

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology in Information Technology

By

Kushal Goyal M
REG. NO. 38120043



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

NOVEMBER 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI– 600119
www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Kushal Goyal M (Reg. No: 38120043) who carried out the project entitled “**INTEGRATED APARTMENT BUILDING MANAGEMENT SYSTEM**” under my supervision from June 2021 to November 2021.

Internal Guide

Dr. S. Gowri M.E., Ph.D.,

Head of the Department

Dr. R. Subhashini M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **Kushal Goyal M** hereby declare that the project report entitled **INTEGRATED APARTMENT BUILDING MANAGEMENT SYSTEM** done by me under the guidance of **Dr. S. Gowri Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Information Technology.

DATE: 13 / 11 / 2021

PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D, Dean**, School of Computing, **Dr. R. Subhashini, M.E., Ph.D. Head of the Department of Information Technology** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S. Gowri M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

The Integrated Apartment Building Management System (IABMS) is an Android application based system that users can use to deal with all the complicated matters involved in apartment management and communication between residents and non-residents. Traditionally, residents have to contact managers or security of apartments first to ask for services. It usually takes a long time to negotiate among residents, managers and visitors. Instead of wasting time in such a procedure, IABMS provides an efficient way so that residents can straight away authorise and pre-approve gate entries, generate security alerts, issue gatepass and message security directly from the ABMS application. Residents can also check their history of activities, payments, and online status of their visitors. Repairmen can also check their own history of appointments, contracts, and upcoming appointments.

In addition to dealing with affairs between residents and non-residents, the system provides powerful functions that not only give managers a clear picture about the status of the apartments, daily help, delivery personals, vehicle movement etc., but also gives them helpful information, including the number of times a non-resident has been in and out of the apartment building premises. This complicated information is represented both by easy charts and detailed tables, by which a manager can evaluate every daily help and maintenance team precisely and easily. All this-valuable information is stored in a highly reliable cloud database system. The key feature is that managers who are not familiar with the cloud database systems can backup and restore the database easily with by just one click.

This project is based on an app-based architecture, which provides a single point of entry to the phone system and can easily be accessed everywhere. This system is implemented by Java and XML programming languages.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	v
	LIST OF FIGURES	vii
1	INTRODUCTION	1
	1.1 Background Study	1
2.	AIM AND SCOPE OF THE PRESENT INVESTIGATION	2
	2.1 Scope	2
	2.2 Existing System	2
	2.3 Proposed System	3
	2.4 Technical Feasibility	3
3.	SOFTWARE REQUIRMENTS SPECIFICATION	5
	3.1 Hardware Components	5
	3.2 Software Components	6
4.	Design and Planning	6
	4.1 General Overview	6
	4.2 DFD Diagram	7
5	Implementation Details	8-10
	5.1.1 Angular JS	
	5.1.2 JavaScript	
	5.1.3 XML	
	5.1.4 JSON	
	5.1.5 jQuery	
6	Scope of Future Enhancement & Conclusion	11

REFERENCES	14
APPENDIX (Result and Discussions)	11
A. SCREENSHOTS	12
B. SOURCE CODE	20

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
1.	Home Page	12
2.	Settings Page	13
3.	Allow Entries Page	14
4.	Security Alert Page	15
5.	Message to Guard Page	16
6.	Activity Page	17
7.	Community Page	18
8.	Residents Page	19

1. INTRODUCTION

The Apartment Building Management System is a mobile-based intelligent security application to enhance the safety of gated communities. The system augments the existing security in an intelligent and non-intrusive manner. Some of the capabilities of ABMS that a resident can utilize are:

- Real-time domestic help management - Get instant information about presence/absence of your domestic help (e.g. Maids/Drivers/Cooks/Nannies) in the community. You will have the ability to hire domestic help from the app after reviewing their ratings/reviews.
- Digital Gate Pass If you are giving any item (cash/food items/clothes/etc.) to your domestic help you can issue digital Gate Pass through the app. This eliminates the need to sign on a piece of paper and get a validation call from the gate.
- Pre-authorization of your guest for hassle-free welcome at the gate. Guest will get an OTP in SMS that will be used at the gate to provide a seamless entry.
- Raise instant "Security Alert" for main gate guards: In case you observe any suspicious activity that is compromising safety and security of fellow residents, you can notify using "Security Alert" feature, which can be treated with utmost priority by the security staff.

1.1 BACKGROUND OF THE STUDY

For an apartment manager, it is most important to save time in dealing with all the complicated matters involved in apartment management and communication between the residents and non-residents. Computer systems and the internet have become easy and useful tools in management. As more and more complicated procedures and communication involved in apartment management develops, every apartment manager wants an easy and user-friendly solution based on the internet.

From the perspective of the manager and tenant, both of them would expect to deal with everything immediately and correctly by a reliable, easy-to-use, internet app-based apartment system management system might be needed to address these issues and problems encountered to ensure blood transfusion safety.

2. AIM

The main aim of developing this system is to ensure that every person, package or vehicle entering the community is authorized by a resident. The idea is simple: IABMS digitizes the traditional intercom and public addressing system experience and puts it all into one easy-to-use app at a fraction of the cost of installing or maintaining an intercom system.

2.1 SCOPE

There has been a surge in the popularity of society-management apps in India. They are becoming the channel for societies and apartment complexes to tap essential services while minimizing interactions with the outside world. The stage is set for a clash of the apps for market leadership.

2.2 EXISTING SYSTEM

Most of the existing apartment management software only focuses on the payment and accounting system, which means the manager still has to use lots of energy in communication between residents and non-residents. The Maintenance and troubleshooting with the existing intercom system is cumbersome as much as installation of a wireless intercom system is easy, the maintenance is not and is not so flexible as all the procedures are done manually. This increases the paperwork of the office which includes the bill payments also there arises confusion among doing various maintenances The need for a complete apartment management packages led to the design of the internet based management system, providing a comprehensive tool to residents and apartment managers.

2.3 PROPOSED SYSTEM

The proposed system is fully computerized, It does absolutely nothing on paper. As all the entire working is done with the aid of computers, this will result in faster processing and improved performance. This information can be accessed quickly by the click of a button when compared to earlier retrieval from hardcopies of files and registers as all the details stored in the cloud. Compared to the stand alone static application, this system is app-based and so can be accessed through whole internet. It is indubitable that security is an important issue in this project. Without solid secure protection, someone could access the database and retrieve confidential and sensitive information such as personal profiles. According to the goals of the project, the app-based apartment management system will allow residents access and manage their own information about visitors, daily help, deliveries and the history of past activities anytime and anywhere, just by few clicks. Even to make, modify, cancel gate entries, residents can also finish it by just few clicks without long term of negotiation. To managers, they can save precious time on communication between residents and non-residents. Furthermore, by knowing what kinds of problems frequently happened during particular time periods, they can act on and deal with the situation raised accordingly. Residents and Managers also can evaluate every daily help of the community by information of overall ranking and their attendance history.

2.4 TECHNICAL FEASIBILITY

The apartment management system supports the economic feasibility to a great extends. Development of the system and the cost of hardware and software are not high. This reduces effort and time of us. This makes software economically feasible as well. The performance of this software is more accurate, more user friendly, effective and error free.

3 .SYSTEM CONFIGURATION

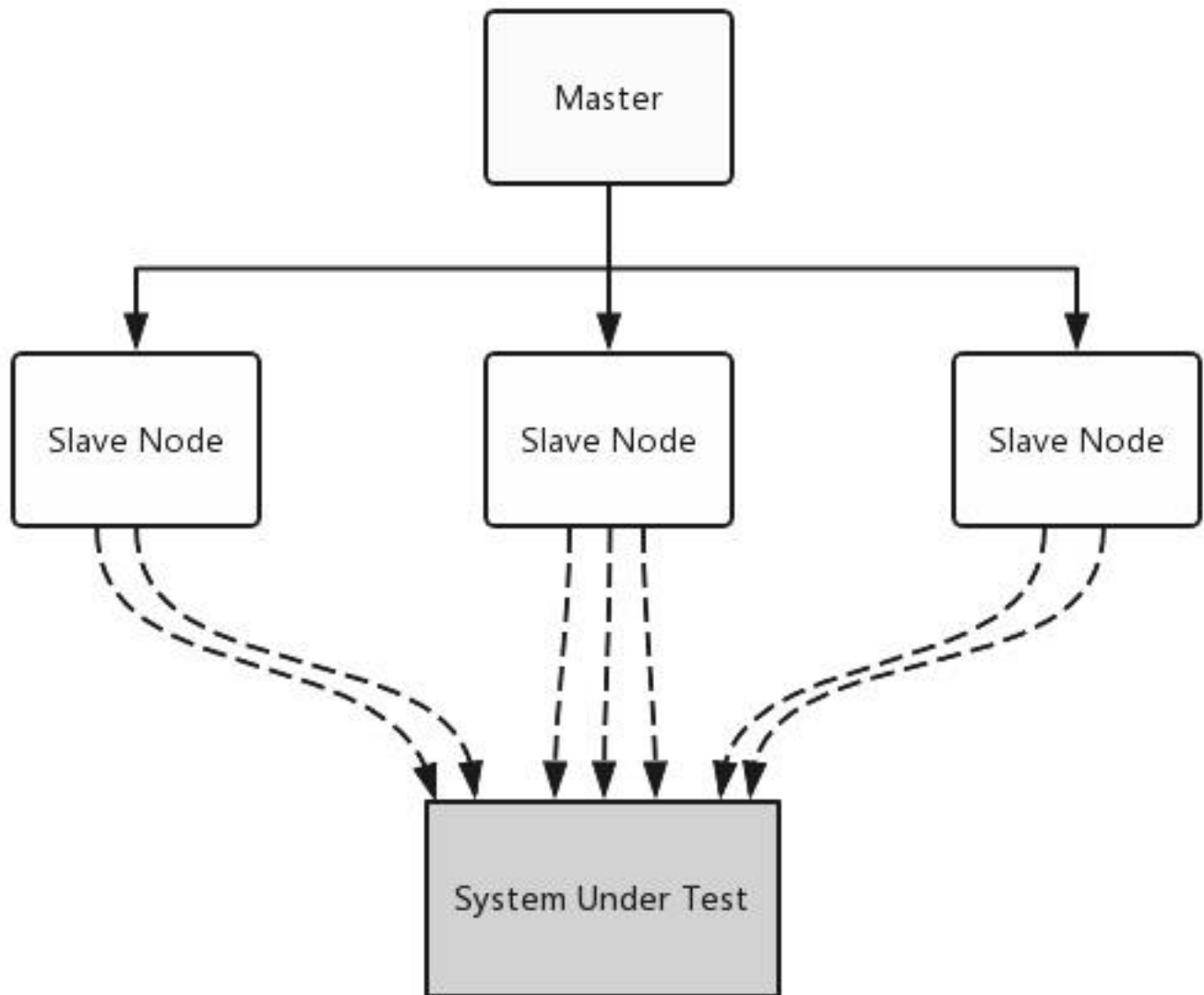
3.1 HARDWARE COMPONENTS

Number	Description
1. Processor	2 nd Generation Intel Core or newer
2. Storage	8 GB RAM or more
3. Monitor	1280 x 800 minimum screen resolution

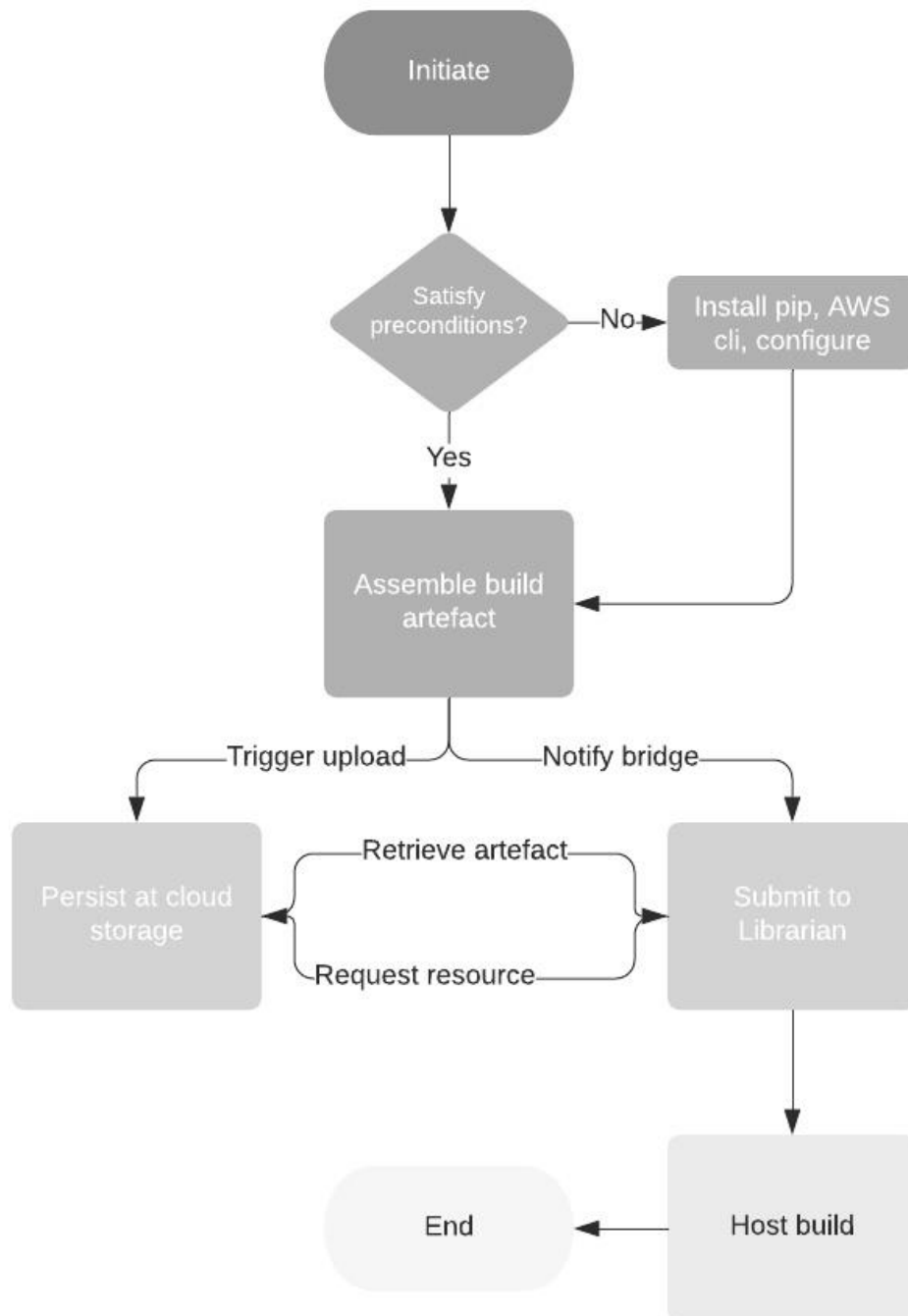
3.2 SOFTWARE COMPONENTS

Number	Description
1. Programming Software	Android Studio IDE and SDK Tools
2. Programming Language	AngularJS, JavaScript, XML, jQuery, JSON
3. Back-End	GoDaddy Cloud Hosting / Amazon Cloud Service

4. GENERAL OVERVIEW:



DFD DIAGRAM



CHAPTER 5 : IMPLEMENTATION DETAILS

In this Section we will do Analysis of Technologies to use for implementing the project.

5.1.1 AngularJS

AngularJS is a JavaScript-based open-source front-end web framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–viewmodel (MVVM) architectures, along with components commonly used in rich Internet applications. AngularJS is the frontend part of the MEAN stack, consisting of MongoDB database, Express.js web application server framework, Angular.js itself, and Node.js server runtime environment.

The AngularJS framework works by first reading the Hypertext Markup Language (HTML) page, which has additional custom HTML attributes embedded into it. Angular interprets those attributes as directives to bind input or output parts of the page to a model that is represented by standard JavaScript variables. The values of those JavaScript variables can be manually set within the code, or retrieved from static or dynamic JSON resources. AngularJS is built on the belief that declarative programming should be used to create user interfaces and connect software components, while imperative programming is better suited to defining an application's business logic.

5.1.2 JavaScript

JavaScript often abbreviated as JS, is a programming language that conforms to the ECMAScript specification.[11] JavaScript is high-level, often just-in-time compiled and multi-paradigm. It has dynamic typing, prototype-based object-orientation and first-class functions. Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. Over 97% of websites use it client-side for web page behavior, often incorporating third-party libraries.

All major web browsers have a dedicated JavaScript engine to execute the code on the user's device. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

The ECMAScript standard does not include any input/output (I/O), such as networking, storage, or graphics facilities. In practice, the web browser or other runtime system provides JavaScript APIs for I/O. JavaScript engines were originally used only in web browsers, but they are now core components of some servers and a variety of applications. The most popular runtime system for this usage is Node.js. Although there are similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

The JavaScript code is executed when the user submits the form, and only if all the entries are valid, they would be submitted to the Web Server. JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly.

5.1.3 XML

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification of 1998 and several other related specification—all of them free open standards—define XML. The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services. Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data.

A programming language consists of grammar rules and its own vocabulary which is used to create computer programs. These programs instruct the computer to perform specific tasks. XML does not qualify to be a programming language as it does not perform any computation or algorithms. It is usually stored in a simple text file and is processed by special software that is capable of interpreting XML.

Because XML brings sophisticated data coding to Web sites, it helps companies integrate their information flows. By creating a single set of XML tags for all corporate data, information can be shared seamlessly among Web sites, databases, and other back-end systems. But the revolutionary power of XML lies in supporting transactions between businesses. When a company sells a good or service to another company, a great deal of information needs to be exchanged—about prices, terms, specifications, delivery schedules, and so on. HTML's one-size-fits-all nature makes such exchanges difficult, if not impossible, over the Internet. With XML, all the necessary information can be shared electronically, allowing complex deals to be closed without any human intervention.

5.1.4 JSON

JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax. It is commonly used for transmitting data in web applications (e.g., sending some data from the server to the client, so it can be displayed on a web page, or vice versa). You'll come across it quite often, so in this article we give you all you need to work with JSON using JavaScript, including parsing JSON so you can access data within it, and creating JSON.

To obtain the JSON, we use an API called XMLHttpRequest (often called XHR). This is a very useful JavaScript object that allows us to make network requests to retrieve resources from a server via JavaScript (e.g. images, text, JSON, even HTML snippets), meaning that we can update small sections of content without having to reload the entire page.

5.1.5 jQuery:

jQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax. It is free, open-source software using the permissive MIT License. As of May 2019, jQuery is used by 73% of the 10 million most popular websites. Web analysis indicates that it is the most widely deployed JavaScript library by a large margin, having at least 3 to 4 times more usage than any other JavaScript library. jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, the meuble widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and Web applications.

CHAPTER 6: SCOPE OF FUTURE ENHANCEMENT & CONCLUSIONS

Scope of Future Enhancement

For further maintenance, applying standard of JAVA Module 2 will make it easier to maintain the system. There are two major reasons. The first reason is that the system is modulated and less coupling among all components. The second reason is that because business logic is extracted from the code and page so that it is convenient to add or modify functions without modifying the whole page which performs the web page to users. For example, the results of all inquiry are presented as tables to be showed on the page. When we need to change inquiry, we might have to rewrite and redesign the appearance of the page. Therefore, I would like to change the design of this project to JAVA Module 2.

Conclusion

The system objectives specified in the requirement documents are believed to have been satisfactory met. The system gives good result in the testing. The processing of the proposed system is simple and is in regular order. The various plans employed in the project are user friendly and easy to understand. The system has been designed in manner so as to provide valuable, timely and accurate information. The system was effective and efficient in the estimate generation process. Any native user can understand the problem, solve it and generate reports through the system.

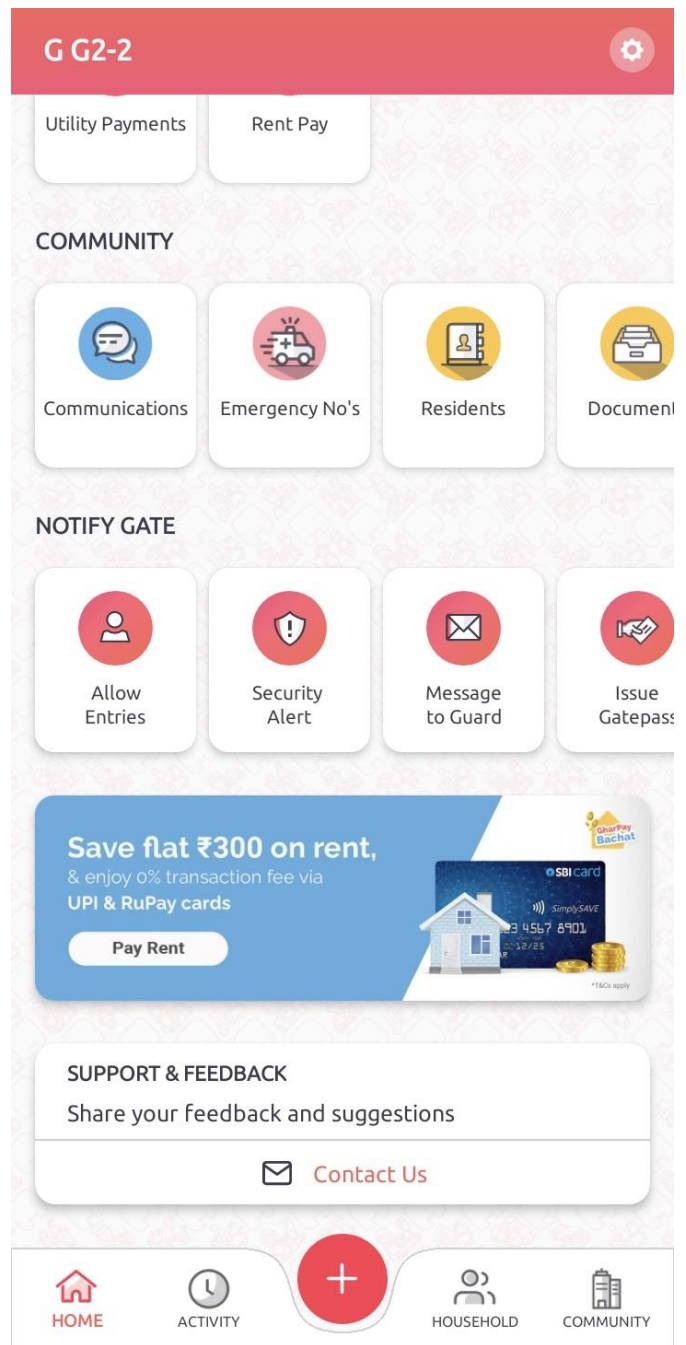
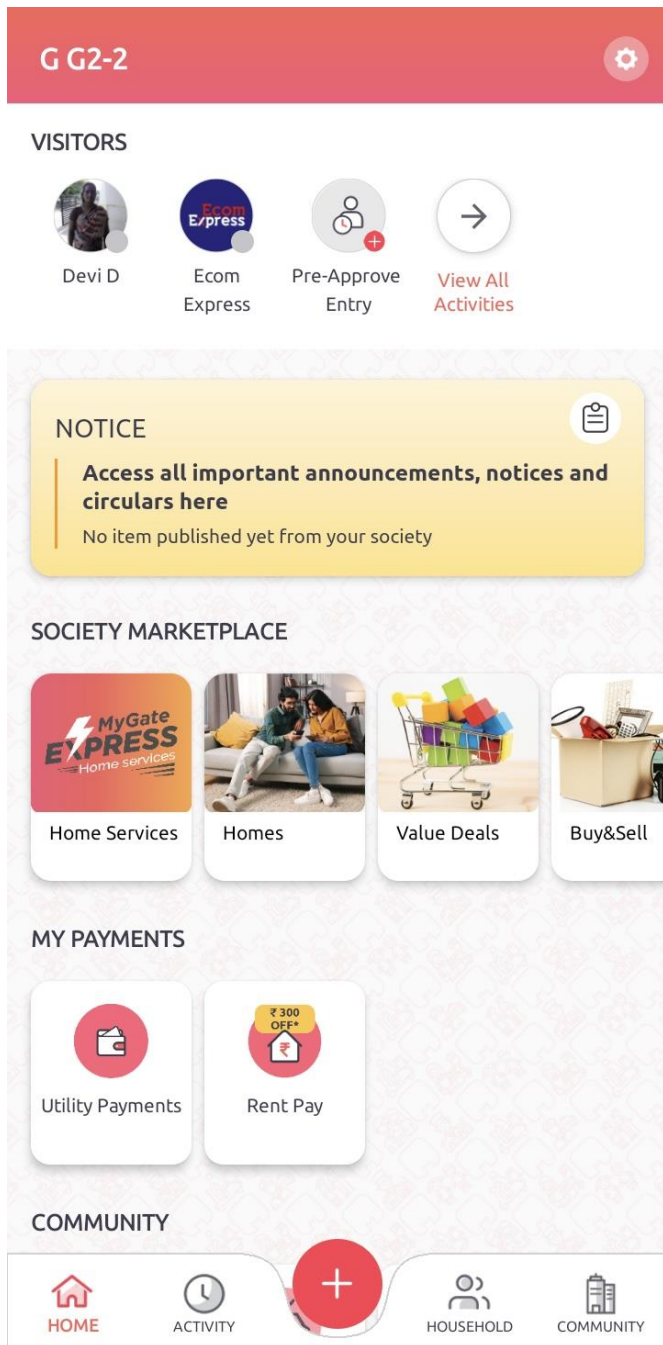
References

1. "Professional JavaScript for App Developers" Nicholas C. Zakas.
2. "Android Programming with Kotlin for Beginners" by John Hort.

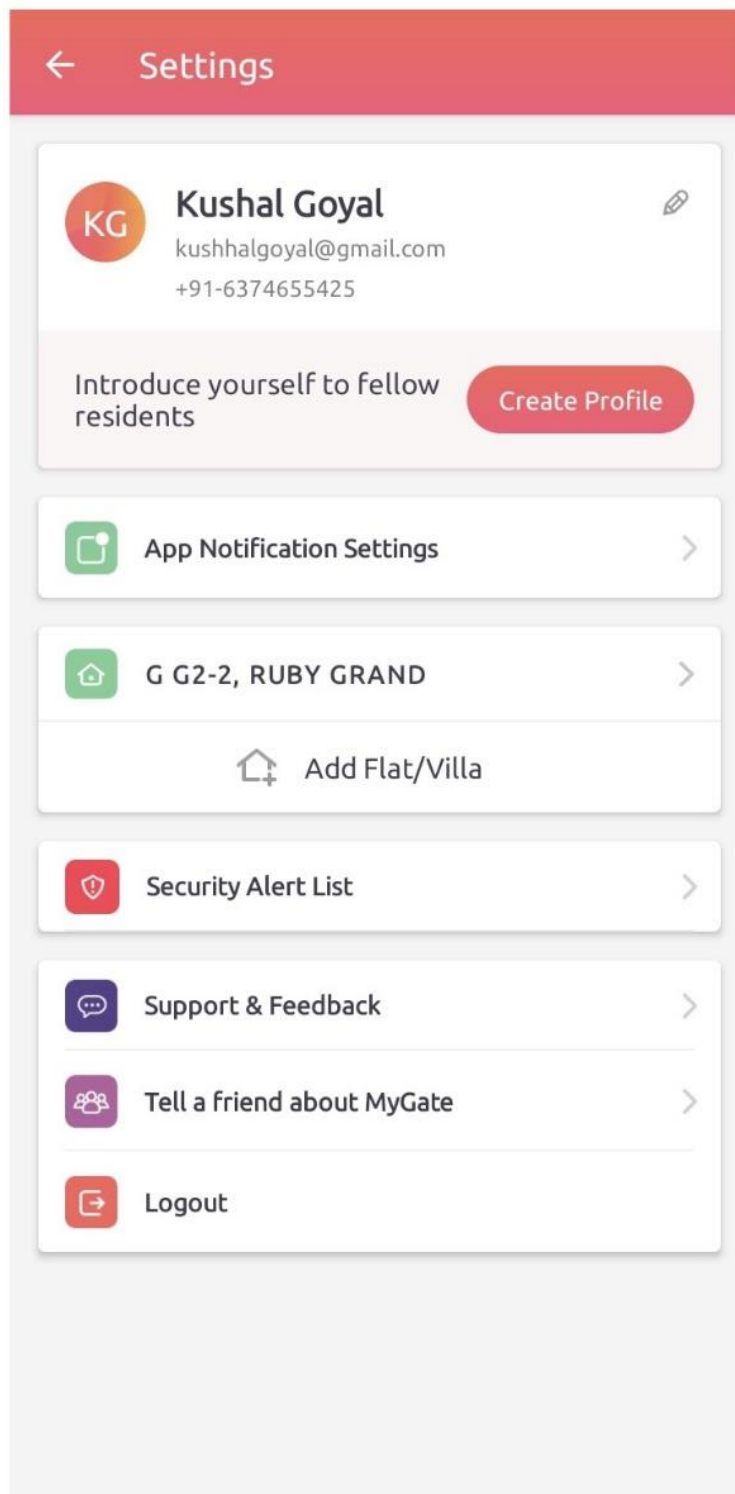
APPENDIX:

Screenshots :

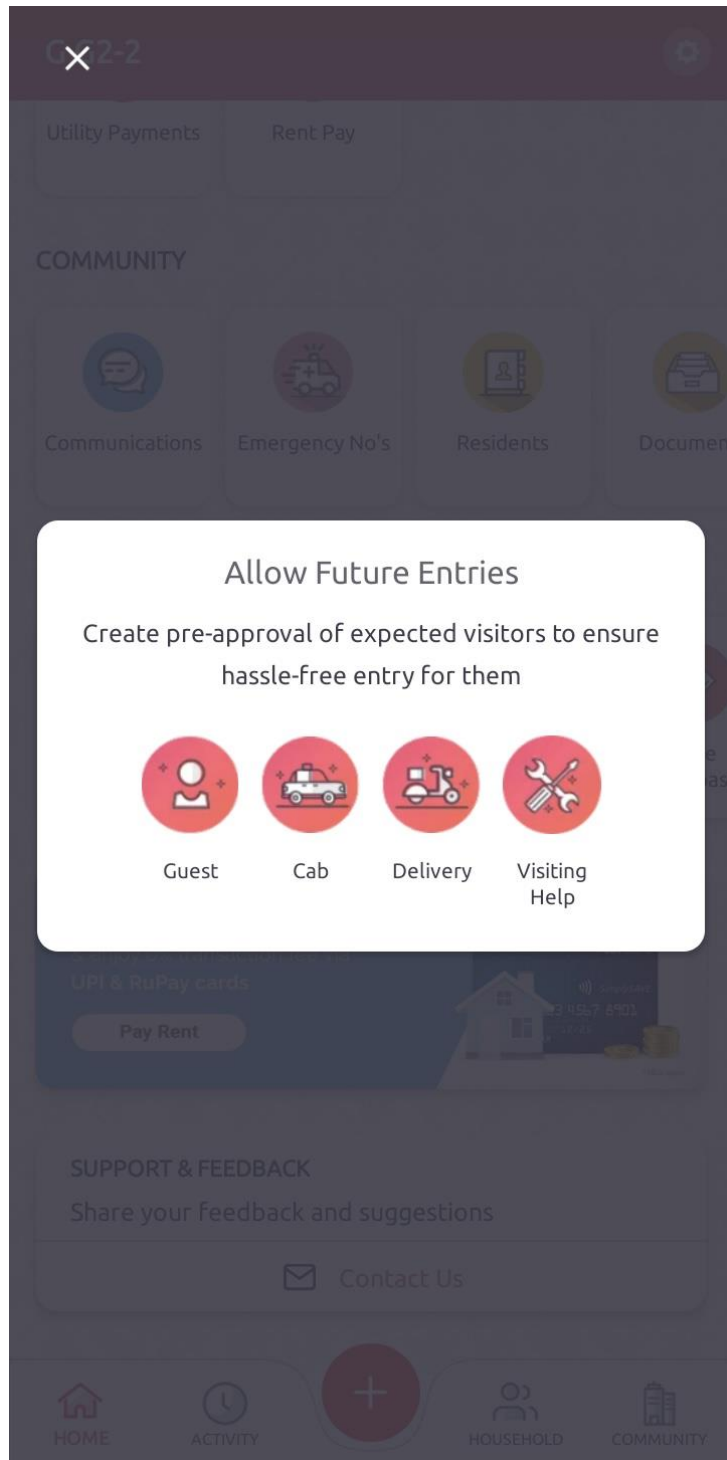
Home Page :



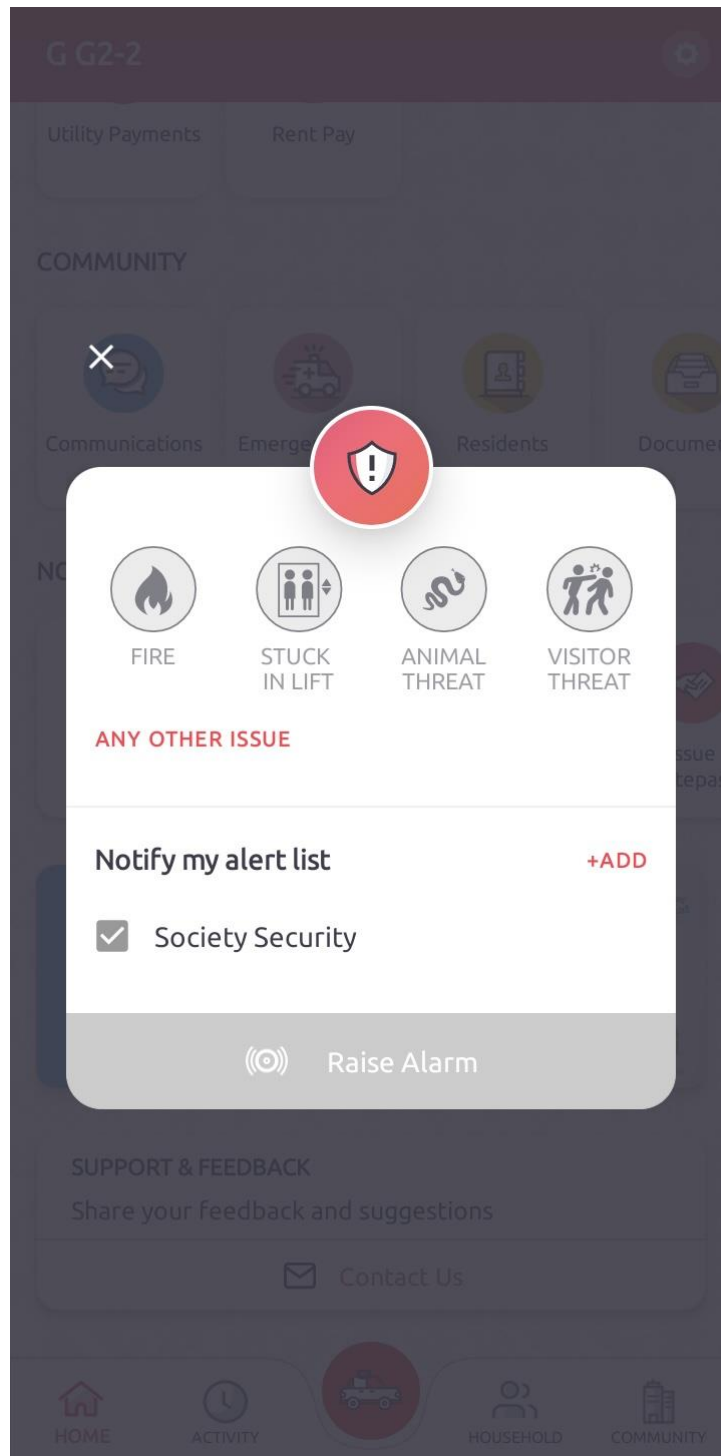
Settings Page :



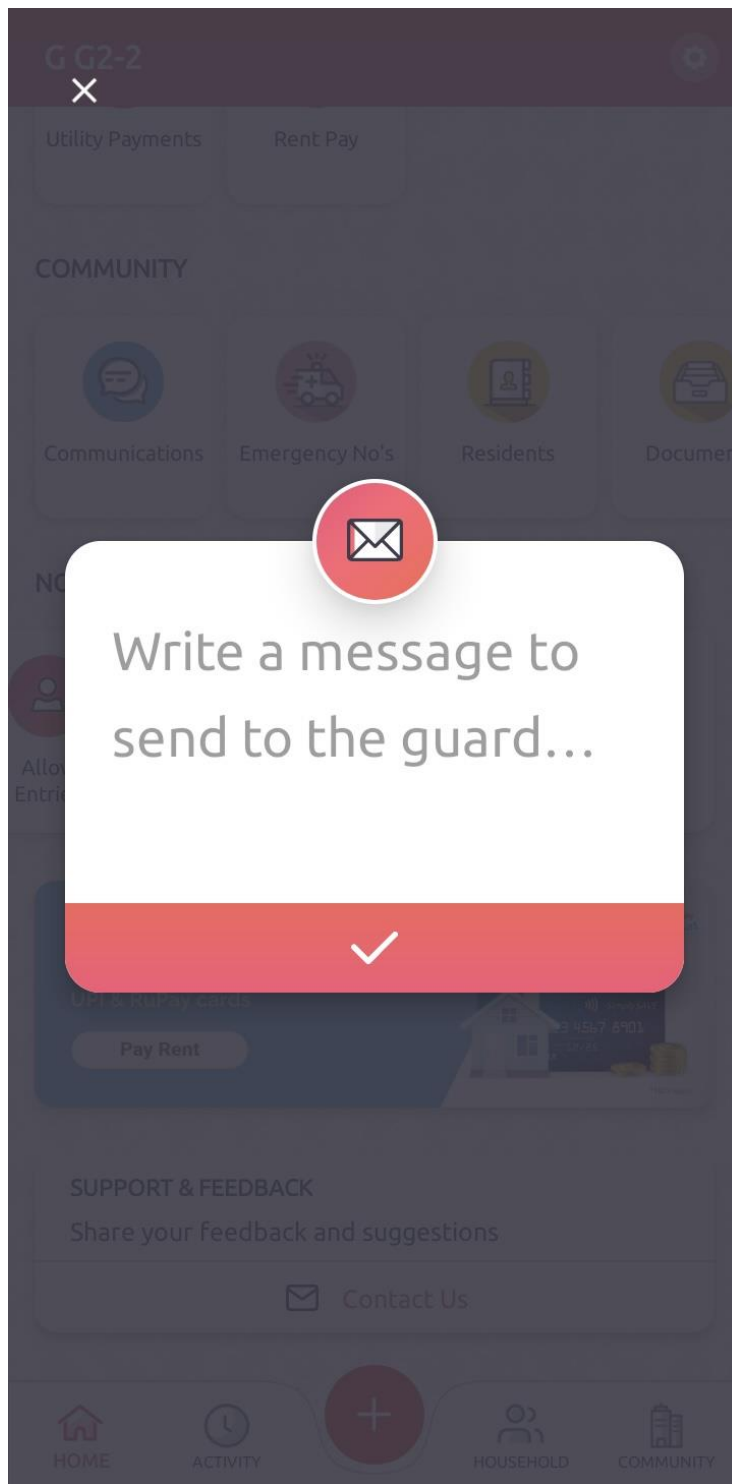
Allow Entries Page :



Security Alert Page :



Message to Guard Page :



Activity Page :

G G2-2

 FILTER  DATE 

 **Devi D**

LEFT  03:40 PM • MAID

  Rate Now  Attendance

 **Delivery**

LEFT  10:51 AM

 Prem  Ecom Express
 Allowed by you

  Wrong entry

Yesterday

 **Devi D**

LEFT  04:10 PM • MAID

  Rate Now  Attendance

03 Nov

 **Devi D**

ADDED  10:34 AM • MAID

 HOME

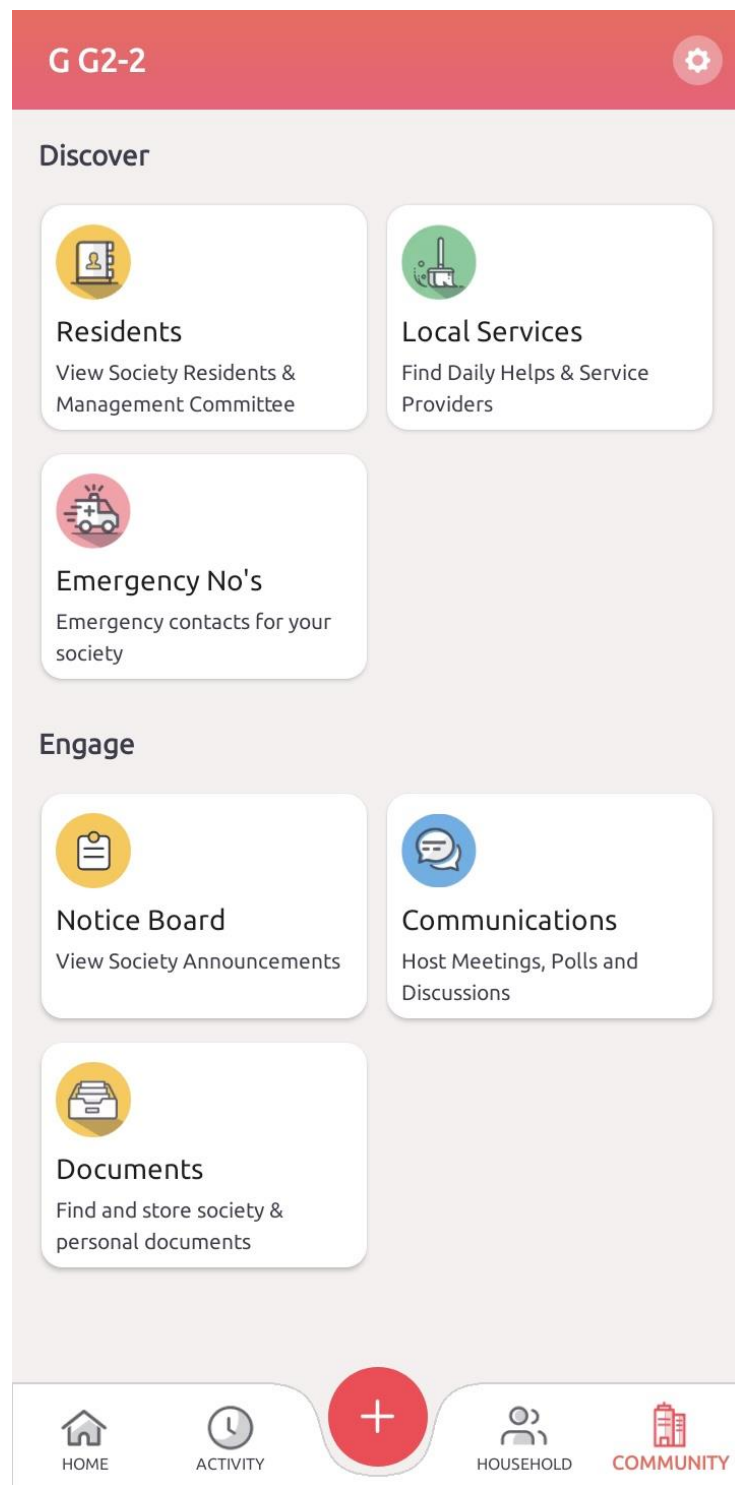
 **ACTIVITY**



 HOUSEHOLD

 COMMUNITY

Community Page:



Residents Page :

Residents of Ruby Grand		
	Management Committee	>
BUILDINGS		
	A	>
	B	>
	C	>
	D	>
	E	>
	F	>
	G	>
	H	>
	I	>
	J	>
	K	>
	L	>
	M	>

SOURCE CODE :

ExampleInstrumentTest.java:

```
package com.example.roshan.apartmentdemo;

import android.content.Context;
import android.support.test.InstrumentationRegistry;
import android.support.test.runner.AndroidJUnit4;

import org.junit.Test;
import org.junit.runner.RunWith;

import static org.junit.Assert.*;

/**
 * Instrumented test, which will execute on an Android device.
 *
 * @see <a href="http://d.android.com/tools/testing">Testing documentation</a>
 */
@RunWith(AndroidJUnit4.class)
public class ExampleInstrumentedTest {
    @Test
    public void useAppContext() {
        // Context of the app under test.
        Context appContext = InstrumentationRegistry.getTargetContext();

        assertEquals("com.example.roshan.apartmentdemo", appContext.getPackageName());
    }
}
```

Home :

```
package com.example.ruby.granddemo.Activity;

import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Toast;

import com.example.ruby.granddemo.Database.QueryUtility;
import com.example.ruby.granddemo.R;

public class MainActivity extends AppCompatActivity {

    private class GetDatabaseTask extends AsyncTask<Void, Void, QueryUtility> {

        @Override
        protected QueryUtility doInBackground(Void... voids) {
            return QueryUtility.getInstance(getApplicationContext());
        }
    }
}
```

```

@Override
protected void onPostExecute(QueryUtility queryUtility) {
    myQuery = queryUtility;
    Toast.makeText(MainActivity.this, "Successfully connected to DB", Toast.LENGTH_SHORT).show();
    //Toast.makeText(MainActivity.this, queryUtility.getSession(), Toast.LENGTH_SHORT).show();
    autoSignIn();
}
}

QueryUtility myQuery;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    GetDatabaseTask getDatabaseTask = new GetDatabaseTask();
    getDatabaseTask.execute();
    getSupportActionBar().hide();
}

public void startTenantLoginActivity(View view) {
    startActivity(new Intent(this, TenantLoginActivity.class));
}

public void startOwnerLoginActivity(View view) {
    startActivity(new Intent(this, OwnerLoginActivity.class));
}

public void autoSignIn() {
    if(myQuery.getSession() == null) {
        return;
    }
    if(myQuery.getSession().equals("owner")) {
        startActivity(new Intent(this, OwnerDashboardActivity.class));
        finish();
    }
}
}

```

Resident :

```

package com.example.roshan.apartmentdemo.Activity;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ListView;
import android.widget.Toast;

import com.example.roshan.apartmentdemo.Helper.CustomAdapter;

```

```

import com.example.roshan.apartmentdemo.R;

public class flatlist extends AppCompatActivity {
    ListView list;
    String[] itemname ={
        "A Block",
        "B Block ",
        "C Block ",
        "D Block ",
        "E Block "
    };
    String[] flatcity ={
        "Chennai",
        "Chennai",
        "Chennai",
        "Chennai",
        "Chennai"
    };

    Integer[] imgid={
        R.drawable.person,
        R.drawable.pn,
        R.drawable.hai,
        R.drawable.man,
        R.drawable.model
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_flatlist);
        getSupportActionBar().setTitle("Your Flat");
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        getSupportActionBar().setDisplayShowHomeEnabled(true);
        CustomAdapter adapter=new CustomAdapter(this, itemname, imgid,flatcity);
        list=(ListView)findViewById(R.id.list);
        list.setAdapter(adapter);

        list.setOnItemClickListener(new AdapterView.OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> parent, View view,
                                   int position, long id) {
                String Slecteditem= itemname[+position];
                Toast.makeText(getApplicationContext(), Slecteditem, Toast.LENGTH_SHORT).show();

            }
        });
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (item.getItemId() == android.R.id.home) {
            finish();
        }
        return super.onOptionsItemSelected(item);
    }
}

```

Home_layout.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView android:layout_width="match_parent"
android:layout_height="match_parent"
xmlns:android="http://schemas.android.com/apk/res/android">

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/primaryDarkColor"
        android:orientation="vertical">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="16dp"
            android:text="Apartment Demo App"
            android:textAppearance="@style/TextAppearance.AppCompat.Title"
            android:textColor="#FFF" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="16dp"
            android:layout_marginRight="16dp"
            android:layout_marginBottom="16dp"
            android:alpha="0.5"
            android:text="@string/developed"
            android:textColor="#FFF" />
    </LinearLayout>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Welcome"
        android:textAlignment="center"
        android:padding="8dp"
        android:layout_marginTop="16dp"
        android:textAppearance="@style/TextAppearance.AppCompat.Headline" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAlignment="center"
        android:text="Let's get you started"
        android:layout_marginBottom="16dp"
        android:textAppearance="@style/TextAppearance.AppCompat.Medium" />

    <Button
```

```

style="@style/Widget.AppCompat.Button.Colored"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginLeft="16dp"
android:layout_marginRight="16dp"
android:layout_marginTop="16dp"
android:onClick="startTenantLoginActivity"
android:text="I'm a Tenant" />

<Button
    style="@style/Widget.AppCompat.Button.Colored"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="16dp"
    android:layout_marginRight="16dp"
    android:layout_marginTop="16dp"
    android:onClick="startOwnerLoginActivity"
    android:text="I'm an owner" />
</LinearLayout>

</ScrollView>

```

AndroidManifest.xml :

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.roshan.apartmentdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".Activity.MainActivity"
            android:theme="@style/AppTheme">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".Activity.TenantLoginActivity" />
        <activity android:name=".Activity.OwnerLoginActivity" />
        <activity android:name=".Activity.OwnerDashboardActivity" />
        <activity android:name=".Activity.flatlist" />
        <activity android:name=".Activity.tenantlist" />
        <activity android:name=".Activity.EditTenantDetails">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.roshan.apartmentdemo.Activity.OwnerDashboardActivity" />
        </activity>
        <activity android:name=".Activity.EditFlatDetails">
            <meta-data

```

```

        android:name="android.support.PARENT_ACTIVITY"
        android:value="com.example.roshan.apartmentdemo.Activity.OwnerDashboardActivity" />
    </activity>
    <activity android:name=".Activity.TenantView" />
    <activity android:name=".Activity.FlatView"></activity>
</application>

</manifest>

```

build.gradle :

```
apply plugin: 'com.android.application'
```

```

android {
    compileSdkVersion 28
    defaultConfig {
        applicationId "com.example.roshan.apartmentdemo"
        minSdkVersion 21
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(include: ['*.jar'], dir: 'libs')
    implementation 'com.android.support.constraint:constraint-layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    implementation 'com.android.support:cardview-v7:27.1.0'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
    implementation 'com.android.support:appcompat-v7:27.1.0'
    implementation 'com.android.support:design:27.1.0'
}

```