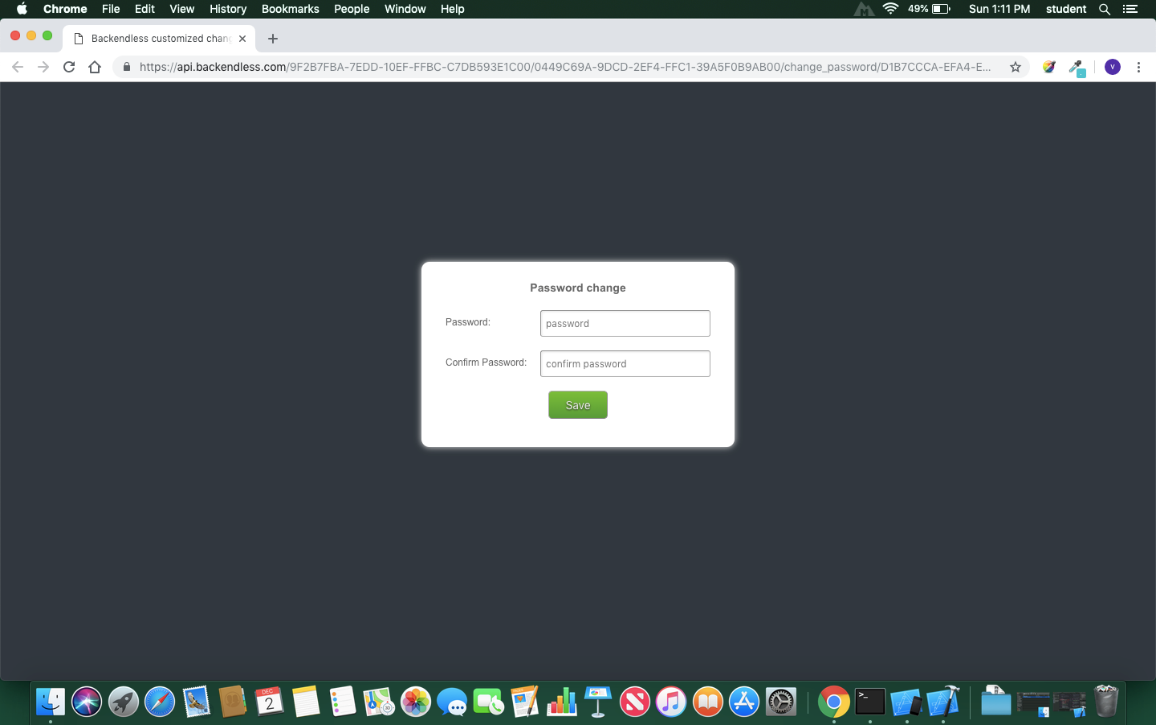


Fig.18: Reset password page

**7.4 Messaging:**

Admin and User can Directly send messages in this application. By clicking on messages tab, admin is redirected to chats page. By clicking on particular chat, admin can view message sent by user and he can reply to users messages as shown in the below screenshots.

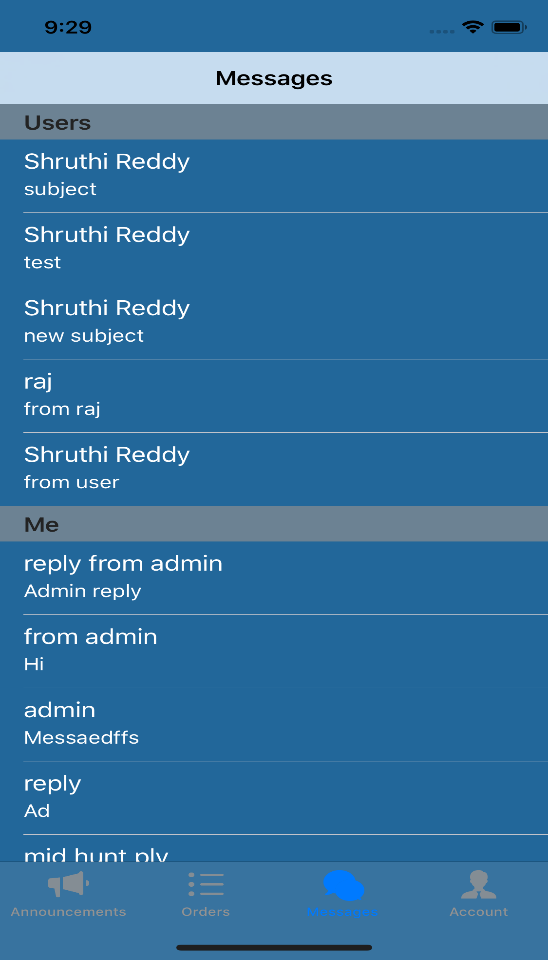
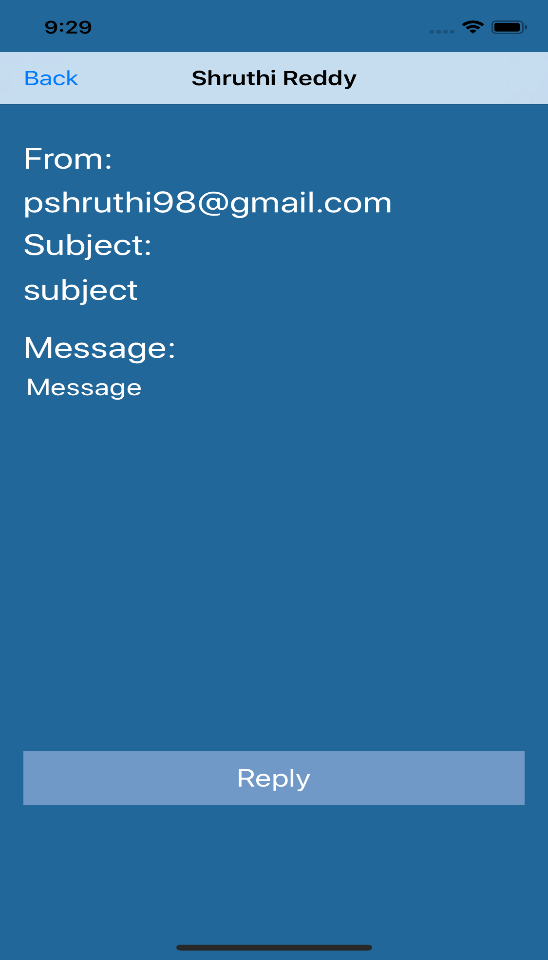
** **

Fig.19: Chats View Fig.20: Full Message Details

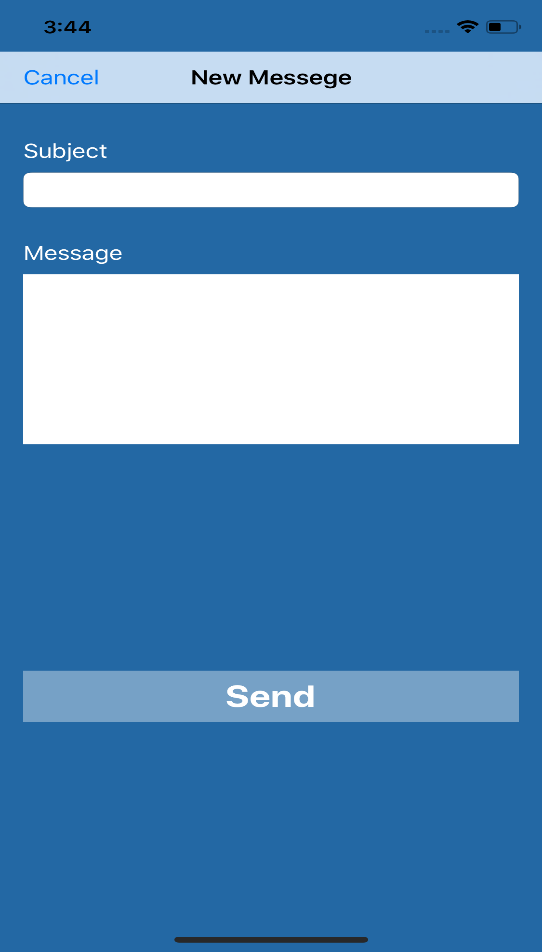
****

Fig.21: New Message View

**7.5 Announcements:**

Admin can view all the posted announcents in the announcement page. Admin has access to post, edit & delete announcement.

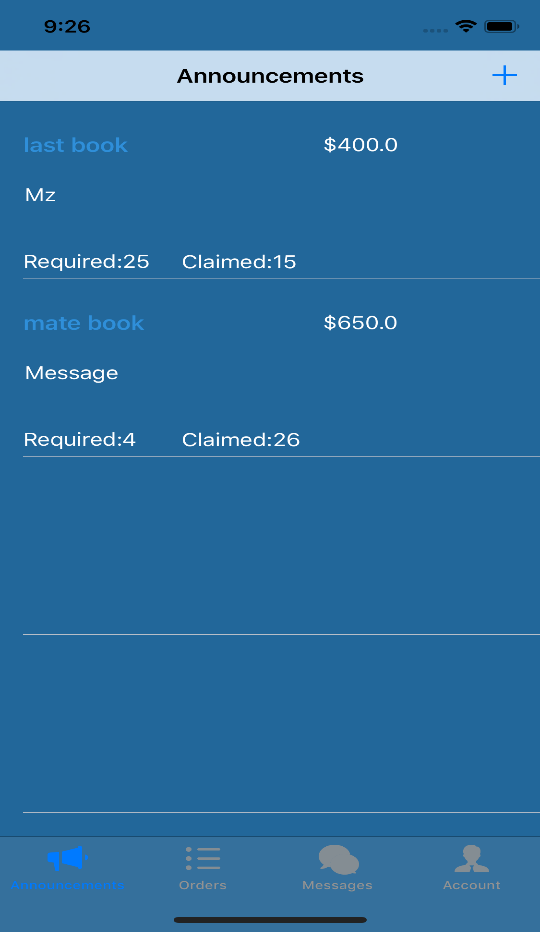


Fig.22: Announcements

**7.5.1 Post Announcement:**

To post an announcement admin has to click on ‘+’ sign in announcement page, it redirects to post announcements page. Admin has to fill all the fields product name, quantity, price of product, description of the product. Fields product name, quantity, price of product are required fields. Admin has to fill these three mandatory fields to post announcement. If admin tries to post announcement with any of these three required fields empty, alert message is displayed to fill mandatory fields.

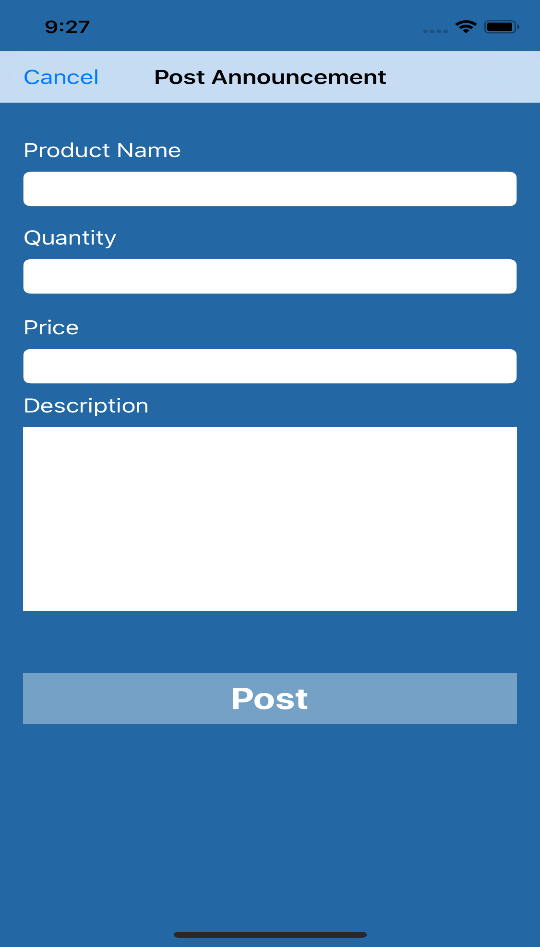
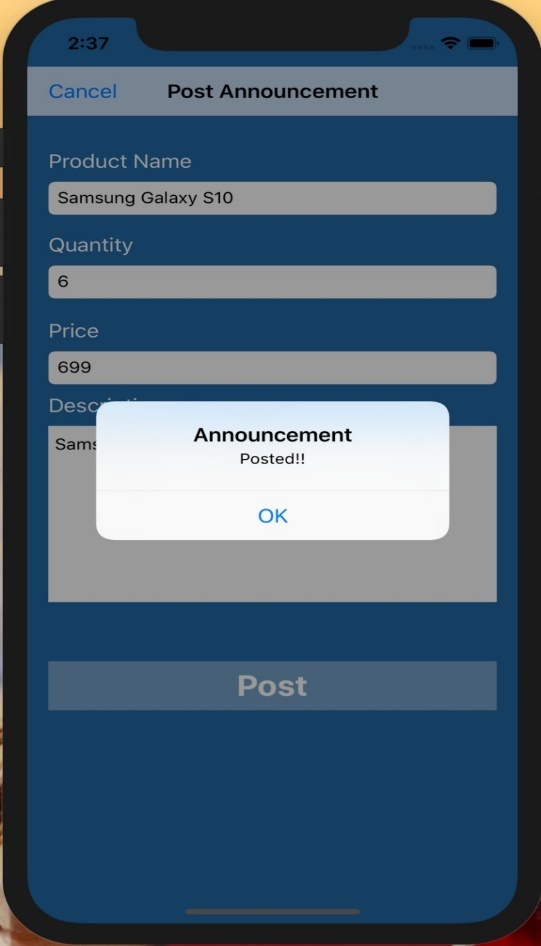
 

Fig.23: Post Announcement Fig.24: Login Failed Alert

**7.5.2 Update Announcement:**

Admin can update his already posted announcement. When admin clicks on already posted announcement it is redirected to product details page where admin can edit and update his previous announcement.

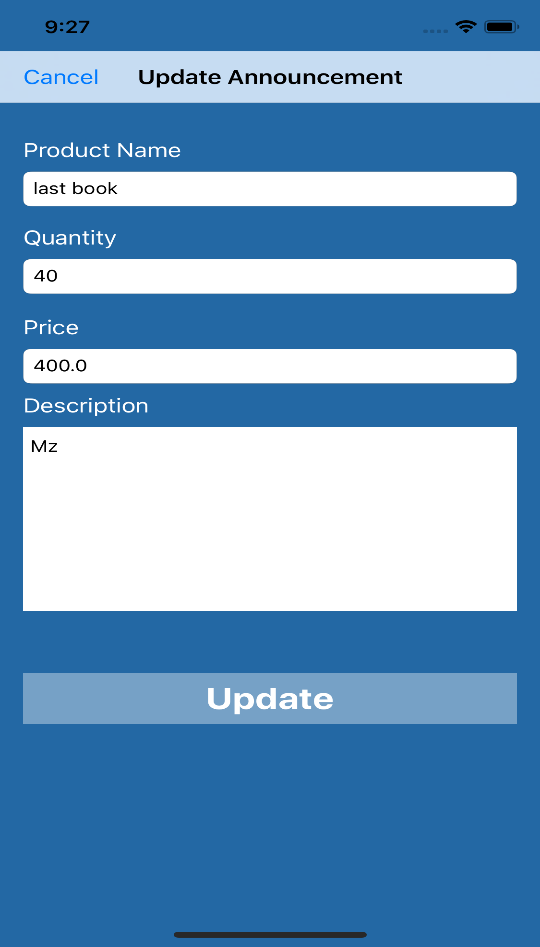
****

Fig.25: Update Announcement

**7.5.3 Delete Announcement:**

To delete an announcement admin can perform delete operation directly in announcement page by swiping product from right to left. Please see the below image.

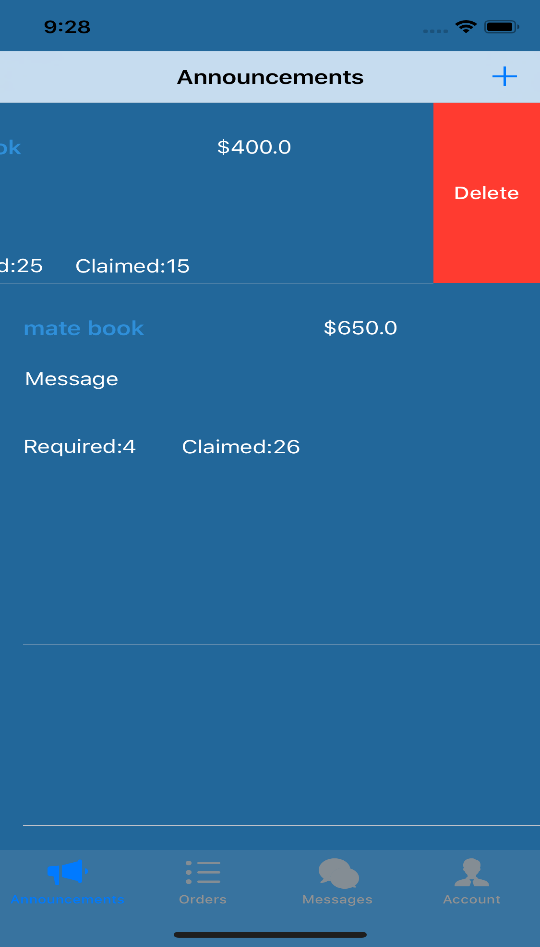
****

Fig.26: Delete Announcement

**7.6 Order View**

Admin can view all orders by clicking orders tab. Orders is displayed by Name of the user and quantity of products user sending to admin. Please refer below image. Admin can view order details by clicking on the required order he/she wants to view. By clicking “Confirm order” admin can confirm items shipped by user.

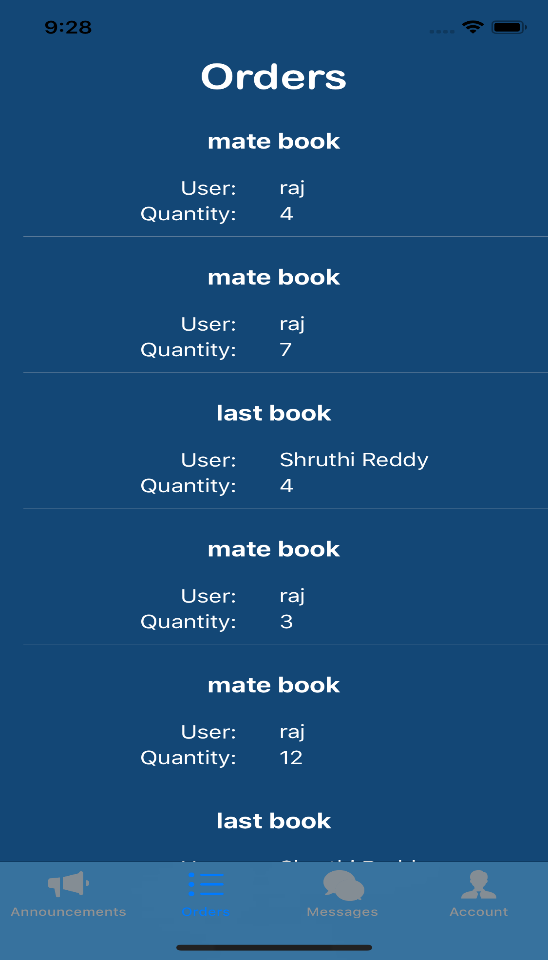
****

Fig.27: Orders View

**7.6.1 Approve and Reject User Claims:**

Admin can approve or reject user claims by swiping orders left. If the claim is approved, User can be able to request a shipping label to ship claimed products. If the claim is rejected order status is updated to rejected and user cannot request a shipping label.

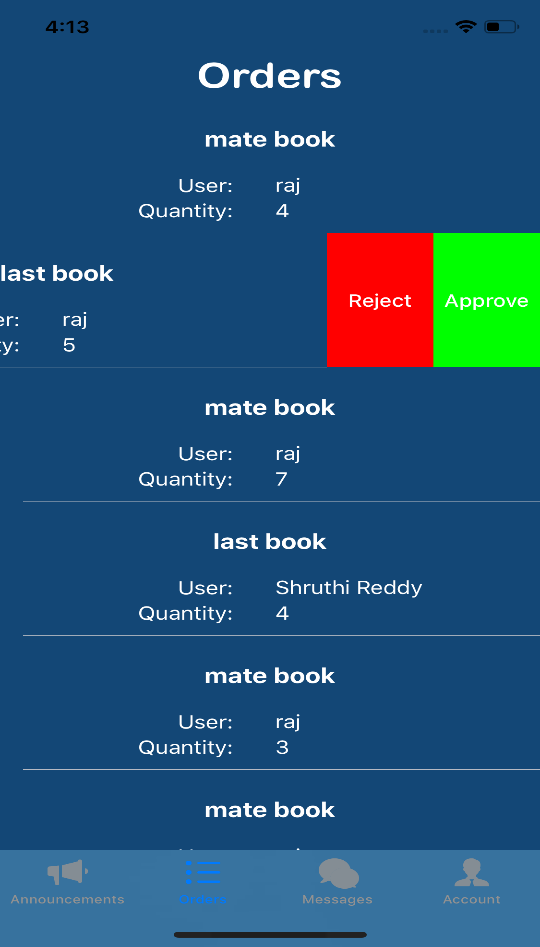


Fig.28: Approve or Reject View

**7.6.2 Upload Shipping Label:**

After Claimed products request is approved, user can request a shipping label for the products after the request is approved by admin. An Upload shipping label button is enabled where admin can choose a shipping label and upload via gallery.

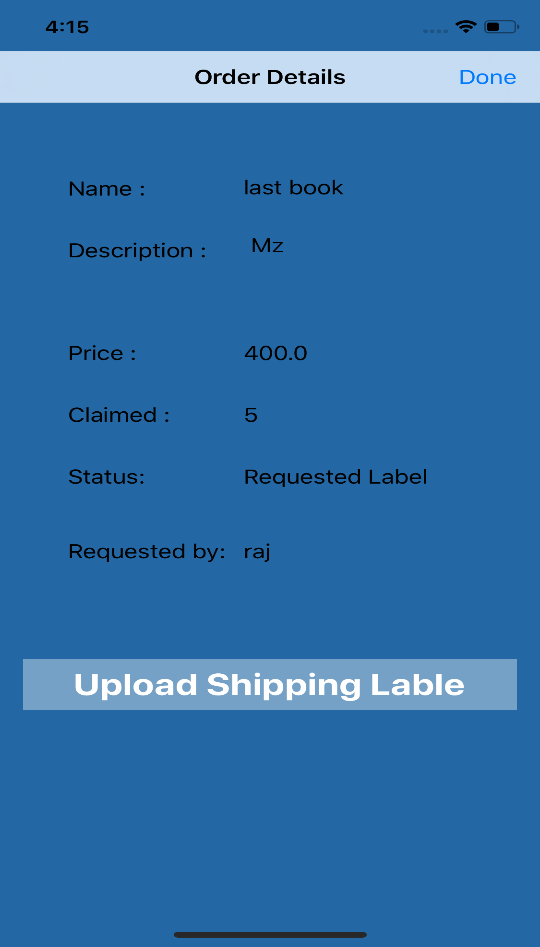
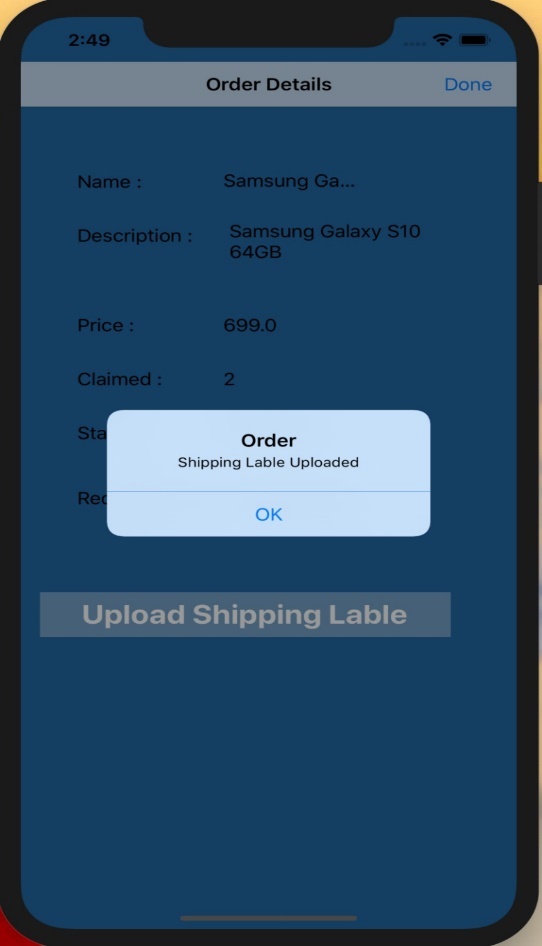
 

Fig.29: Upload Shipping Label Fig.30: Label updated alert

**7.6.3 Confirm Order:**

Admin can confirm order by looking at shipping receipt uploaded by user. It is just to conform whether user has shipped to correct address or not & claimed products or not.

****

Fig.31: Confirm Order

**7.7 Admin profile page**

In Admin profile page, details of the user are displayed. Like user’s name, e-mail address, address, phone number. Admin personal details are displayed here.

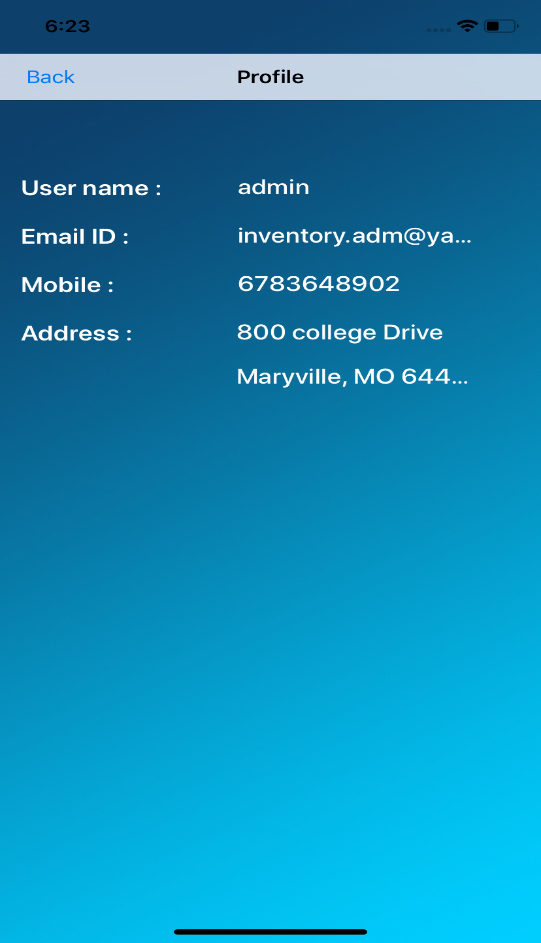


Fig.32: Admin profile page

**USER MANUAL**

**7.8 Dashboard:**

User can view all the announcements posted by admin on dashboard.

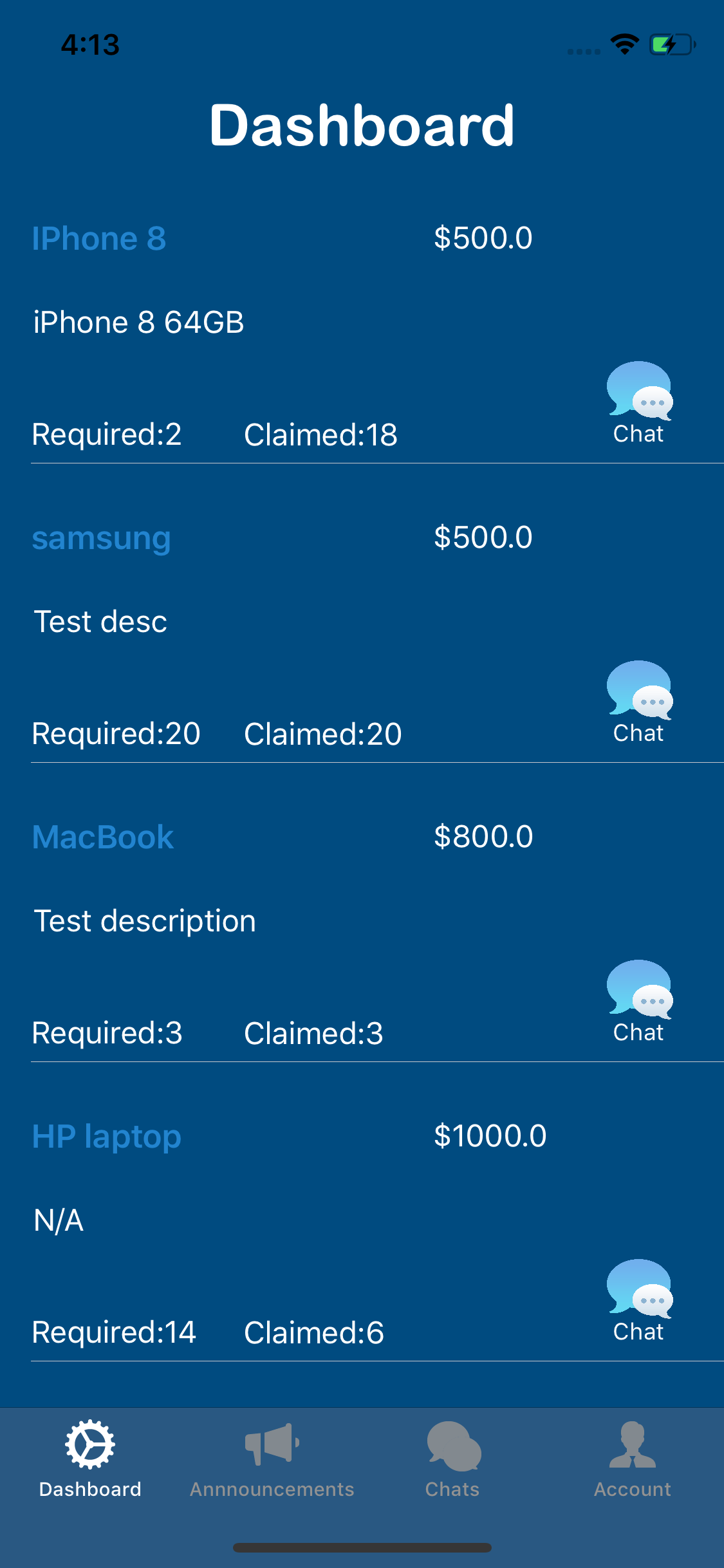


Fig.33: Dashboard

**7.8.1 Claim Products:**

From Dashboard, User can claim products, details of the product and can message to admin.

User can enter number of products he wants to claim. When user claims an order is created.

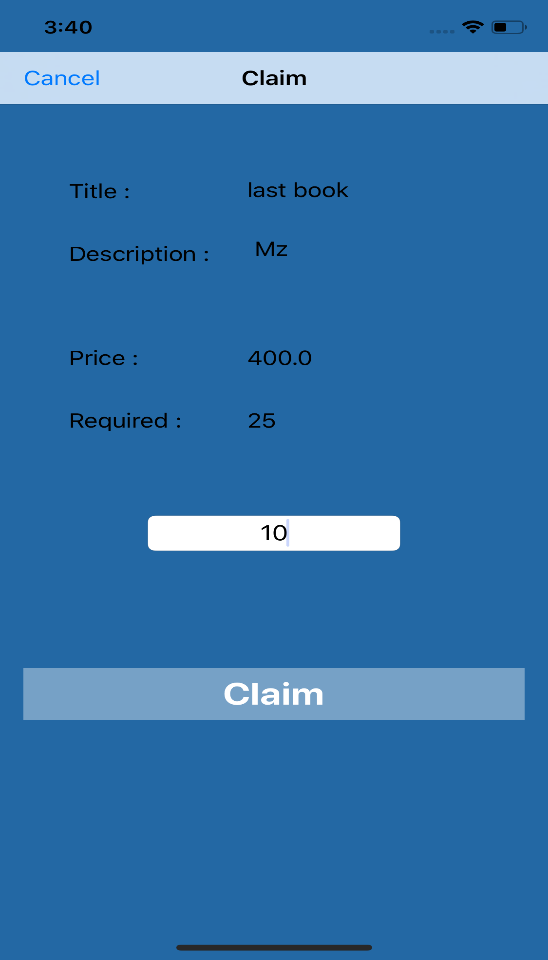


Fig.34: Claim products view

**7.8.2 Modify Claimed Products:**

User can modify his claim before admin’s approval, After admin’s approved his claim, he can’t modify the claim.

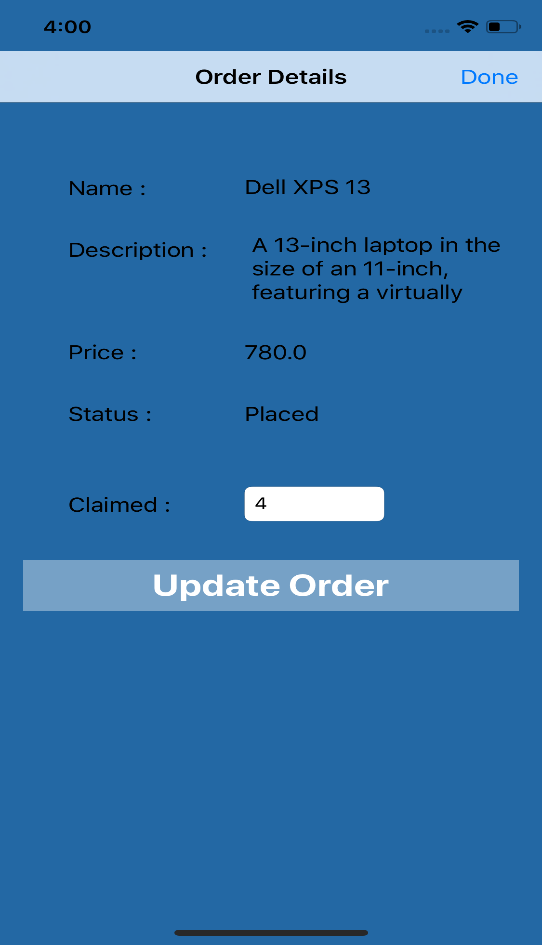
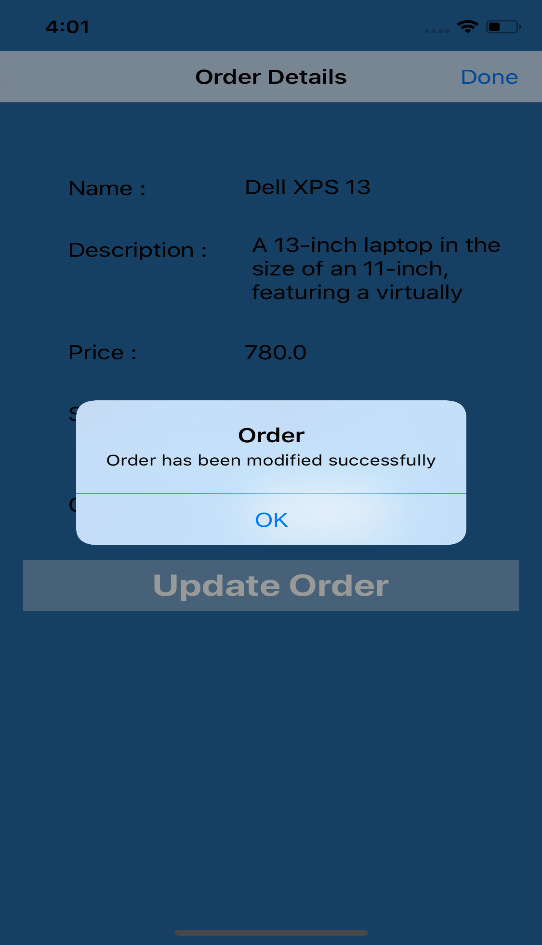
 

Fig.35: Update claimed order Fig.36: Update claimed order alert

**7.9 Order View**

User can view all orders by clicking orders tab. Orders is displayed by Name of the product, status of the order and quantity of products user sending to admin. Please refer below image. user can view order details by clicking on the required order he/she wants to view. Complete details of the order is displayed with number of products he claimed.

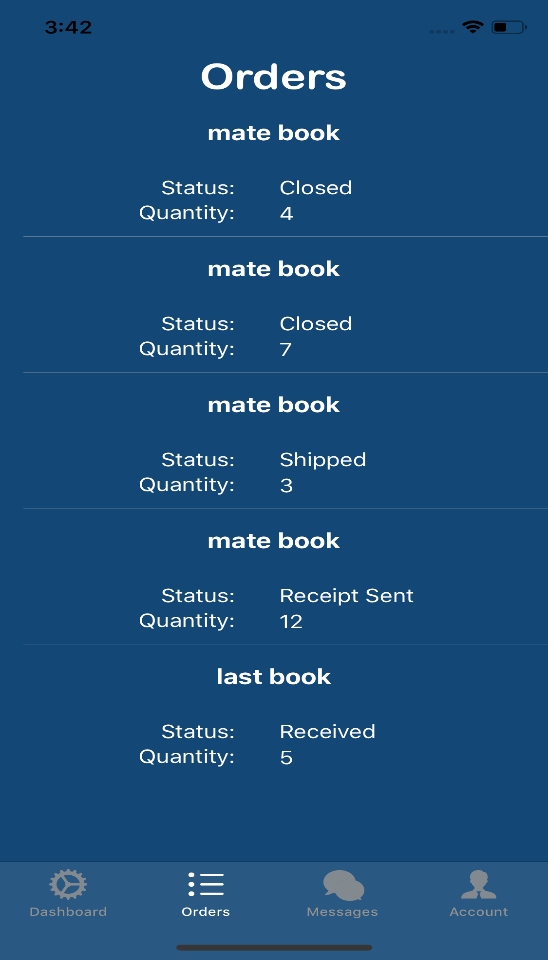
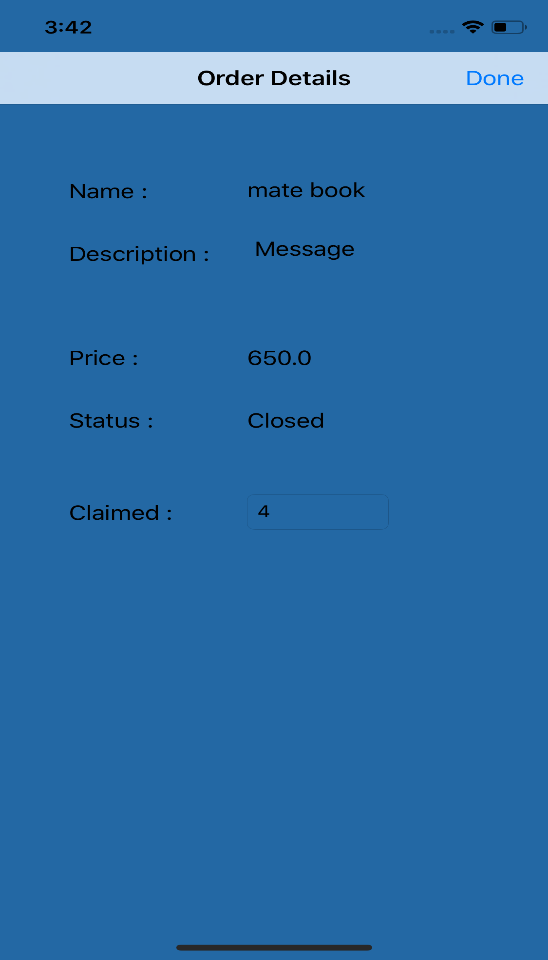
** **

Fig.37: Orders View Fig.38: Order Detailed View

**7.9.1 Request Shipping Label:**

Upon admin’s approval for claimed products, user can be able to request a shipping label for shipping the claimed products. After requesting shipping label a feedback message is sent to user. Feedback message to user “Requested shipping label successfully”.

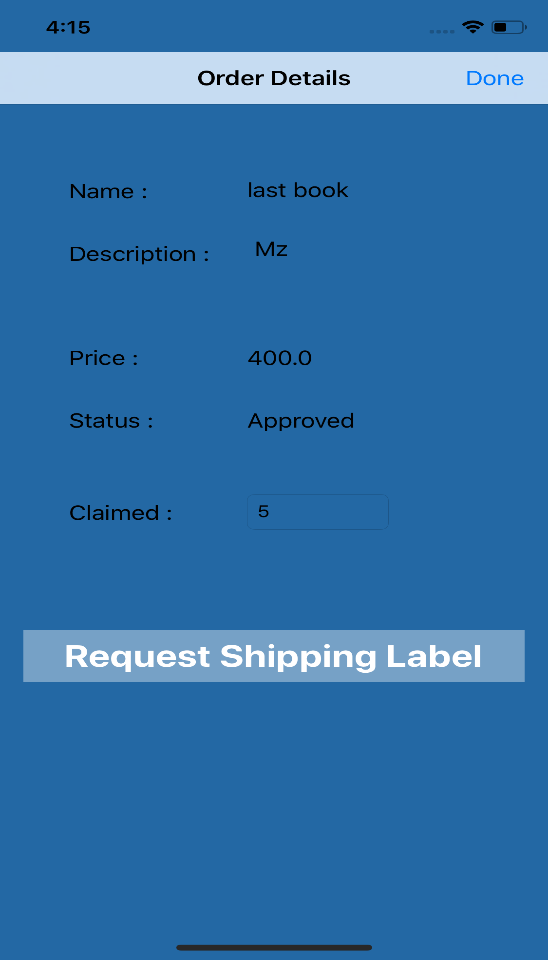
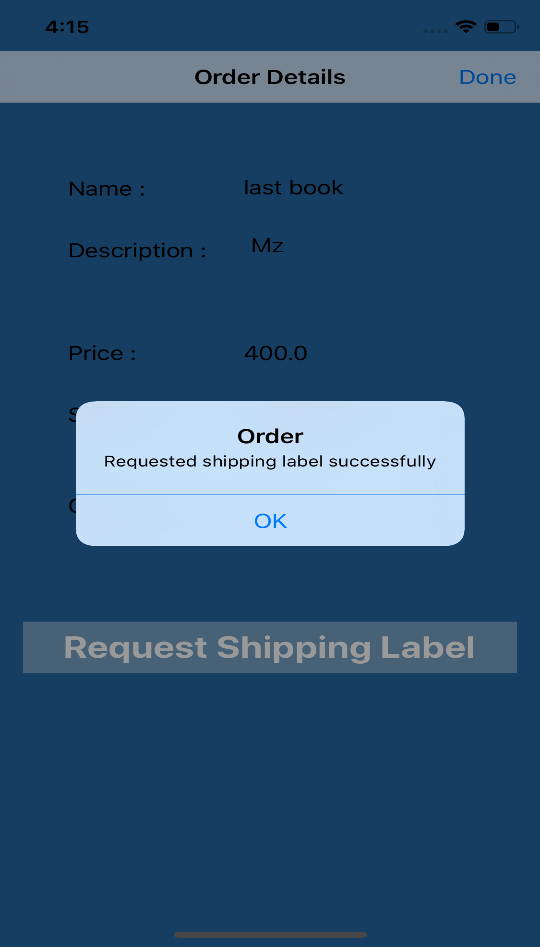
 

Fig.39: Request Shipping Label Fig.40: Request Shipping Label alert

**7.9.2 Download Shipping Label:**

After admin uploads shipping label, user can be able to download shipping label. When user clicks on download shipping label button, a label image is downloaded to user’s gallery.

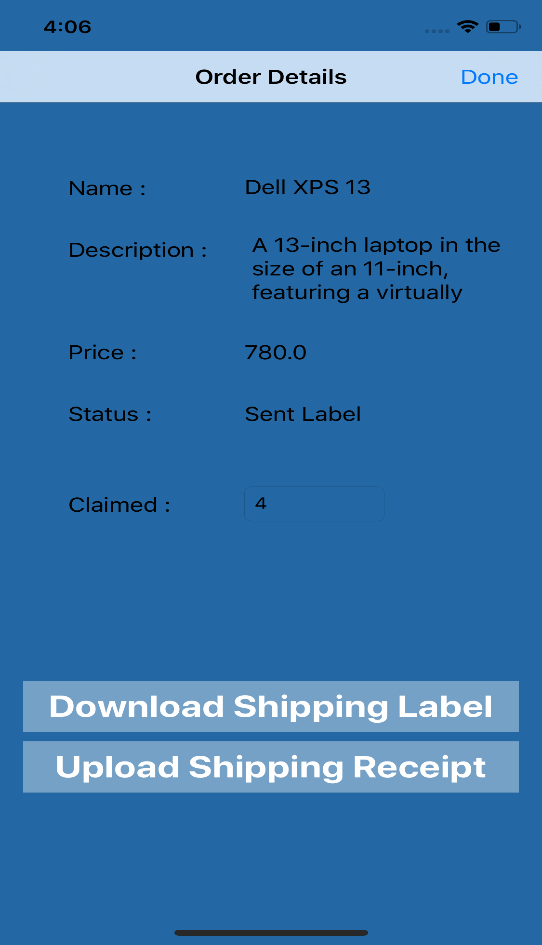
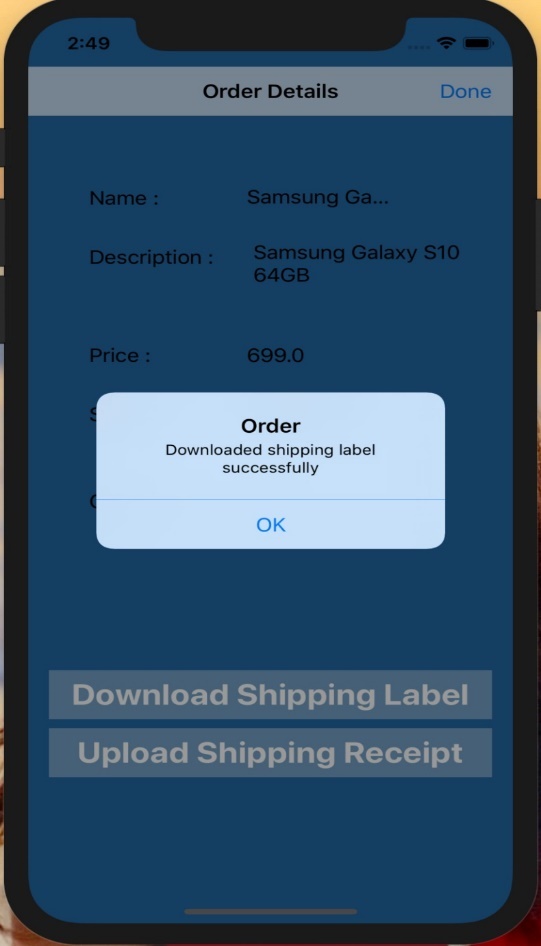
** **

Fig.41: Download Shipping Label Fig.42: Download Shipping Label alert

**7.9.3 Upload Shipping Receipt:**

After shipping the claimed products, user will upload shipping receipt via gallery to admin to confirm users order.

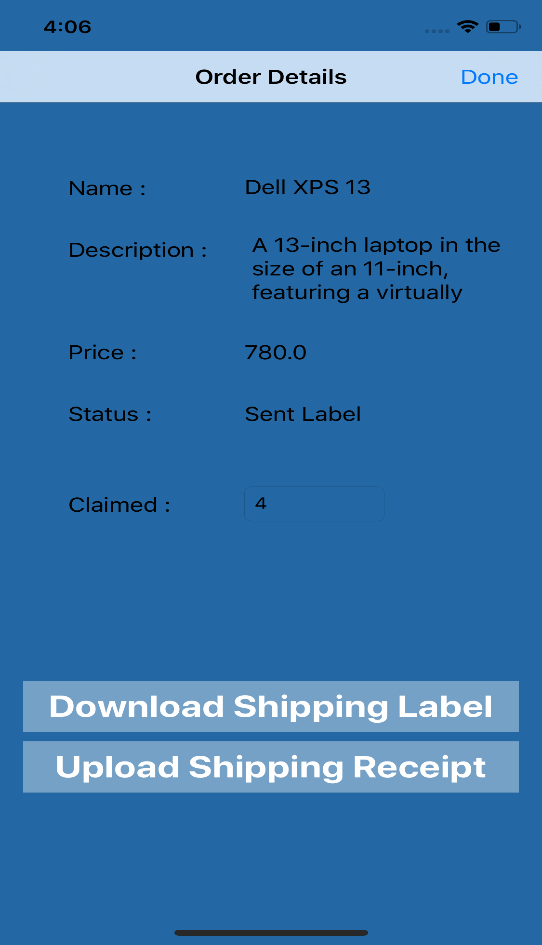
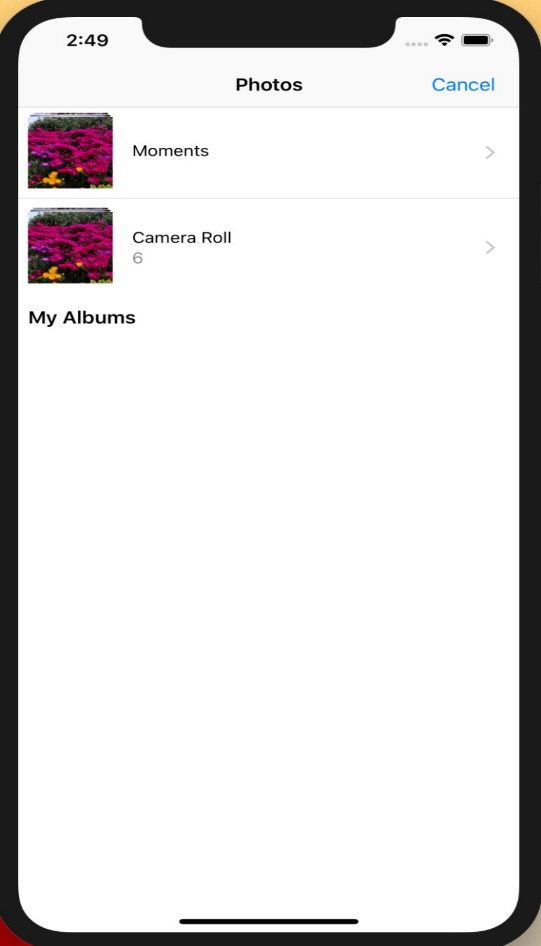
** **

Fig.43: Upload Shipping Label Fig.44: Users gallery.

**7.10 User profile page**

In User profile page, details of the user are displayed. Like user’s name, e-mail address, address, phone number. User personal details are displayed here.

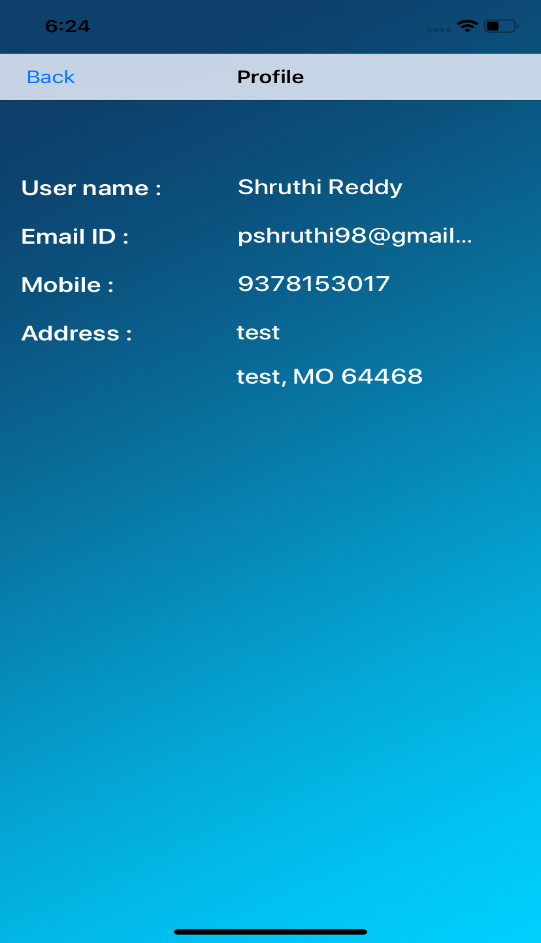
 

Fig.45: User Account Fig.46: User Profile Page

**7.11 Payment**

After receiving order, admin can update status to order received via a button. After status updated to order received user can request for payment. Admin order status will be updated to pay request and pay button is enabled. Admin can process the request and alert is generated according to the status of payment.

**7.11.1 Admin – Order Received:**

After receiving product admin can update status by clicking buttons “Order Received or Order Not Received”.



Fig.47: Order received view

**User – Request Payment:**

After admin update order status to order received a button is enabled for user to request for a pay.

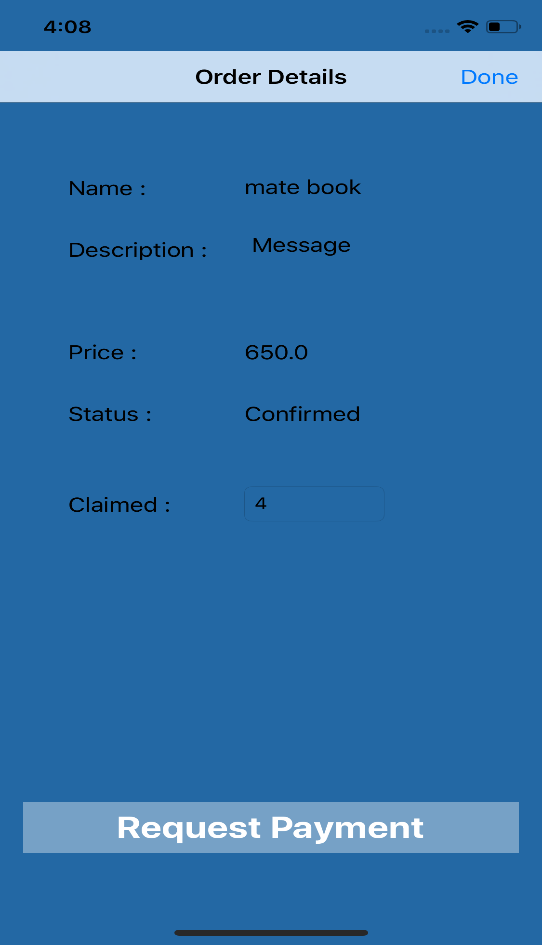
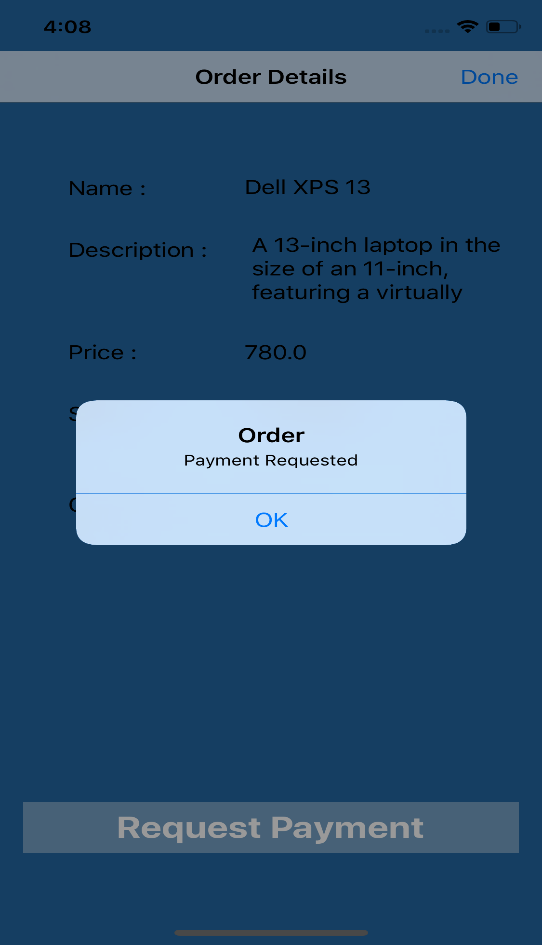
 

Fig.48: Request payment Fig.49: Request payment alert

**Admin – Record Payment:**

After users request for payment, admin can record payment

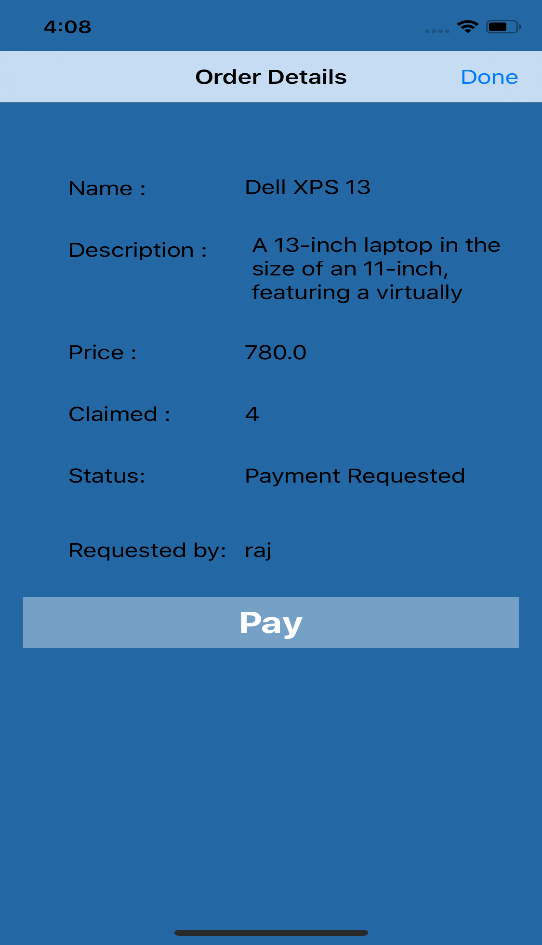
 

Fig.50: Record payment Fig.51: Record payment alert

**UNIT 8: TESTING**

**8.1 Login**

App users can login with valid credentials otherwise Feedback message must be displayed.

**Test Cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
| 1 | Username: pshruthi98@gmail.com  Password: \*\*\*\*\*\*\*\* | Login successful  Redirects to dashboard view | Logged in successful  Redirected to dashboard view | None | Success |
| 2 | Username: pshruthi98@gmail.com  Password: | Login Failed: login or password missing or null | Login Failed: login or password missing or null | None | Success |
| 3 | Username:  Password: \*\*\*\*\*\*\*\* | Login Failed: login or password missing or null | Login Failed: login or password missing or null | None | Success |
| 4 | Username: pshruthireddy@gmail.com  Password: \*\*\*\*\*\*\*\* | Login Failed: Invalid login credentials | Login Failed: Invalid Login and password | None | Success |

**8.2 Registration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
| 5 | Name: Vamshi Raj  Email: [raj.jennaikode@gmail.com](mailto:raj.jennaikode@gmail.com)  Address: 1121 N college drive, Maryville, MO – 64468  Mobile: 9012273248  Password: \*\*\*\*\*\*\*\*  Confirm Password: \*\*\*\*\*\*\*\* | Registration Successful | Registration Successful | None | Success |
| 6 | Name: Shruthi Reddy  Email: [pshruthi98@gmail.com](mailto:pshruthi98@gmail.com)  Address: 1121 N college drive, Maryville, MO – 64468  Mobile: 9013273248  Password: \*\*\*\*\*\*\*\*  Confirm Password: \*\*\*\*\*\*\*\* | Account already exists | Account already exists | None | Success |
| 7 | Name: Vamshi Raj  Email: [raj.jennaikode@gmail.com](mailto:raj.jennaikode@gmail.com)  Address: 1121 N college drive, Maryville, MO – 64468  Mobile:  Password: \*\*\*\*\*\*\*\*  Confirm Password: \*\*\*\*\*\*\*\* | Enter values for all fields | Alert: Enter values for all fields | None | Success |
| 8 | Name: Vamshi Raj  Email: [raj.jennaikode@gmail.com](mailto:raj.jennaikode@gmail.com)  Address:  Mobile: 9012273248  Password: \*\*\*\*\*\*\*\*  Confirm Password: \*\*\*\*\*\*\*\* | Enter values for all fields | Alert: Enter values for all fields | None | Success |

**8.3 Forgot Password**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
| 9 | Email: pshruthi98@gmail.com | Reset password link is sent to the registered email | Reset password link is sent to the registered email | 5 | Success |
| 10 | Email: pshruthi99@gmail.com | Reset password link is not sent to email because email is not the registered email | Reset link is not delivered | 5 | Success |
| 11 | Password: \*\*\*\*\*\*\*\*  Confirm Password: \*\*\*\*\*\*\*\* | Password reset successful | Password reset successful | 9 | Success |
| 12 | Password: \*\*\*\*\*\*\*\*  Confirm Password: \*\*\*\*\* | Password reset not successful | Password reset not successful | 9 | Success |

**8.4 Messaging**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
| 13 | Subject: Claimed products  Message: Can I know reason to reject my products | Message is delivered to admin | Message is sent to admin by user | 1 | Success |
| 14 | Reply to user: You requested too many products | Admin replied to user | Reply message is successfully delivered to user by admin | 1,13 | Success |
| 15 | Subject: Request for payment products  Message: When you are going to do transaction | Message is sent to admin by user | Message is successfully delivered to user | 1 | Success |
| 16 | Reply from admin: I will be doing soon after your products arrived | Admin reply sent to user | Reply message delivered successfully | 1,15 | Success |

**8.5 Profile Page**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.6 Post Announcement**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.7 Edit Announcement**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.8 Delete Announcement**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.9 Approve or Reject claimed products**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.10 Upload shipping label**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.11 Confirm order**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.12 Record Payment**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.13 Dashboard**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.14 Claim products**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.15 Modify claimed products**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.16 Request Shipping label**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.17 Download shipping label**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.18 Upload shipping receipt**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**8.19 Request Payment.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Inputs** | **Expected Result** | **Actual Result** | **Pre-deccessor** | **Testcase Result** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |