**First increment Report**

**Existing Services/API:**During the first increment, we have implemented Registration and Login services. We even developed our own API (using Apache CXF for REST service, spring dependency injection, Hibernate to map java classes to database tables and MySQL database.)   
We are going to use some existing API’s in the second iteration.

<http://www.supermarketapi.com/>

<http://developer.groceryserver.com/>

<http://search.ams.usda.gov/FarmersMarkets/v1/data.svc?wsdl>

<https://www.zomato.com/api>

While developing our project during first phase, we used information like class’s overview, developer guides and some tutorials from the following sources (mentioned below) while implementing the features like shared preferences, using RESTful web service from android, Toasts.

<http://developer.android.com/reference/android/content/SharedPreferences.html>

<http://www.tutorialspoint.com/android/android_shared_preferences.htm>

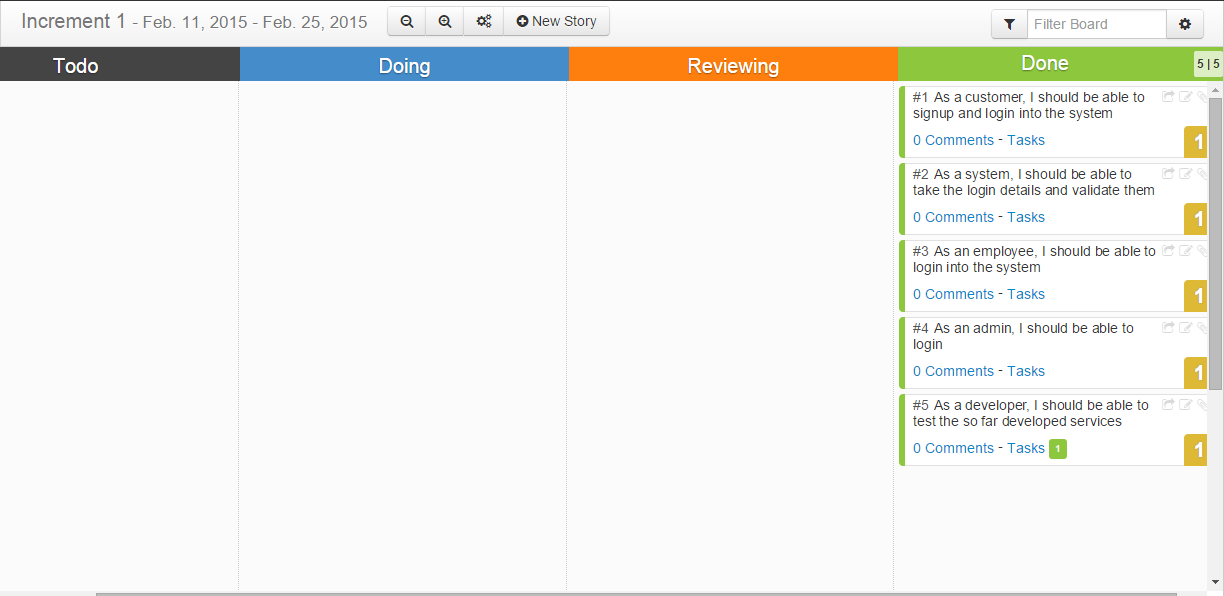
<http://www.lynda.com/Android-tutorials/Connecting-Android-Apps-RESTful-Web-Services/163757-2.html>

<http://stackoverflow.com/questions/6047194/how-to-call-restful-web-service-from-android>

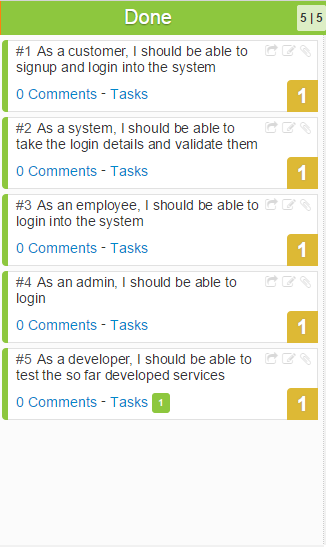
**Detail design of Services:**

So far we have developed registration and login services (developed our own API). We will explain in detail about the design and implementation of REST services in later sections.

The picture below shows the stories and progress of our project at the end of first iteration.



**User stories for signup and login services:**

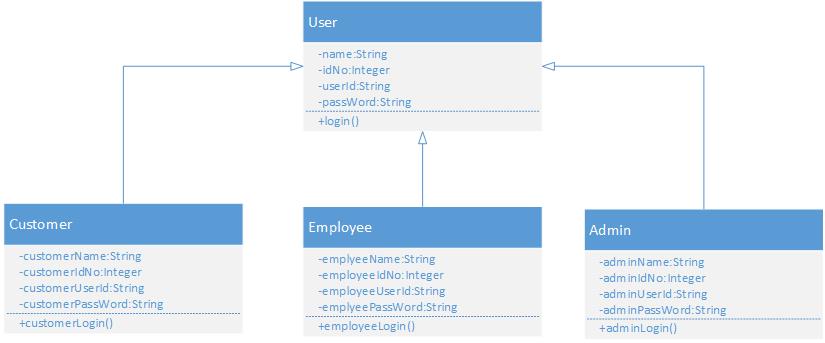


**Service description:**

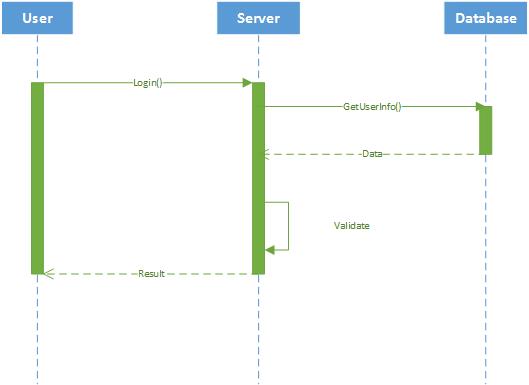
In the application we are developing, a user should be able to register and login into the application to select the items and add them to cart and order. An employee should be able to login into the application to view and process the orders. An admin should be able to login to track the transactions, update items in database and registers new employees.

Hence login is essential for all types of application users.

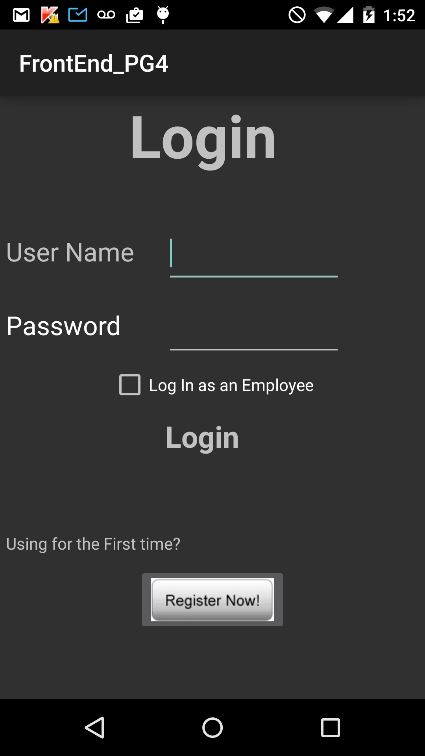
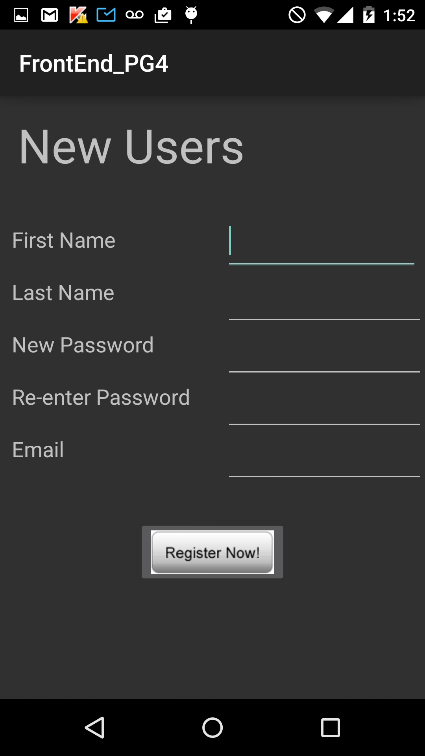
**Class diagram:**



**Sequence diagram:**

****

**Design of mobile client interface:**Since we have developed the login and signup services, these (below) are the interfaces for login and register. We concentrated mainly on functionality and less on user interface.

In the first iteration, we worked more on backend (creating our own API service) and functionality of the mobile client. We will concentrate and improvise the user interface during the fourth iteration.

**Implementation:**

**Implementation of REST services:**

We implemented the REST services using Apache CXF. It is open source and multi featured web services framework. It takes care of all the http requests. Every request from the mobile client goes to CXF servlet.

**Implementation of user interface:**

For this iteration, we totally used five activities (application pages). The first activity (Home page) is the login page where customer, employee and admin will be able to login to the application. We used two buttons for login and registering. Upon entering the credentials into the username password fields and taping the login button, user dashboard page will appear (if the credentials are correct). A new user can register using the register page. Upon taping the Register Now button in the login page, the registration page will appear. After entering the values into respective fields and taping the Register Now button, registers the new customers. If the customer is registering for the second time, he will see a toast saying him that he already registered.

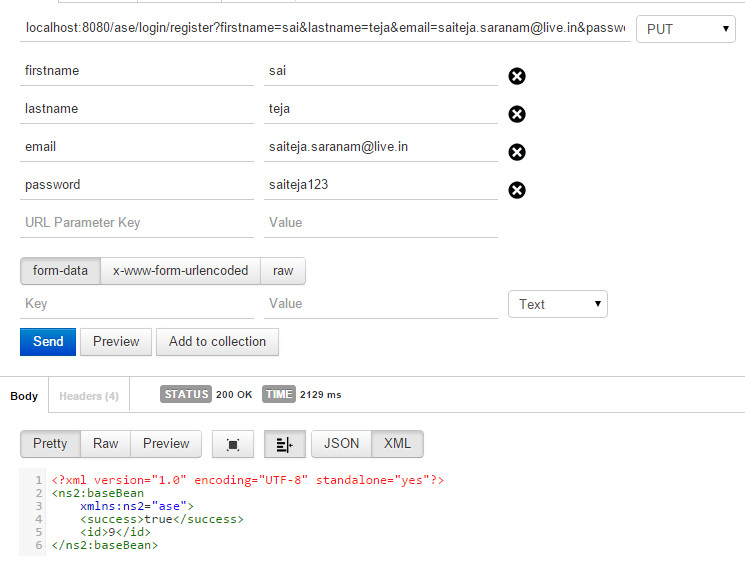
There’s a check box in the login page which is checked only when an employee or admin login to the application. If the employee login to the application, employee page will appear (In the later iterations all the orders placed by the customers will appear here). If admin logs into the application, an admin page will appear.

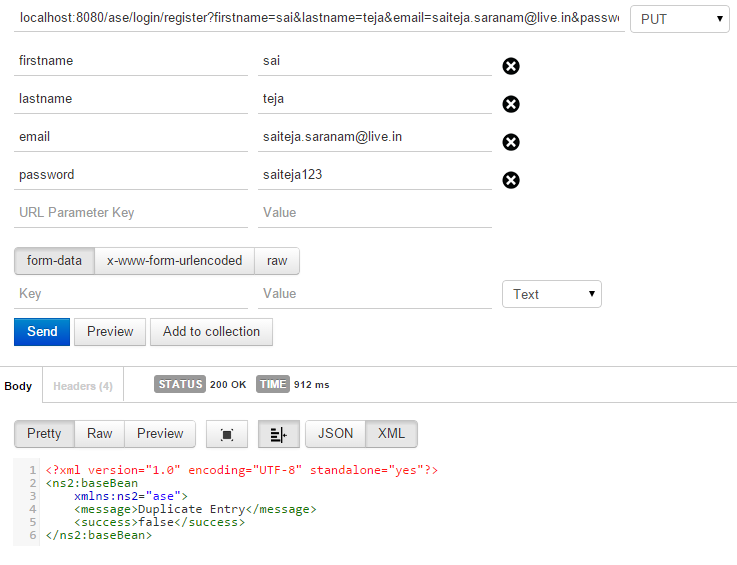
**Testing:**

As we implemented our own API, we tested it using Postman (an extension of chrome browser).

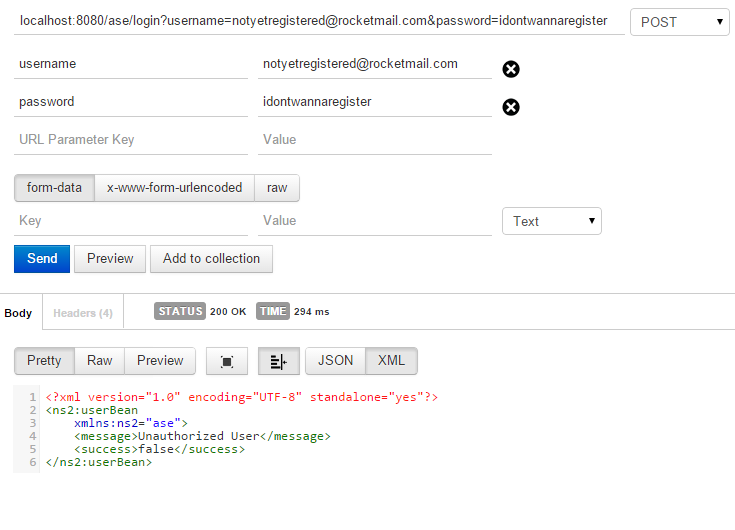
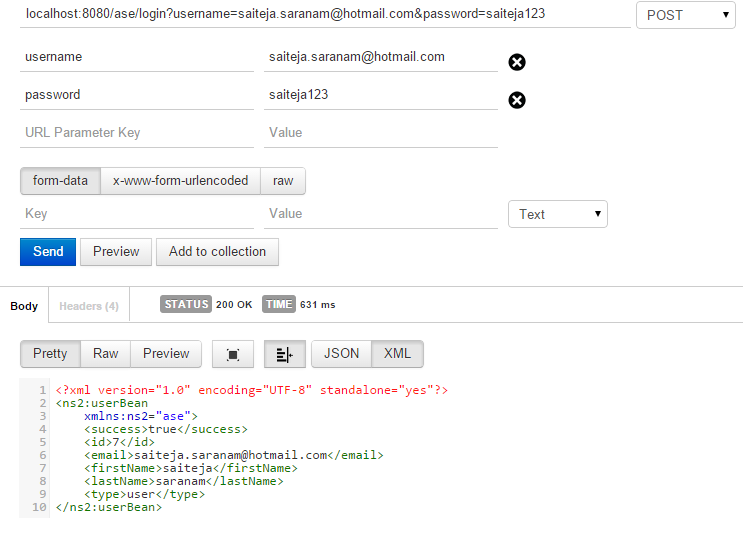
**Testing of Sign Up service :**

Sign up:

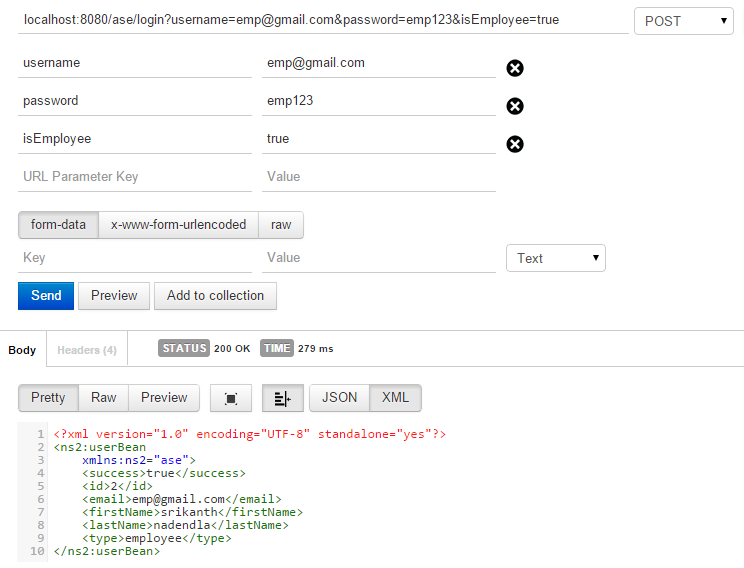


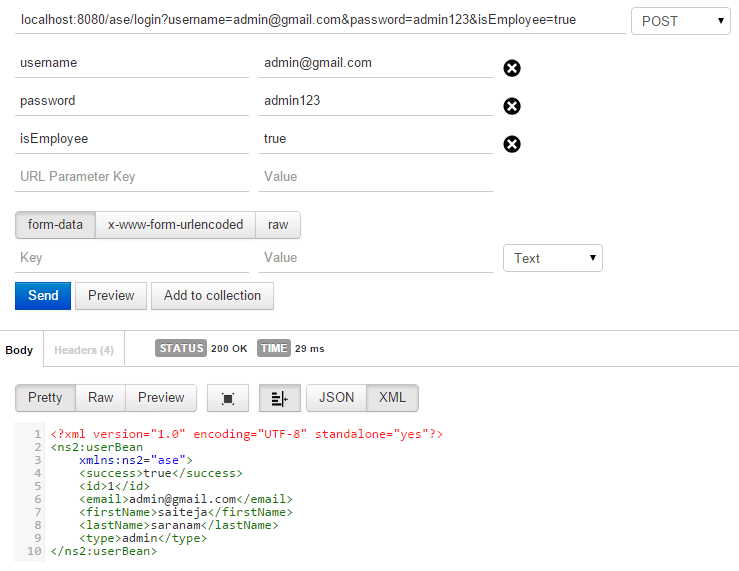


Testing of Login service:

User login: 

Employee login:

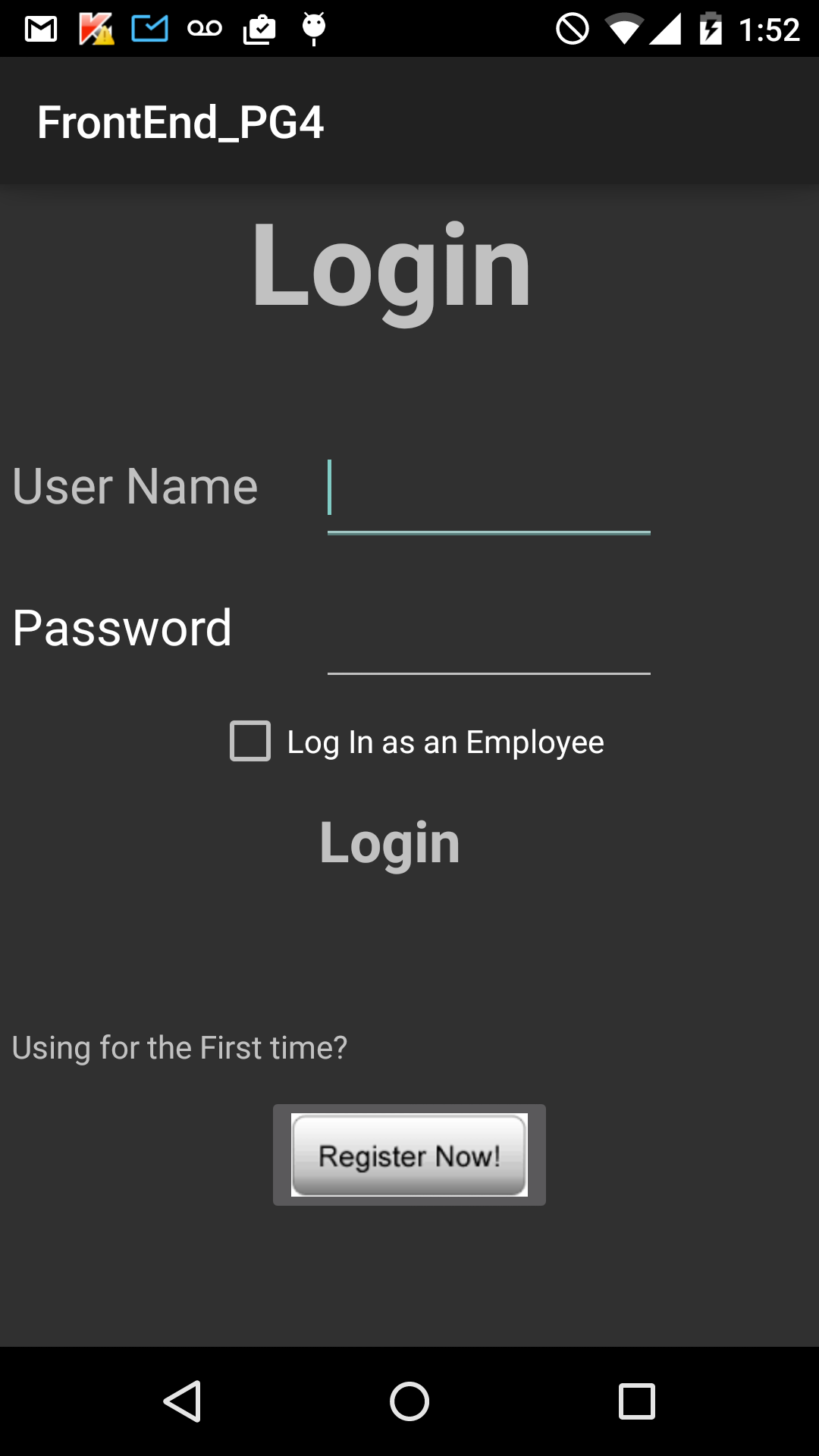


Admin login: 

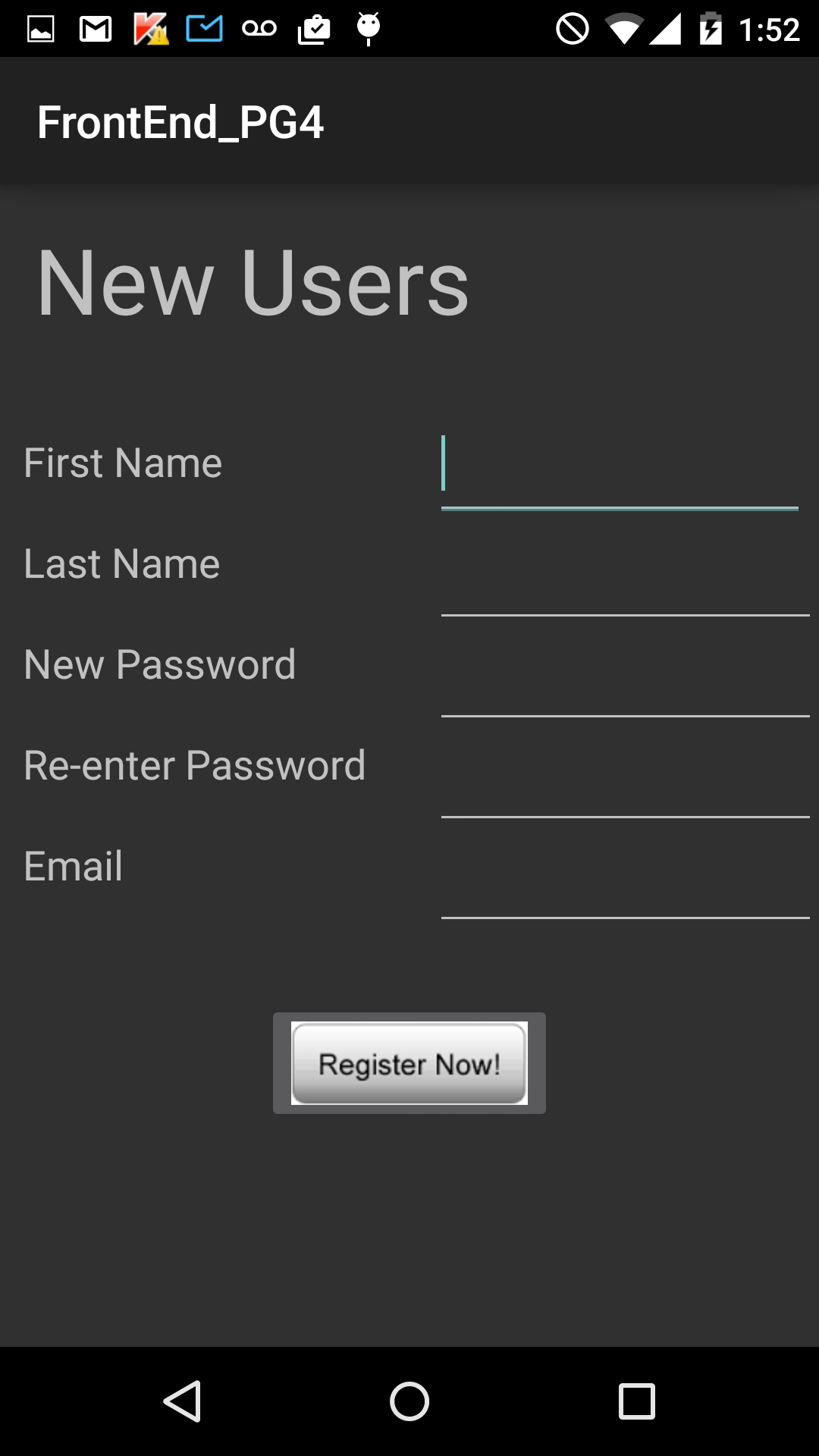
**Report:**

Mobile client Screenshots:

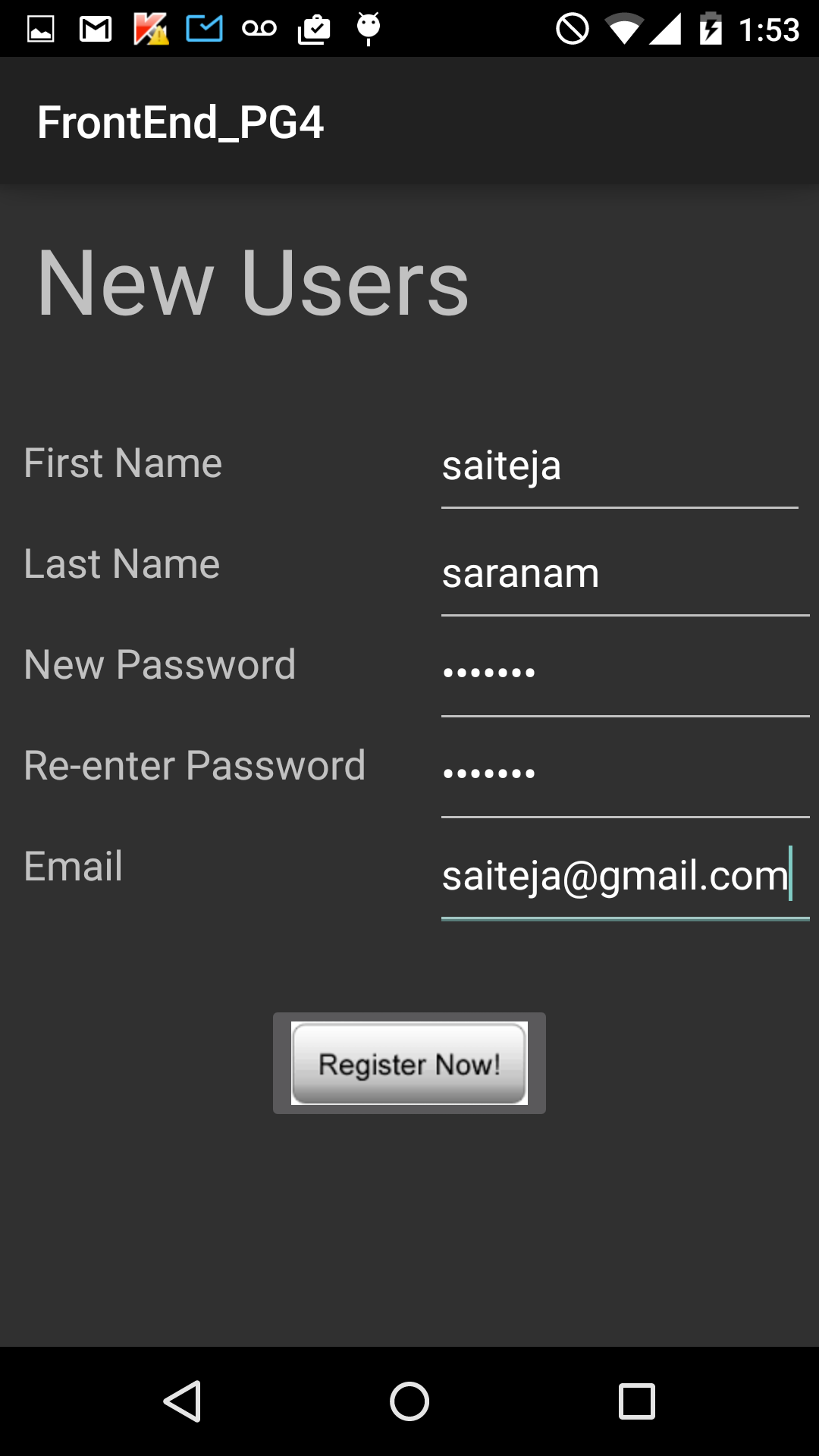
Login page (Home page):

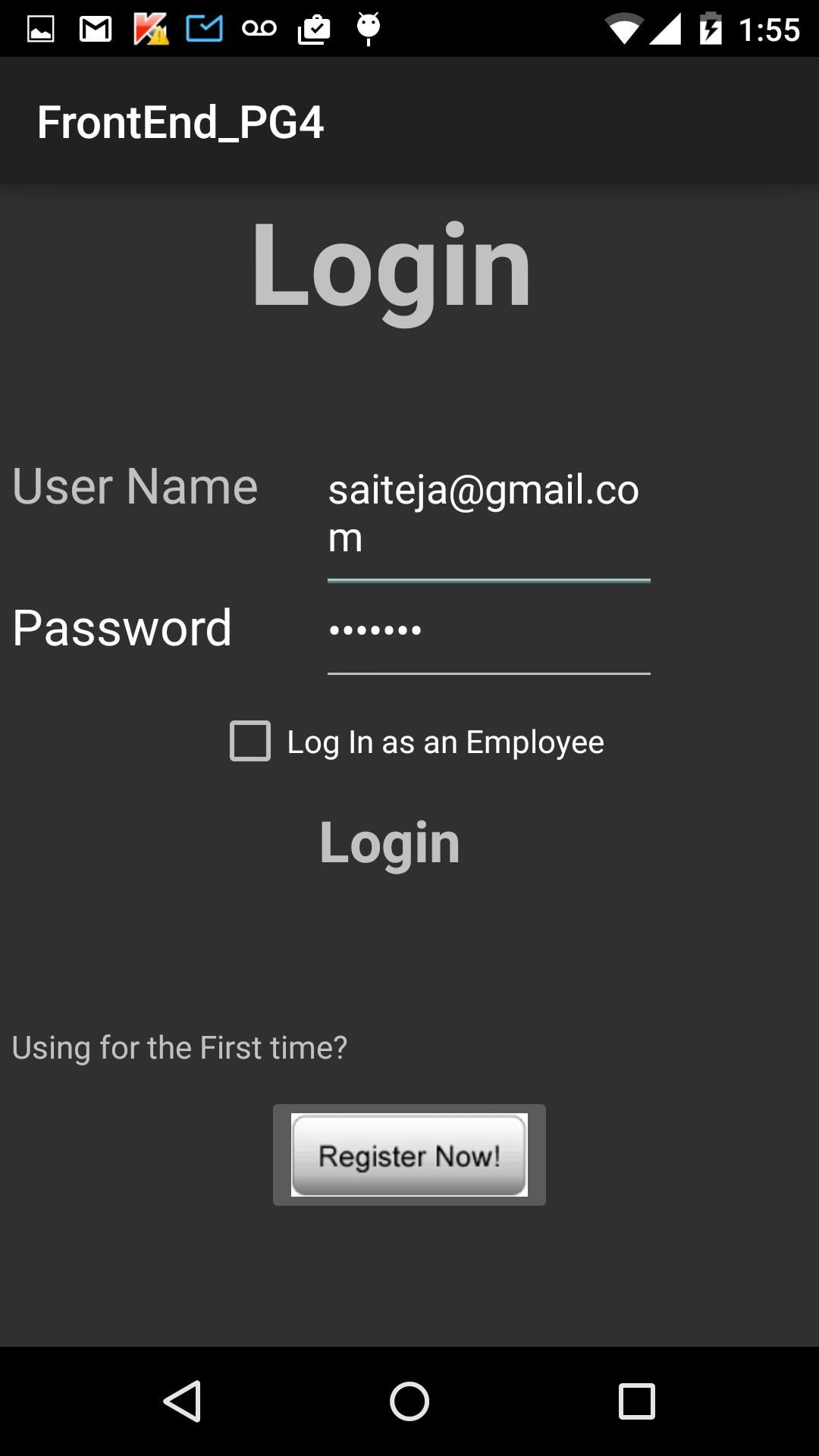


Sign up page:



While Registering:

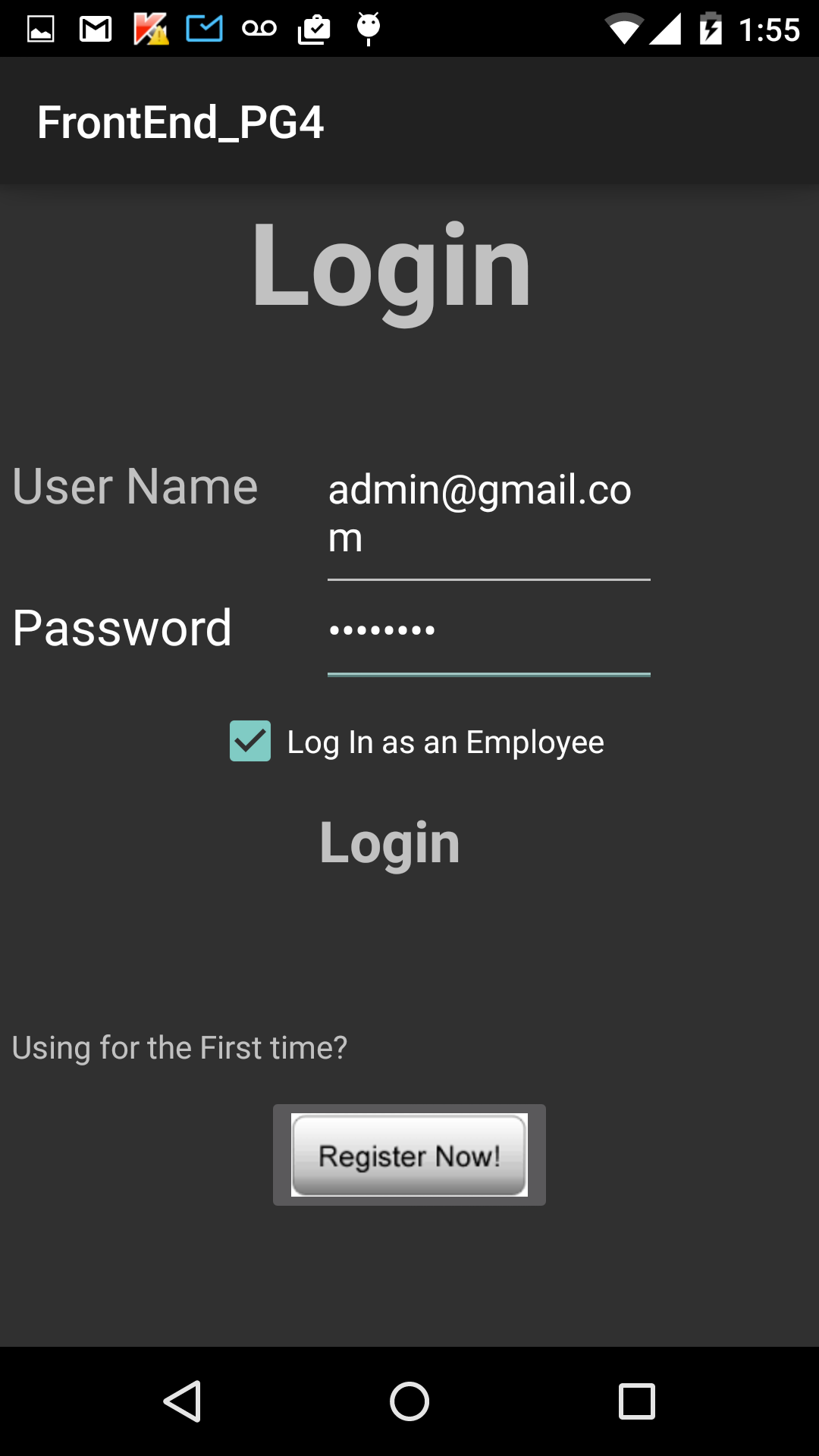


While Signing in (user sign in):  


After successful sign in (user):



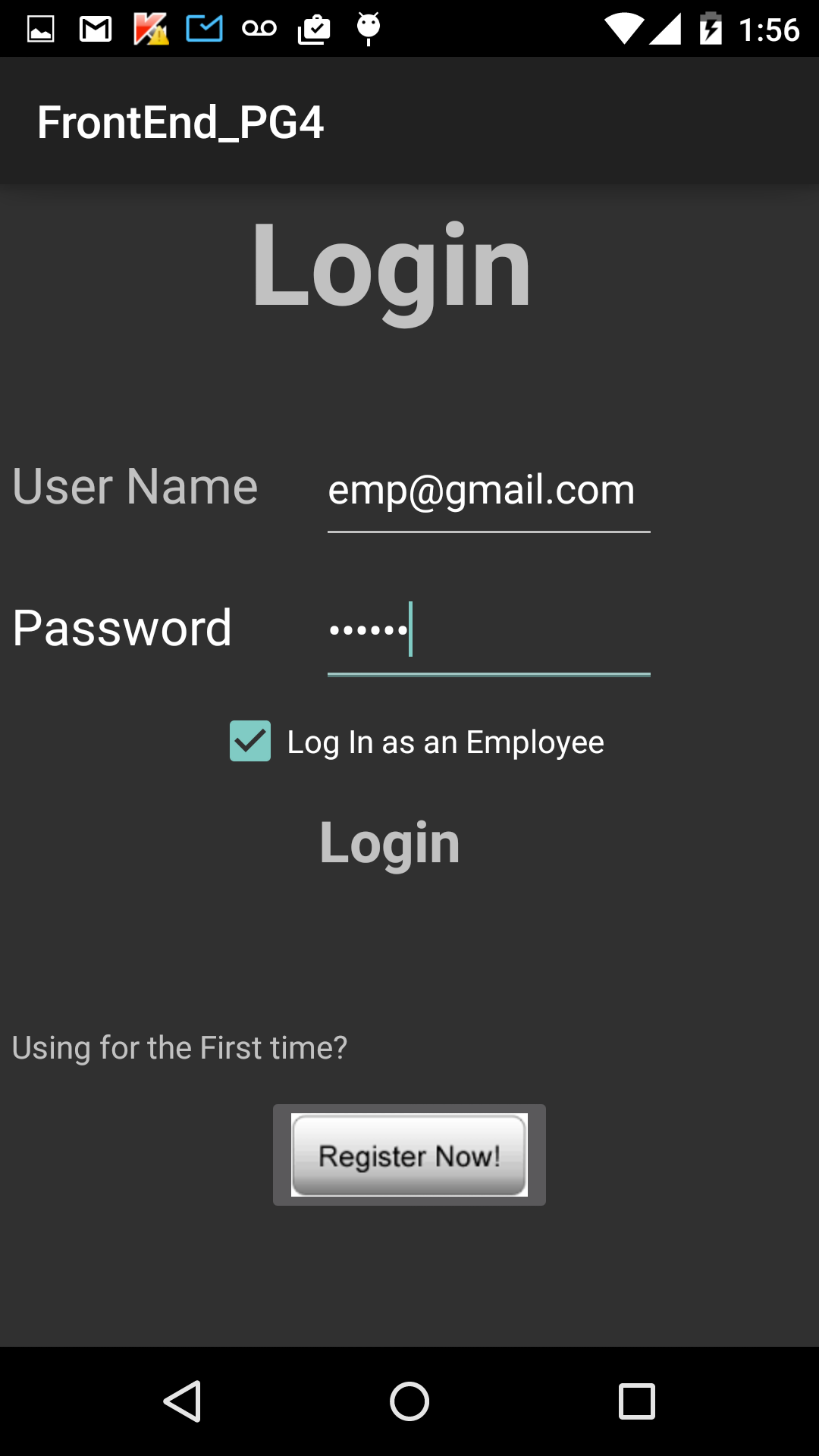
Admin sign in:



Upon successful sign in (admin):



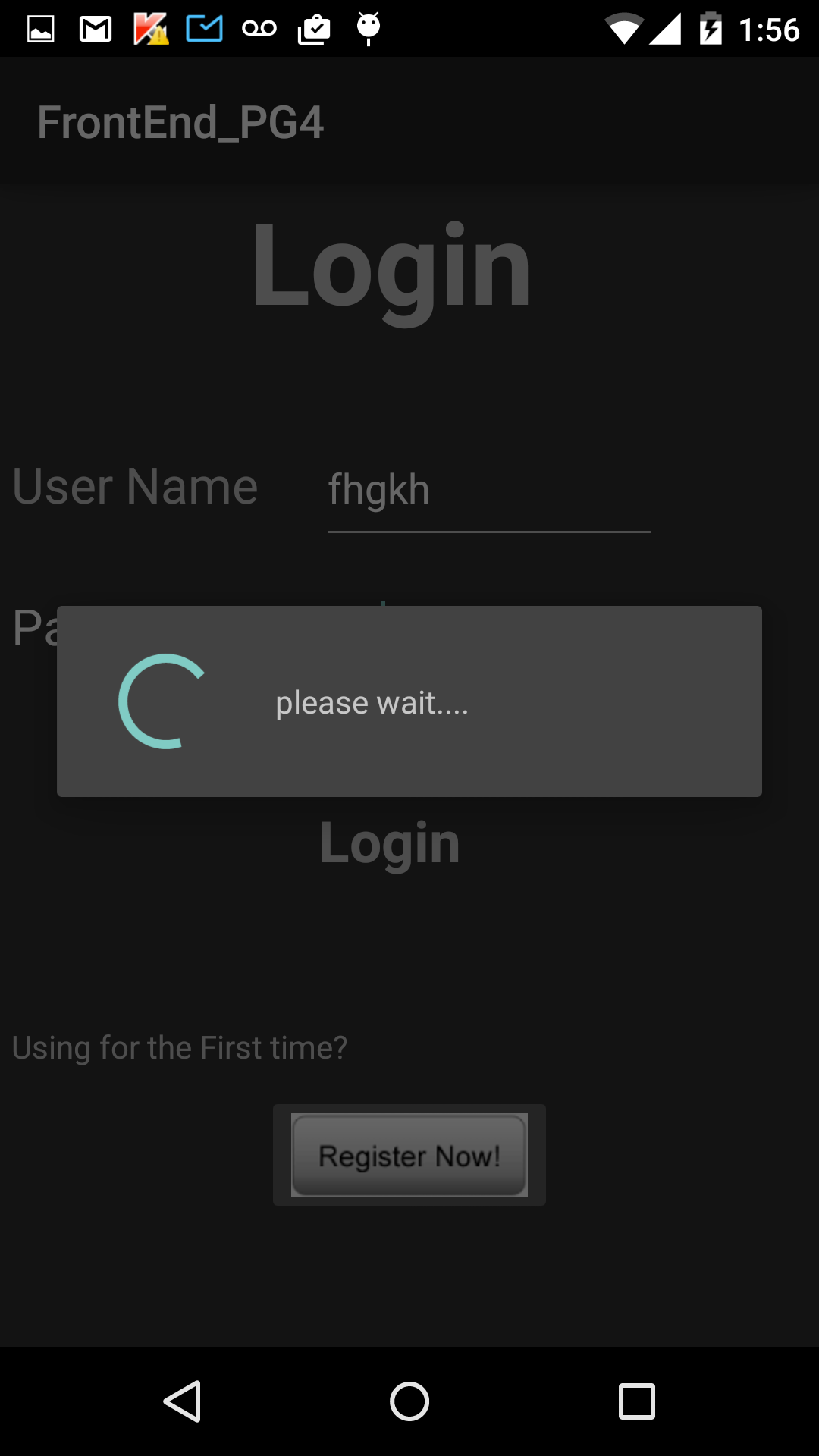
Employee sign in:

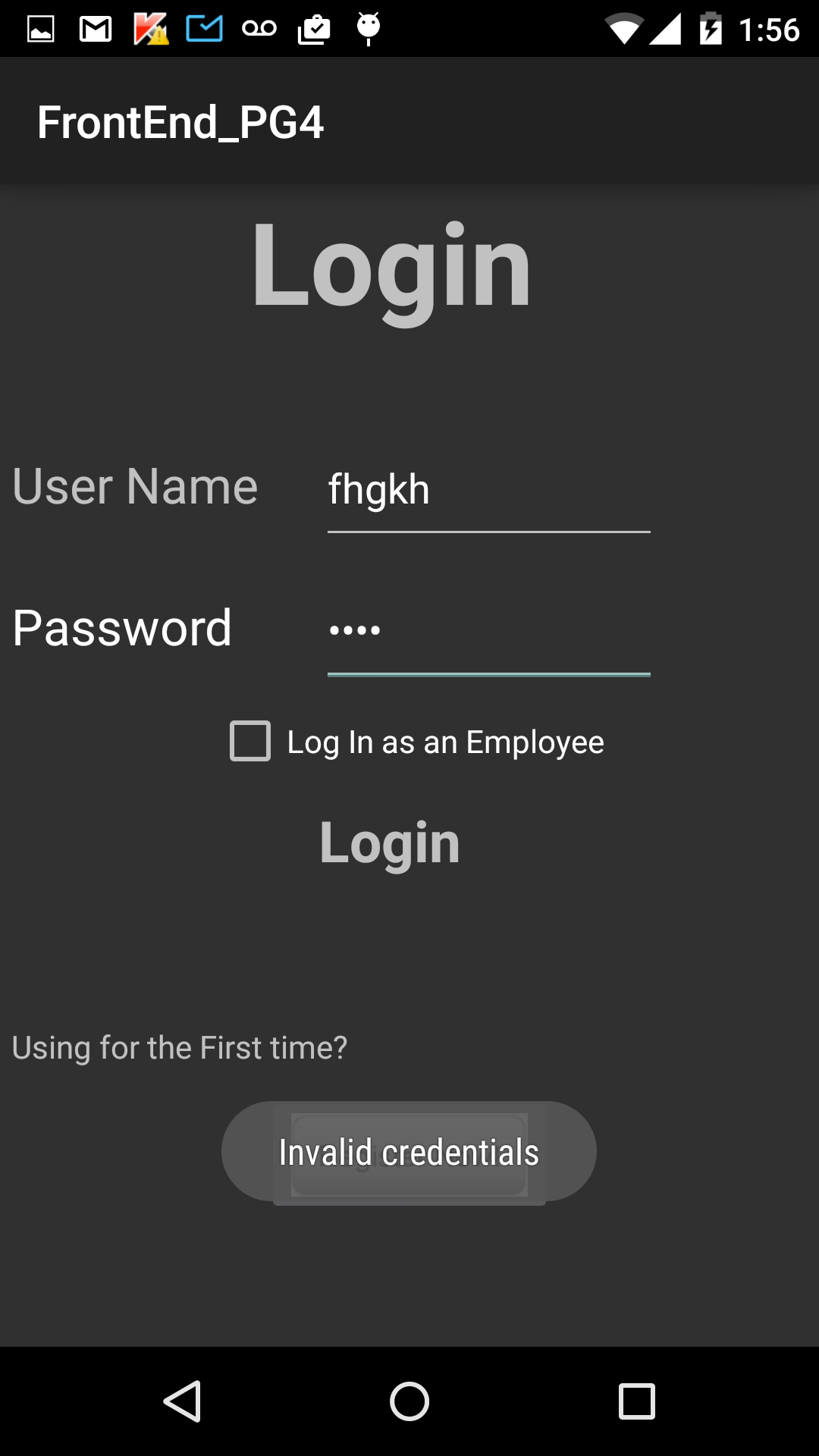


Upon successful sign in:

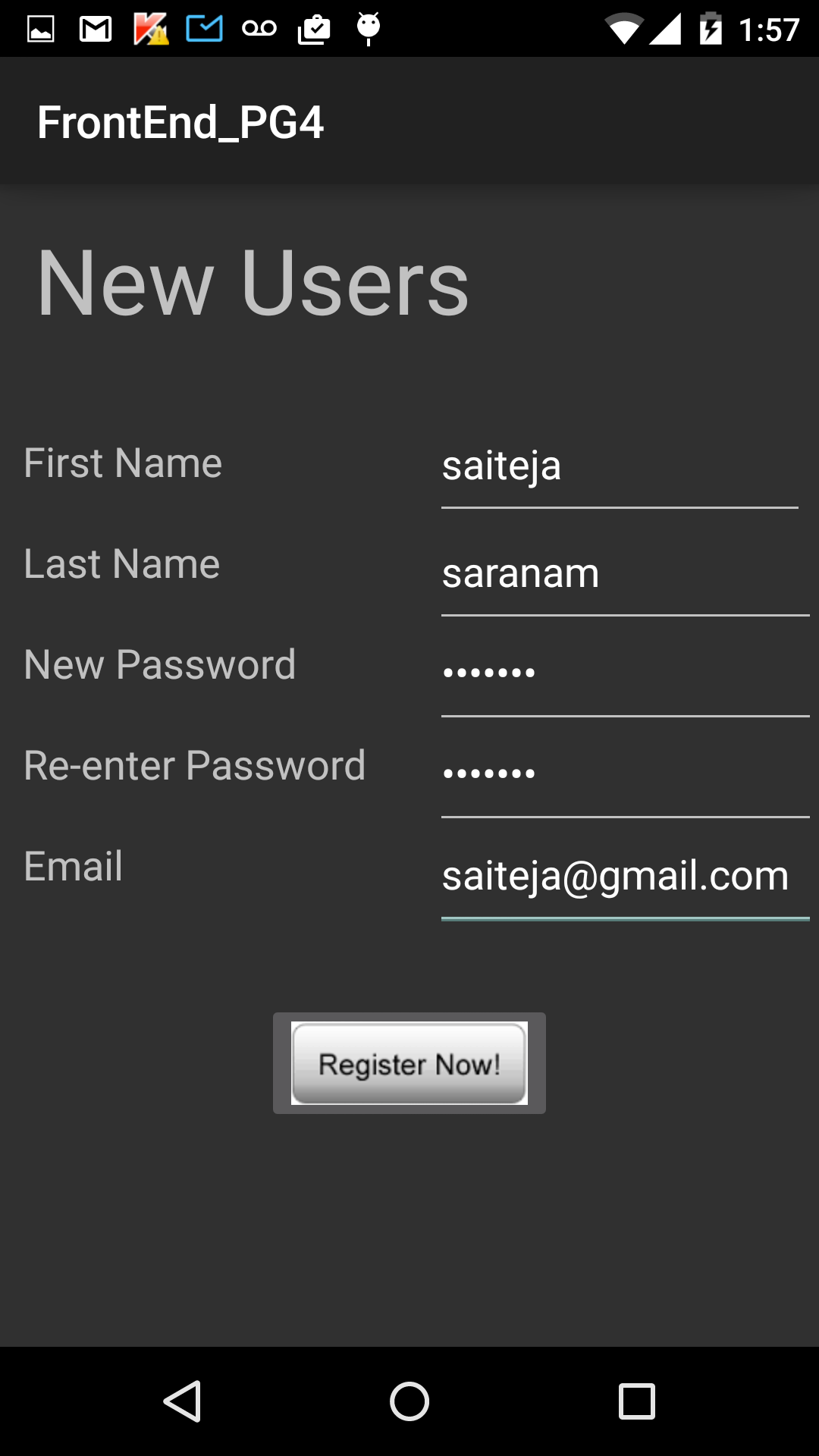


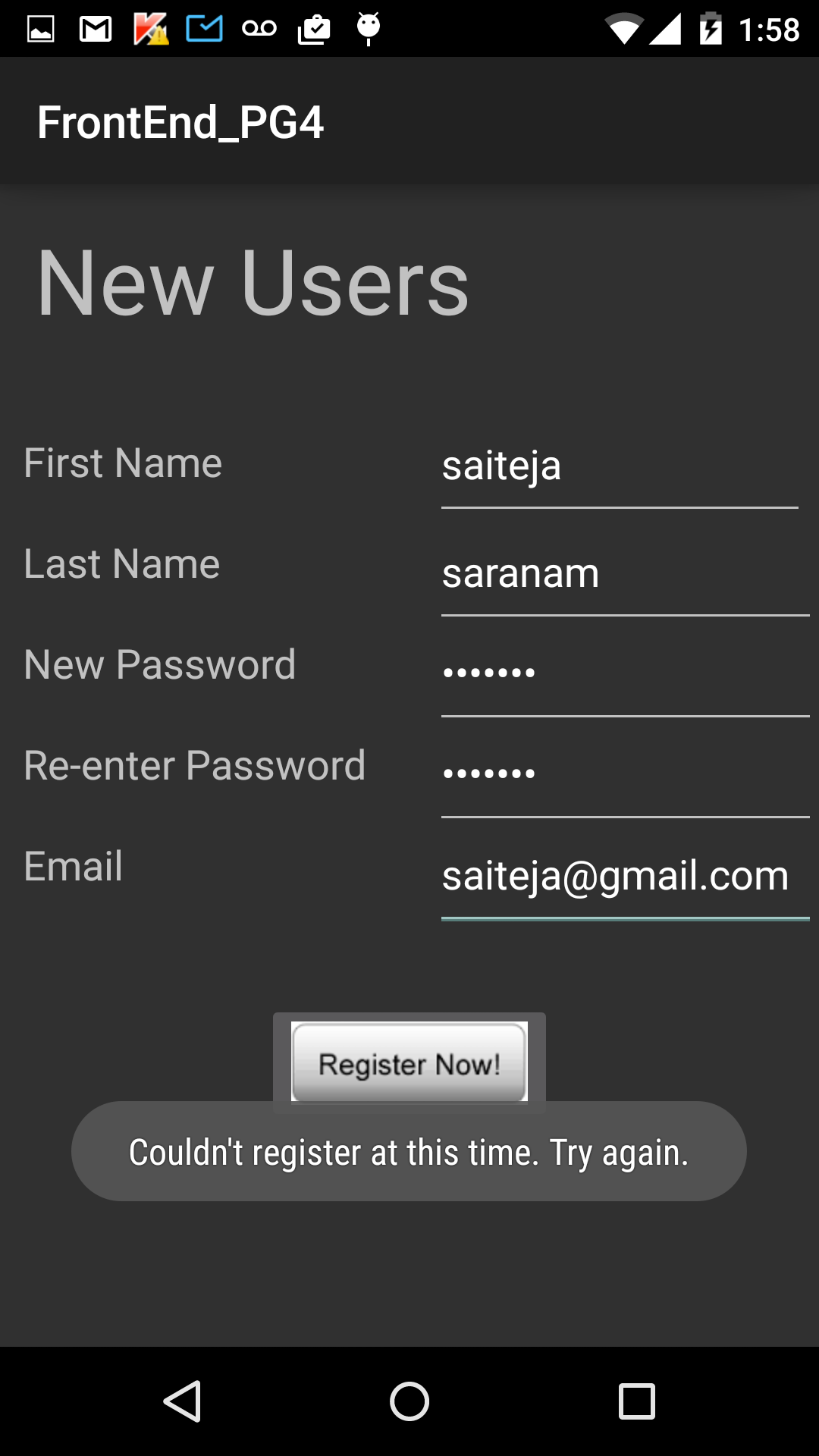
Entering Invalid credentials:





User registering for the 2nd time with same email:





**Explanation for the design and implementation of mobile client:**

We are developing the android application using IntelliJ idea. So far, we created 5 activity pages. The main activity (home page) is the Login page. A registered user can be able to login

With this page. For employees to login, we have provided a check box, so when an employee sign in, he should check that box. We assumed both admin and employee comes under employee category. So, even admin should make sure the box is checked while sign in.

For new user, we have provided a button for register. Upon taping the Register button, the register page will appear. A new user should make use of this page to register.

When a registered user tries to register for second time providing the same email, a toast will appear warning him that he cannot register again.

Using the email and password provided at the time of registering, a user will be able to sign in.

Upon successful sign in, user dashboard page will appear.

When a user enters wrong credentials, a toast will appear saying that the credentials are invalid.

All of these scenarios can be seen from the above screen shots.

For our project, only admin has the ability to register an employee. Admin registers new employees and gives them credentials. So, we haven’t provided the registration page for employee.

Since we manually stored employee details in the employee table in database, we were able to sign as an employee as shown in above screen shots.

**Explanation for implementation of backend:**

We used MySQL as database for our project. We developed and created REST services using Apache CXF and Spring dependency injection. For mapping Java classes to database tables, we used Hibernate.

Apache CXF is open source and multi featured web services framework. It takes care of all the http requests. Every request from the mobile client goes to CXF servlet.

Spring dependency injection maintains the independency of classes and at the same time gluing them together. This helps while doing unit testing and increases the possibility of reusing the classes.

**Project Management:**

**Scrumdo link:**

<https://www.scrumdo.com/projects/project/alpha8/iteration/121754/board>

**Implementation status report:**

**Work completed:**

So far we have implemented the login, signup services and implemented the REST services using Apache CXF, database connection (mapping Java classes to database tables) using Hibernate and used Jetty server plugin for testing purposes (we will deploy the backend files to server in later iteration).

|  |  |
| --- | --- |
| **Task** | **Person** |
| Functionality of mobile client | Sai Teja Saranam |
| Call rest services from android | Sai Teja Saranam |
| Layout of mobile client | Srikanth Nadendla |
| Implementation of REST services (Apache CXF) | Sai Teja Saranam, Srikanth Nadendla , Ram Nikhil |
| Hibernate (mapping Java classes to database tables) | Sai Teja Saranam, Srikanth Nadendla |
| Spring Dependency Injection | Nagender Goud, Ram Nikhil |
| Testing REST services | Nagender Goud, Ram Nikhil |
| Designing database architecture | Sai Teja Saranam, Nagender Goud |
|  |  |
| **Contributions** |  |
| **Member** | **Percentage** |
| Sai Teja Saranam | 30 |
| Srikanth Nadendla | 30 |
| Nagender Goud | 20 |
| Ram Nikhil | 20 |

|  |  |
| --- | --- |
| **Name** | **Class id** |
| Sai Teja Saranam | 29 |
| Srikanth Nadendla | 23 |
| Nagender Goud | 31 |
| Ram Nikhil | 35 |