###### VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM- 590014



**A Mini Project Report**

**On**

**Smart Home Security**

Submitted in partial fulfillment of the requirement for the degree of,

Bachelor of Engineering in Computer Science and Engineering

**Submitted By,**

Suman Salimath - 2bv14cs113

Tejaswini - 2bv14cs122

Venuprasad Naik - 2bv14cs124

**Under the guidance of,**

Padmashree Desai

Kavitha H.S

Parikshit P Hegde



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.V.B COLLEGE OF ENGINEERING AND TECHNOLOGY,

HUBLI – 580 031 (India)

(**An Autonomous Institution affiliated to VTU, Belgaum**)

Academic year 2016-17

K.L.E. SOCIETY’S

**B.V. BHOOMARADDI COLLEGE OF ENGINEERING & TECHNOLOGY, HUBLI -580031**

**(An Autonomous Institution affiliated to VTU, Belgaum)**

2016 – 2017



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**Certificate**

This is to certify that Mini Project entitled Smart Home Security is a bonafied work carried out by the student team Ms. Suman Salimath – 2bv14cs113, Ms. Tejaswini - 2bv14cs122, Mr. Venuprasad Naik – 2bv14cs124, in partial fulfillment of completion of Fifth semester B. E. in Computer science and Engineering during the year 2015 – 2016. The project report has been approved as it satisfies the academic requirement with respect to the project work prescribed for the above said programme.

**Guide H.O.D Principal**

**Padmashree Desai Dr. G. H. Joshi Dr. P.G. Tewari.**

**Kavitha H.S**

**Parikshit P Hegde**

**External Viva:**

**Name of the Examiners Signature with date**

**1.**

**2.**

**ABSTRACT**

Our project basically deals with the controlling of the home appliances with the android app and the security one can provide to its residents. The owner of the house can monitor the log history as to who locked and unlocked the door along with the timings. The overall idea is to make ones job easier and to reduce energy wastage as of forgetting the turning off of appliances which in turn lead to energy loss.

**ACKNOWLEDGEMENTS**

We pay our sincere gratitude to our principal Dr. P.G Tewari for giving opportunity to work by providing necessary facilities.

We thank our beloved H.O.D Dr. G.H Joshi, Department of Computer Science, BVB college of Engineering and Technology for his enthusiastic support.

We extend our gratitude to our guide Prof. Padmashree Desai for her valuable guidance throughout.

We also thank Prof. Kavitha .H.S and Prof. Parikshit .Hegde for their support.

We are also thankful to our parents and friends for their encouragement and support from beginning till the end. We also thank each and everyone who are directly or indirectly helpful for successful completion of the project.

Team Members:

Suman Salimath - 2bv14cs113

Tejaswini - 2bv14cs122

Venuprasad Naik - 2bv14cs124

**Chapter Table of Contents Page**

**No. No.**

1. **Preamble** 
   1. Introduction 7
   2. Problem definition 7
   3. Motivation 7
   4. Scope and Objectives of the project 7
   5. Explanation of proposed system with simple block diagram 8
2. **Literature Survey**
   1. Existing System(s) (Advantages and Limitations of existing 9 System)
   2. Description of Target users 9
   3. Advantages/applications of proposed system 9
   4. Constraints or limitations of proposed system 9
3. **Software Requirement Specification**
   1. Overview of SRS 10
   2. Intended audience 10
   3. Requirement Specifications
      1. Functional Requirements 10
      2. Use case diagrams 11

Use case descriptions using scenarios, strictly as per Pressman Template

* + 1. Nonfunctional Requirements
       1. Performance requirements 13
       2. Security requirements 13
       3. Usability 13
       4. Response time 13
  1. Software and Hardware requirement specifications 13
  2. GUI of proposed system (snap shots) 14
  3. Acceptance test plan 15

1. **System Design**
   1. Architecture of the system (explanation) 18
   2. Level 0 DFD (with brief explanation) 18
   3. Detailed DFD for the proposed system 19
   4. Class Diagram (with brief explanation) 21
   5. Class coupling matrix and explanation 24
2. **Implementation**
   1. Implementation 25
   2. Brief description of modules which are implemented in pair 25
   3. Modules descriptions 26
3. **Testing**
   1. Introduction 28
   2. Module test plan (unit testing of each module) and test cases 28
   3. JUNIT testing 28
   4. Usability testing 30
4. **Results** 31
5. **Summary of learnings** 32
6. **Conclusion and future scope** 32
7. **References/Bibliography**  32
8. **Appendix**

**11.1** Explanation on JUNIT tool 33

1. **Preamble**
   1. **Introduction**

Nowadays providing security to house is one of the difficult task, and should be taken care that no unauthorized person will be able to lock or unlock the doors of the house, and also if one has forgotten to switch off the lights or fans of the house by mistake, there must be an option for him to somehow control the power supply, which leads to saving the electricity. Hence there is a need of smart app which allows one to control the fans lights and also doors from the locations they are in.

* 1. **Problem Definition**

Theme includes controlling of home appliances with mobile app. Services one may get are security to get into home only by the owners and residents of that house, to control the appliances Less Energy consumption and time saving. Useful for working and busy schedule people.

* 1. **Motivation**

The purpose of doing IOT based project is to get exposed to how hardware and software are interfaced. As we know nowadays use of phones is in hike and people are more fond of making their life easier by using some apps. So we thought of such an app which helps one to control home appliances through an android app.

* 1. **Scope and Objectives of the project**

All the android mobile phone users can use this app main objective is to maintain security so that only authorized users can get access to the usage of home appliances and especially control on door, The use of appliances is also given priority based on the category of user

* 1. **Explanation of proposed system with simple block diagram**

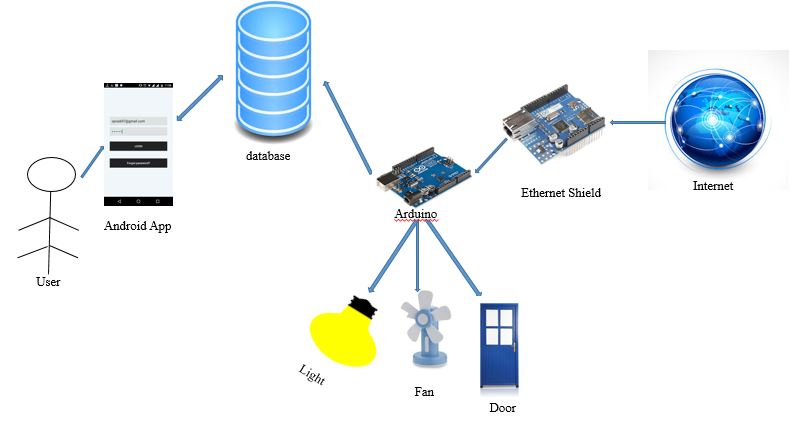


Figure 1

As shown in figure 1 the end user operates on the electrical appliances as well as get access on the control of door based on the category of login, app communicates with the database and then the control is passed on to the arduino which actually turns the appliance on or off which get the database access through Ethernet shield.

1. **Literature Survey**
   1. **Existing System(s) (Advantages and Limitations of existing System)**

The existing systems have the facility to control the appliances through app but as per our survey there is no app which has the access to appliances based on login category.

* 1. **Description of Target users**

House Owner – same house residents – children – servants, if allowed by the admin are the end users who get access to the app, based on the category they get access to the appliances.

* 1. **Advantages/applications of proposed system**

1. Home safety using authentication.

2. Controlling Home lights.

3. Unlocking doors through phone.

4. Controlling electrical appliances through phone. (light, fan)

5. Services differ from person to person and help in safety.

6. Owner can unlock door, control electrical appliances and view log history.

7. Other family members can unlock door and control electrical appliances.

8. Children can and control electrical appliances.

* 1. **Constraints or limitations of proposed system**

1. Only android phone users can access the app.

2. Cannot be accessed without internet.

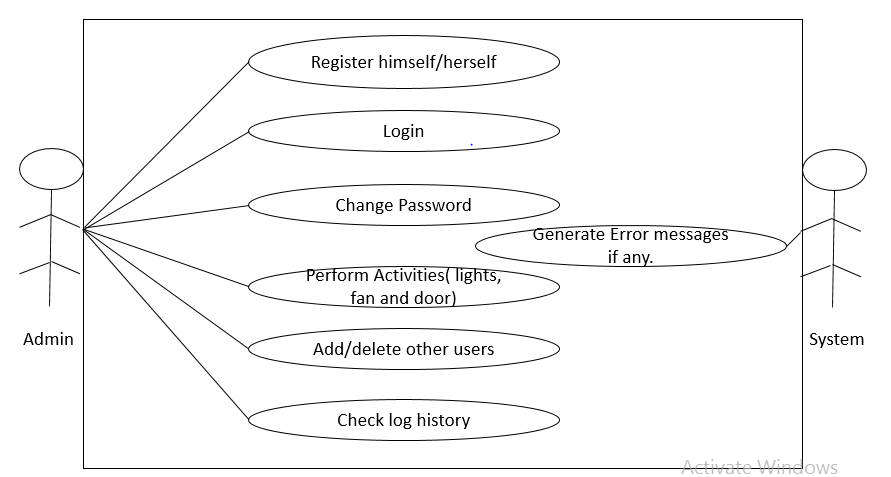
1. **Software Requirement Specification**
   1. **Overview of SRS**

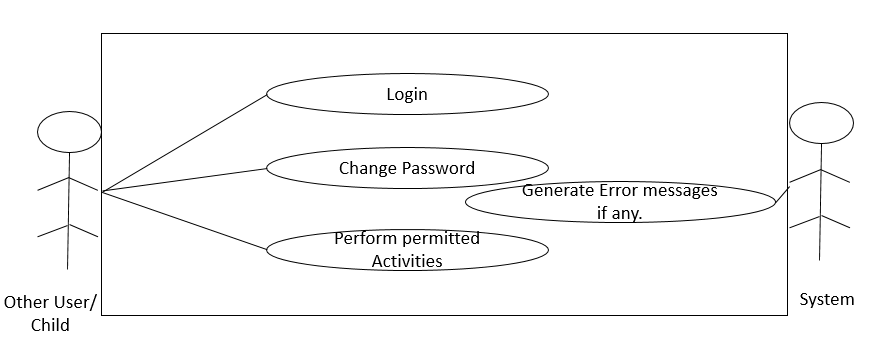
The SRS document explains the requirements for the project including functional requirements and non-functional requirements along with use case diagram and description. Which is helpful for any person who want to design a similar system, or want to learn the existing system to apply in their system.

* 1. **Intended audience**

Developer of the proposed system is the main user of the SRS document, who frequently refers to the requirements and designs the system as to meet all the requirements specified in the SRS document.

* 1. **Requirement Specifications**
     1. **Functional Requirements**
        1. User shall be able to create an account as an admin
        2. Admin shall be able to register/delete other users
        3. Registered user shall be able to login to his/her account by submitting login credentials.
        4. Registered user shall be able to change password of his/her account.
        5. Registered user shall be able to access the account by contacting the admin if he/she forgets the password.
        6. Admin can view the log history which contains the record of users logged in and the time of login
        7. Admin and all other registered users shall be able to turn on and off light and fan, and control doors accept child category
        8. Child user shall be able to turn on and off light and fan
     2. **Use case diagram**





* + - 1. **Use Case Description for Login**:

Actors: registered user

Pre-Conditions: User has an account in the system.

Post Conditions: User will be able to perform all the functionalities of the system provided to him

Success Scenario:

* + - 1. User will enter the email id and password.
      2. System will authenticate the user credentials.
      3. User gets logged in.

Exception Scenario:

1.a User has forgotten the password, he can contact his admin and get the password.

1.b If either the username or password is wrong an error message is displayed.

* + - 1. **Use case description for Register**:

Actors: User (if it’s first time) or Admin

Pre-Conditions: User has installed the app successfully.

Post Conditions: User will be able to register to the app and shall be able to login

Success Scenario:

1. User or admin will enter all the credentials.
2. User gets registered. Or admin registers the other user successfully.

Exception Scenario:

1.a If any of the required field is left empty, error message is displayed

* + - 1. **Use case description for viewing the log history:**

Actors: Admin

Pre-Conditions: Admin has logged in successfully

Post Conditions: Admin shall be able to view the log history containing the details of logged in users and login time.

Success Scenario:

1. Admin will click view log history button.
2. System redirects the screen.
3. Admin will be able to see the details.

Exception Scenario:

2.a If connection to the server is lost, user has to reconnect to the server to see the information.

* + - 1. **Use case description for Changing Password.**

Actors: Registered User

Pre-Conditions: Registered user must be logged.

Post Conditions: Registered user shall be able to change the password successfully.

Success Scenario:

1. Registered user hits the change password button.
2. Registered user enters the mail id and the current password.
3. Registered user enters the new password.
4. Password will be changed in the database.

Exception Scenario:

2.a If the registered user has forgotten his password, he cannot change

* + 1. **Nonfunctional Requirements**
       1. **Performance requirements**

The system must be connected to the server whenever the registered user wants to use the application.

The system server must be on 24x7.

* + - 1. **Security requirements**

Every user must have a registered email Id and password to use the system

* + - 1. **Usability**

The application runs on different versions of android operating system

* + - 1. **Response Time**

The device gives response in 2 ms.

* 1. **Software and Hardware requirement specifications**

Arduino

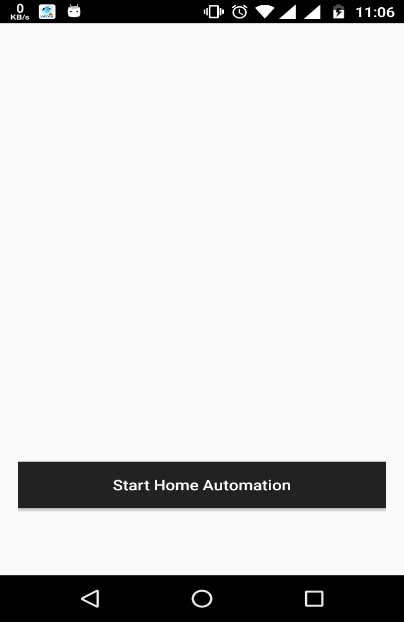
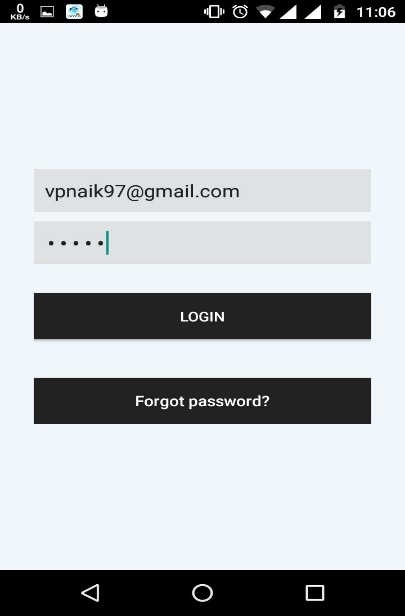
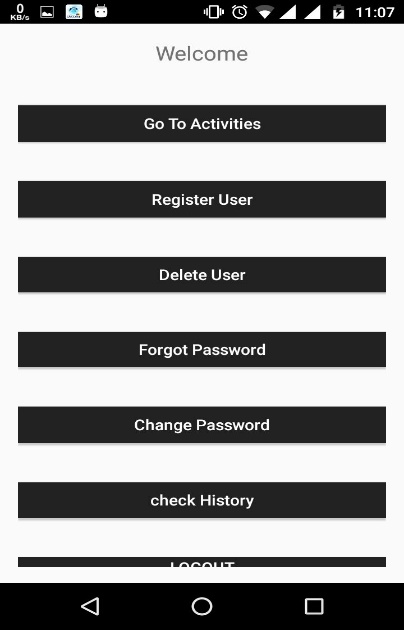
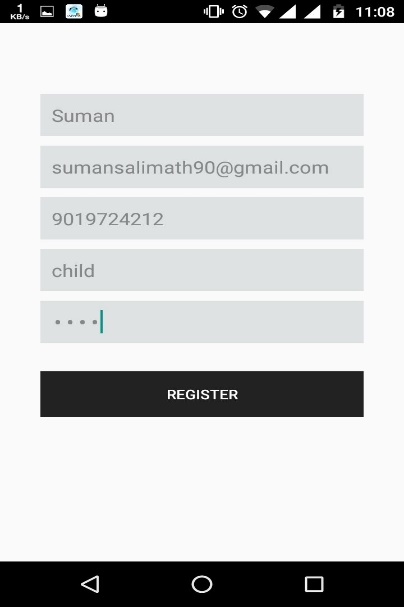
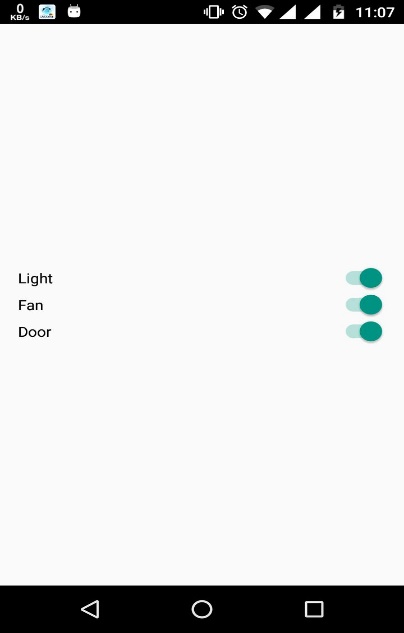
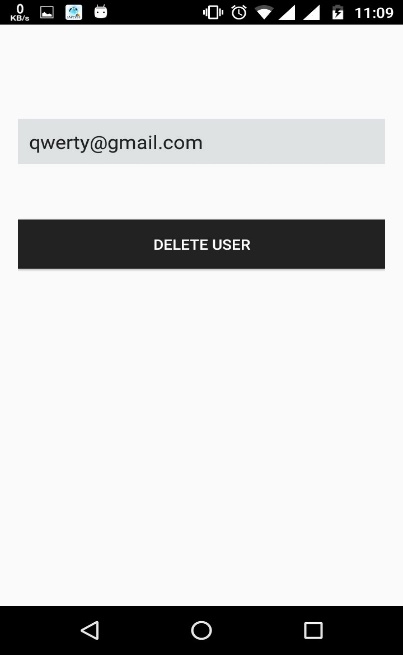
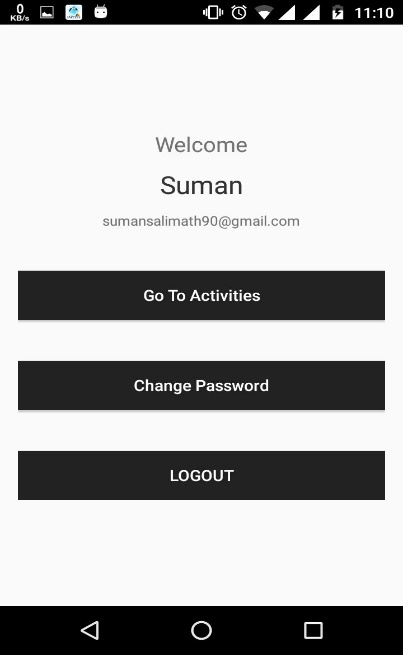
Ethernet Shield

Android Studio

My SQL

Arduino IDE

* 1. **GUI of proposed system (snap shots)**

* 1. **Acceptance test plan**

**Register:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Input description | Expected results | Actual output |
| 1 | The user or the admin fills the details like email Id, phone number, category, password, Username | User is registered |  |
| 2 | If any of the required field is empty | Error message is displayed |  |

**Login:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Id | Input description | Expected results | Actual result |
| 1 | User is already registered | Login screen is displayed |  |
| 2 | User enters valid email Id and password | User is logged in |  |
| 3 | User enters invalid email Id or password | Error message is displayed |  |
| 4 | If any of the required field is left empty | Error message is displayed |  |

**Activities for admin:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Id | Input description | Expected results | Actual results |
| 1 | The admin can add the user by giving the details of the user for registration | User shall be registered if all the details are valid else error message is displayed |  |
| 2 | Admin can access the functionalities like turn on or off fan, light and door | Respective accessories get activated if there is no problem in the server. |  |
| 3 | Click on view history button to check log history | Log history is shown if, no error in server |  |
| 4 | Enter the details to delete a user like email ID | Corresponding user is deleted |  |

**Activities for other user:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Id | Input description | Expected results | Actual result |
| 1 | Other user turns on or off light/ fan through app or opens or closes the door | Light/fan will be turned on or off through arduino or door will be opened or closed. (provided server is on) |  |
| 2 | Enter the details required to change the password email Id, current and new password. | Password will be changed |  |

**Activities for child:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Id | Input description | Expected output | Actual output |
| 1 | Other user turns on or off light/ fan through app | Light/fan will be turned on or off. (provided the server is on) |  |
| 2 | Enter the details required to change the password email Id, current password, new password | Password will be changed |  |

4. **System Design**

* 1. **Architecture of the system**

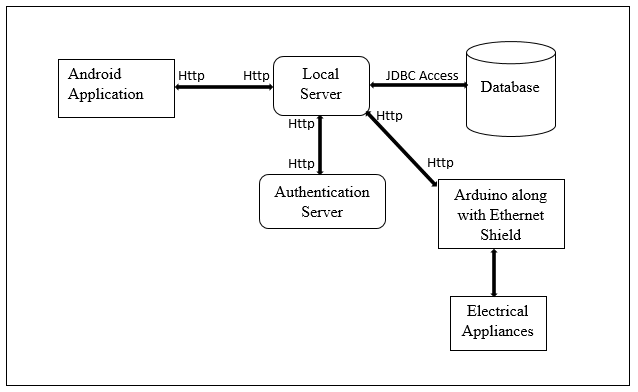


Figure 2

As shown in figure 2 end user uses the android application to access all of his functionalities, the app communicates to the system through the database via server in which all the information is stored, there will be a authentication which provides legal access to the system, actual implementation is done by the arduino device which uses the data in the database and sets the appliances to that value by accessing the data with the help of Ethernet shield.

* 1. **Level 0 DFD**

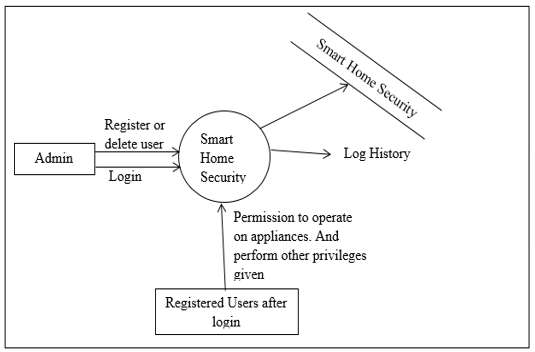


Figure 3

As shown in Figure 3 for our android application system i.e. “Smart Home Security” the admin is the only user who can register and has access to all activities and is like a source. The end user or the registered user can act like a sink with certain privileges. The concerned record for the authentication purpose and other activities are stored in database named “Smart Home Security”. Log history is the report which will be obtained at the end, and which can be accessed only by the admin

* 1. **Detailed DFD for the proposed system**

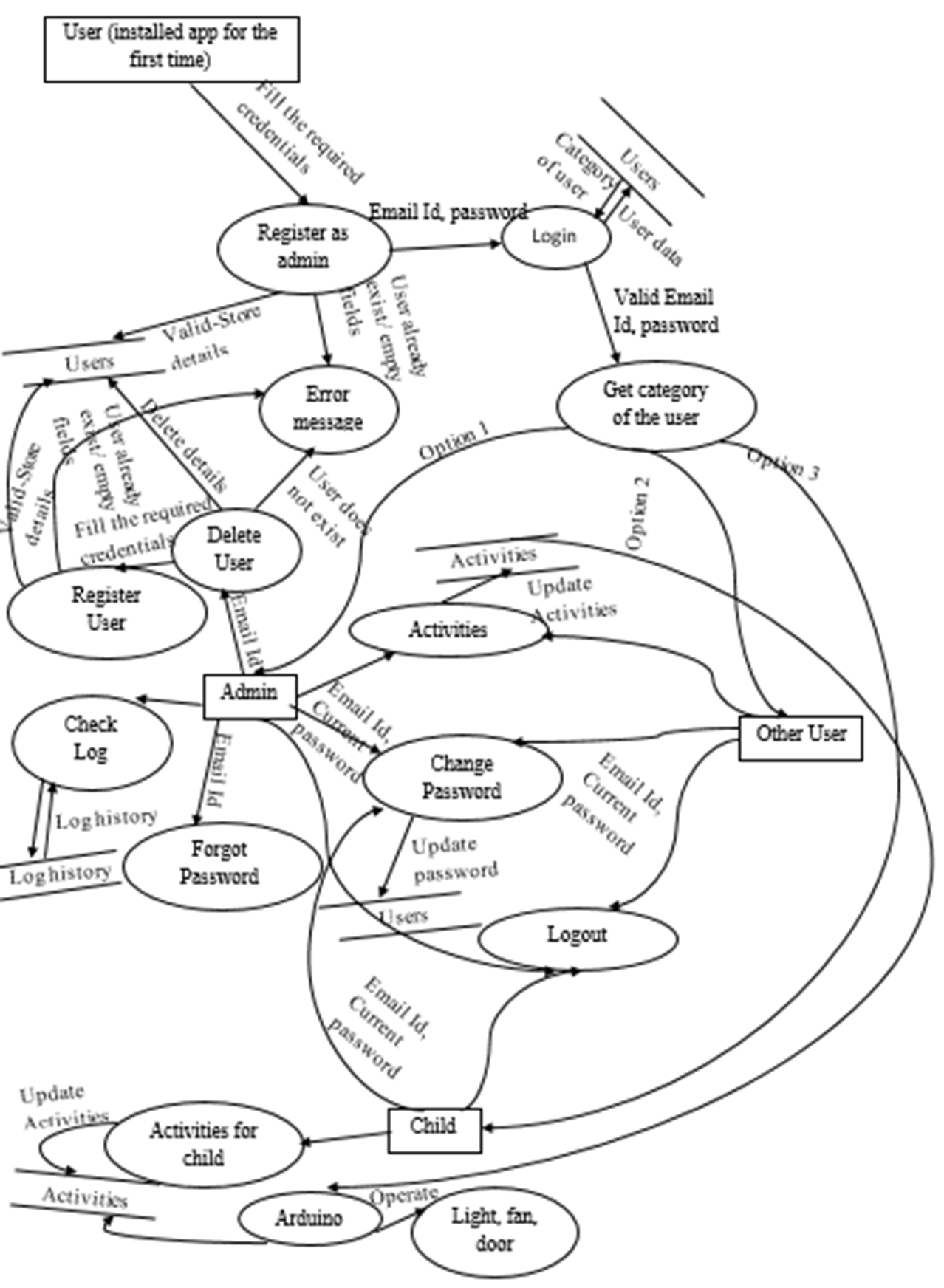


Figure 4

In accordance with the functional requirements presented earlier and from figure 4, the android application “Smart Home Security” provides the user with newly installed app to either register as admin or login as other user or child if is already registered by the admin and has the password to proceed with, the credentials during registration are stored in database under users relation, and the log in time of each user is stored in log history. During every login session the email id and password which are provided as identifying factors are authenticated, if valid credentials are input, the user is redirected to his/her respective screens based on the category of the registered user. For an invalid login credentials respective error message is given to the registered user. The user is not allowed to register twice with the same email Id which is checked during registration.

Once the registered user logs in successfully, he/she access the different functionalities provided to that category of users.

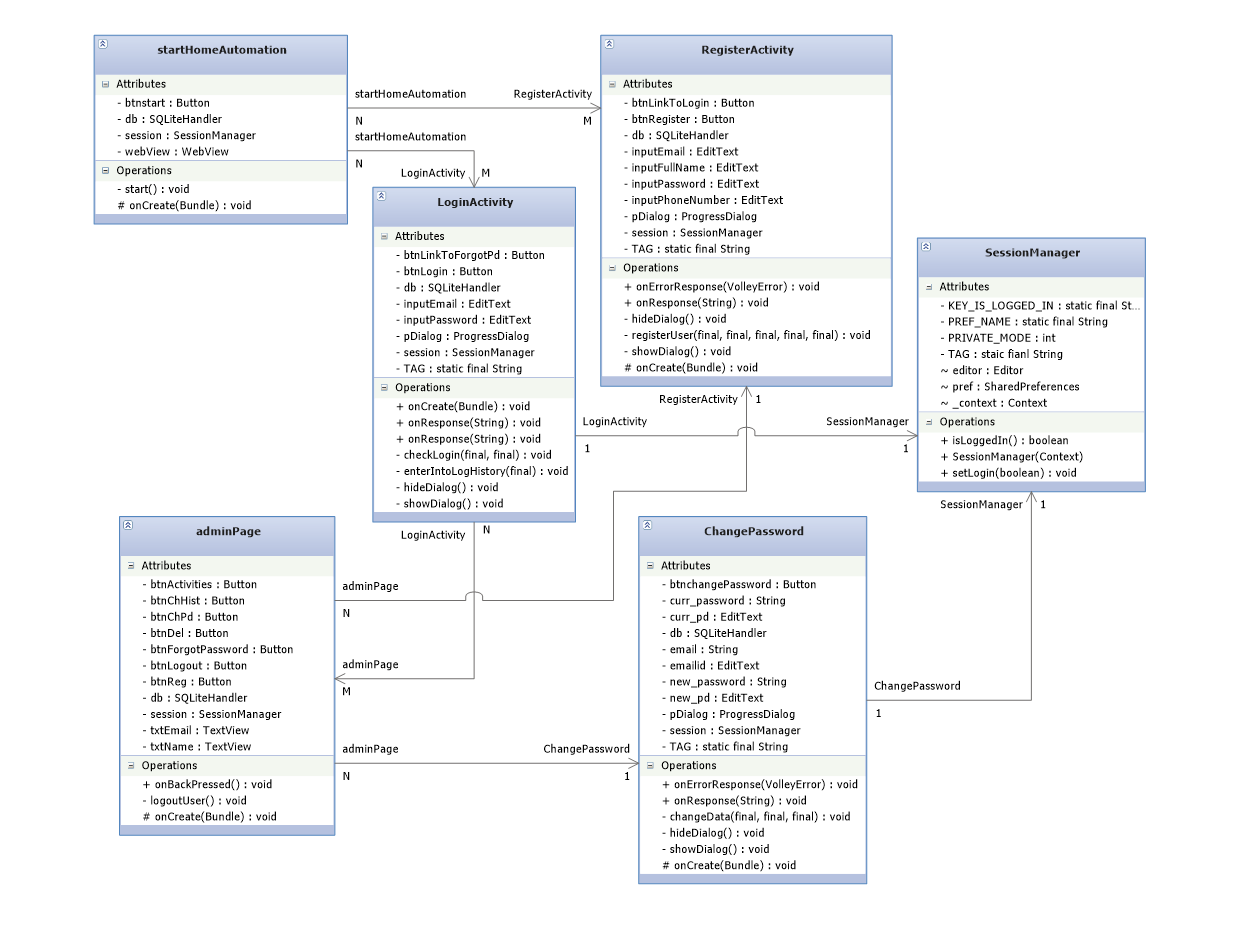
Admin shall be able to register or delete users, by providing all the required credentials like, email Id, Phone Number, category, Password whenever a new user is registered the data is stored in the database with encrypted password and a salt value which is further used to check login password, he shall be able to delete the registered user by providing the email Id, if valid and existing email Id is provided the deletion is achieved by deleting that user’s record in the database, else an error message is displayed telling that the user with that email Id does not exist, if any of the registered user forgets password they have to contact their admin to access their account, admin shall be able to check the log history as to see when and which user has logged in into their account. He can access his activities like to operate upon light, fan, door if already the desired electrical appliances are turned on data is retrieved from the database and is pre-set in the android application. He can log out his account.

Other user can access the activities as the admin does with equal privileges and can change password as the admin does and can logout from his/her account.

Child can access the activities like light, fan, and can also change password and can logout from the account in the same way as other user and admin does.

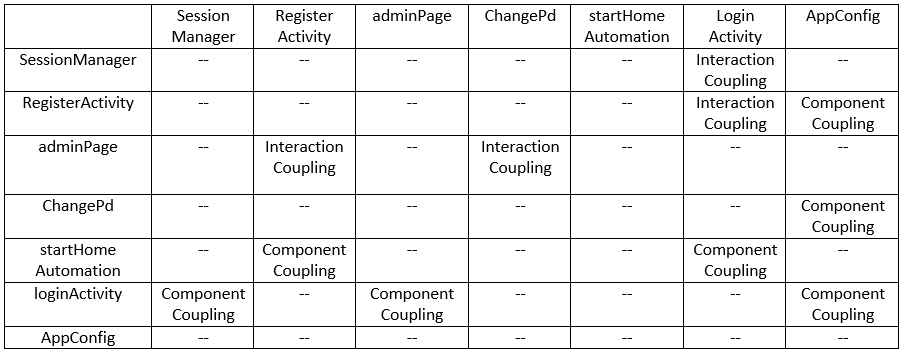
The arduino connected to the server through Ethernet Shield continuously keeps retrieving the data from the server for every 5 seconds and keeps updating the electrical appliances i.e. it turns on or off the device based on the value of 1 or 0 in the database.

* 1. **Class Diagram**

****

|  |  |
| --- | --- |
| RegisterActivity |  |
| OnCreate(): | Sets the RegisterActivity activity. |
| registerUser(): | Takes in all the credentials required and the request is made to server. |
| onResponse(); | Checks the response got from the server, on success stores the details in the SQLite and asks the user to login to continue |
| onErrorResponse(): | An error message is displayed |
| showDialog(): | Shows a dialog if the admin has requested for registration and the process is going on. |
| hideDialog(): | When the register functionality is processed the dialog is hidden. |
| AdminPage |  |
| onCreate(): | Sets the AdminPage activity. |
| onBackPressed(): | Back button is locked. |
| logoutUser(): | Sets the islogged flag to false and sets to login screen. |
| LoginActivity |  |
| onCreate(): | Sets the LoginActivity activity. |
| checkLogin(): | Takes the credentials required for the login activity and requests the server for the authentication. |
| onResponse(): | If the authentication is success the details of the user is stored in SOLite database, to be used further in the application. And based on the category of the user the screen is redirected to their respective screens. |
| onErrorResponse(): | An error message is displayed |
| enterIntoLogHistory(): | If the user is the admin he can request the server to view the log history, the admin is redirected to a web view with the history of users logged in and their login time. |
| showDialog(): | Shows a dialog if the registered user has requested for login and the process is going on |
| hideDialog(): | When the requested functionality is processed the dialog is hidden. |
| AppConfig | Static variables are declared for using the uRL’s of the PHP files. |
| ChangePd |  |
| onCreate(): | Sets the ChangePd activity |
| changeData(): | Takes the current password and new password and request to the server is sent |
| onResponse(): | If updating the password is success a message is displayed giving the confirmation of change |
| onErrorResponse(): | The probable cause for error is displayed |
| showDialog(): | Shows a dialog if the registered user has requested for changing the password and the process is going on |
| hideDialog(): | When the requested functionality is processed the dialog is hidden |
| SessionManager |  |
| SetLogin(): | A flag is set whenever the user logs into his/her account |
| IsLoggedIn(): | Checks whether the user is already logged in, and goes to the respective screen if he’s already logged in |
| StartHomeAutomation |  |
| OnCreate(): | Sets the StartHomeAutomation activity |
| Start(): | Starts the application and goes to register screen if the app is installed and is opened for the first time and goes to login screen if other time the app is opened |

* 1. **Class coupling matrix and explanation**



We have used component coupling where sharing of common elements was required, like here AppConfig is a static class which consists all the static values of the URL’s used for the processing of PHP files. Other classes like login activity register activity and activity page uses the URL from the static class to send requests to the server.

. We have used interaction coupling where based on certain value the screen redirection takes place, like session manager is the class used to decide on redirection of page to login or register, because, if the user has run the app for the first time after the installation, then the user must be redirected to register page, later if he already has the email and password, then he can be redirected to login page, later if he has already used the application in his phone, he is directly given the login page for login, and can access his functionalities

1. **Implementation**
   1. **Introduction**

The programming language used in our project in java, php and arduino programming, coding standards like readability, indentation, modulation are followed as per the standards of java programming, as given in the link.

http://www.oracle.com/technetwork/java/codeconvtoc-136057.html

In the entire process of completing our project we practiced pair programming, one used to be the driver and the other used to be the navigator, the driver used to code the program actually and the navigator used to keep checking the code and guide so that the code could be error free, by practicing this we were able to find out many exceptional cases and errors which were fixed at the coding process only, rather to wait till the testing phase.

* 1. **Brief description of modules which are implemented in pair**

Among all modules Register, login, and activities given for the user like turning on or off the light/fan and closing or opening of door, were implemented in pair, during the implementation many exceptional cases were found like, what if any of the field is left empty, if already registered user again tries to register, if the user who has not yet registered tries to login or if any of the field is left empty while login, if the password or email Id is wrong, and in activities what if the user tries to turn on the device which is already on since the same device can be controlled on multiple devices. These cases were handled due to practice of pair programming which helped us to predict all possible exceptions which could be encountered after deployment.

* 1. **Modules descriptions**

Module name: Register

Input: email Id, username, phone number, category, password of the user who wants to get registered

Output: The user will be registered if the details given are valid.

Pseudo code:

On button click of register

Screen for register is opened

Fill the details: email Id, phone number, username, password, category

Server request is sent to add the details into database.

If the user with same email id doesn’t exist and all the details are valid

The user details are stored in the database, and a message is displayed as the user is successfully registered.

Else

Error response is obtained with probable cause for the failure

Module name: Login

Input: email Id, password of the registered user who wants to login to the system

Output: The user will be logged in if the details given are valid.

Pseudo code:

On button click of login

Screen for login is opened

Fill the details: email Id, password

Server request is sent along with the email Id and password to authenticate.

If the user with given email id and password exists

The user will be logged in, and a message is displayed as: the user is successfully logged in and the boolean value isLoggeged in is set to true.

Activity screen is shown based on the category of the logged in user.

Else

Error response is obtained with probable cause for the failure

Module name: Activities

Input: Toggle switch

Output: The appliances are turned on or off

Pseudo code:

On button click of activity

Screen for activities is opened

Toggle the switch as per requirement

Server request is sent along with the values of switches.

If the server and the arduino along with Ethernet shield is connected and is working perfectly fine

The operation is carried out, arduino sets the appliances based on voltage value

Else

Error response is obtained with probable cause for the failure.

1. **Testing**
   1. **Introduction**

Unit testing was carried out after each module of implementation so that testing of entire system will not become overhead at the end, since our project includes many modules.

White box testing was carried out since the testing was carried out by the programmer itself, all the test cases were written based on the execution of statements in the program, to see whether all of the statements will execute at least once in different scenarios, since the tester and programmer were the same, whenever there was an error which was not handled, it was fixed at the same time, since all the internal functionality was known to the tester.

* 1. **Module test plan (unit testing of each module) and test cases**

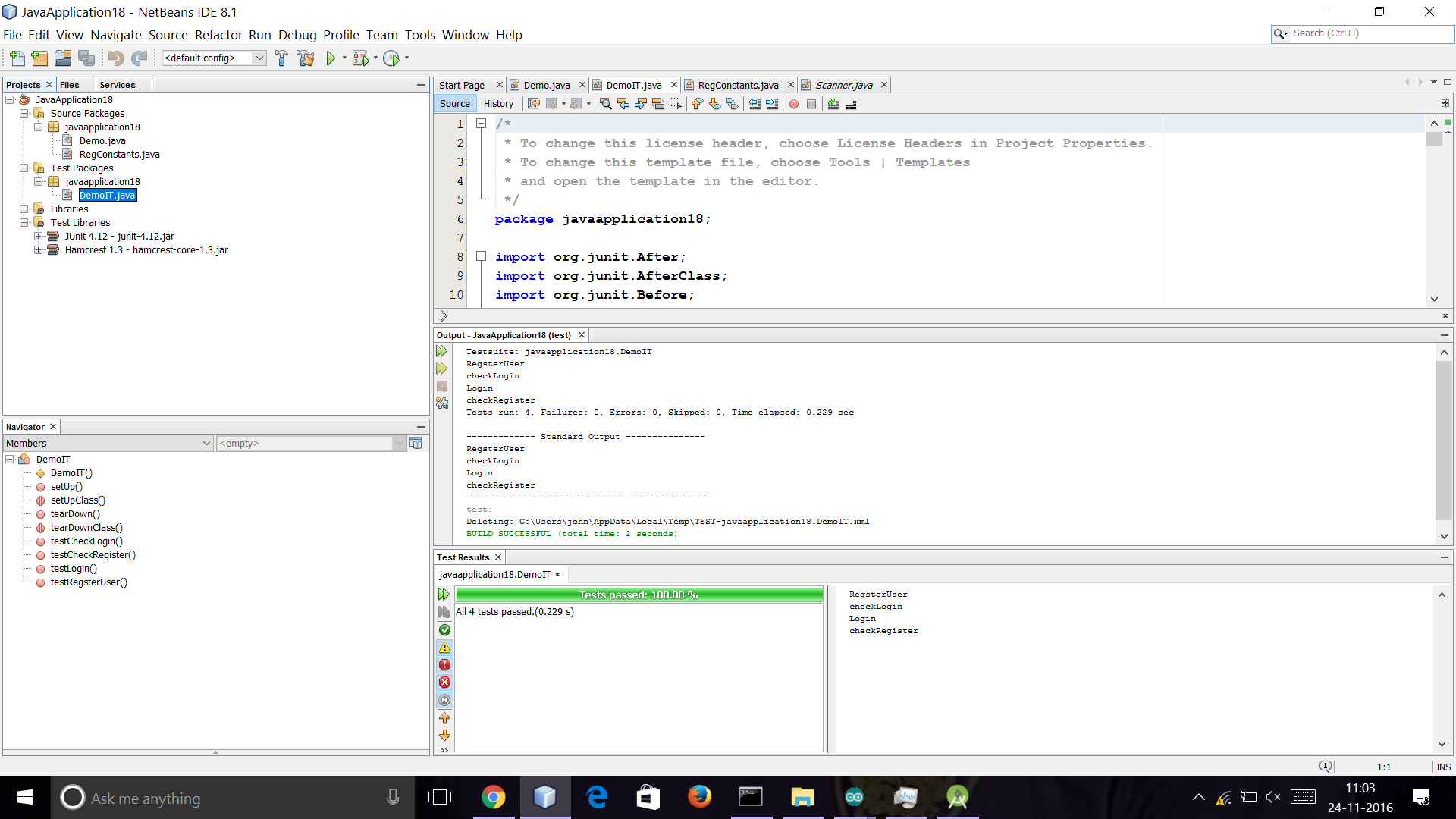
Register:

|  |  |  |  |
| --- | --- | --- | --- |
| Test Id | Input | Expected results | Actual results |
| 1 | Valid email Id, username, password, phone number, category. | User will be registered. | User is registered |
| 2 | Any of the field is empty. | Error message is displayed. | Error message telling fields cannot be empty is displayed. |
| 3 | If the user with same email ID is already registered. | Error message saying, the user is already registered is displayed. | Error message saying, the user is already registered is displayed. |

Login

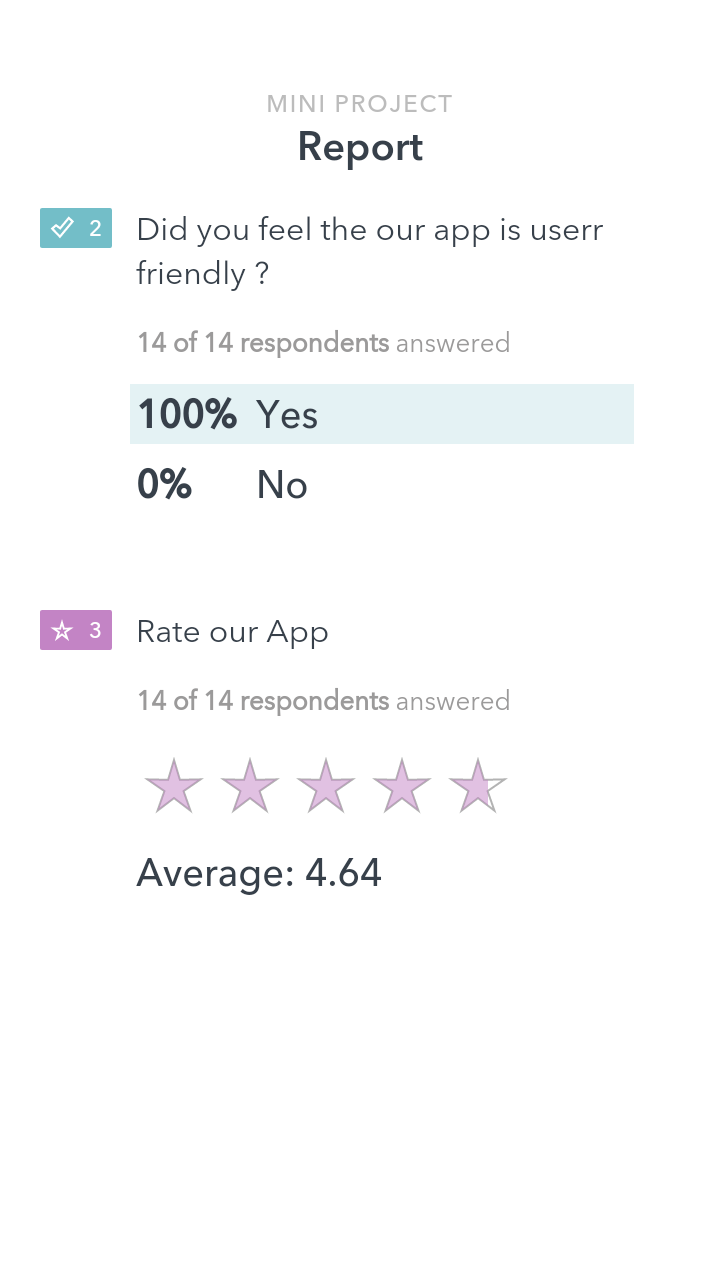
|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Input | Expected results | Actual results |
| 1 | Valid email Id,  Password. | User will be logged in | User is logged in |
| 2 | Any of the field is empty | Error message is displayed. | Error message telling fields cannot be empty is displayed. |
| 3 | Invalid email id or password | User should not be logged in | User is not logged in, giving an error message that, one of the field is invalid |

* 1. **JUNIT testing**

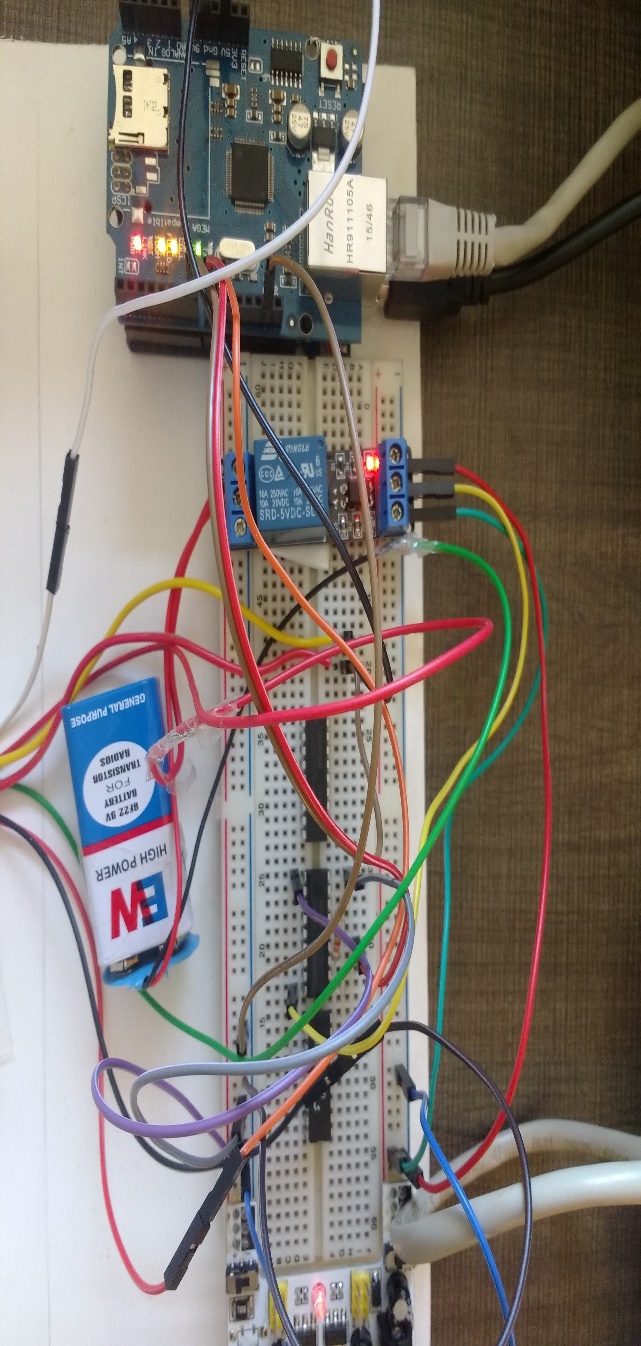


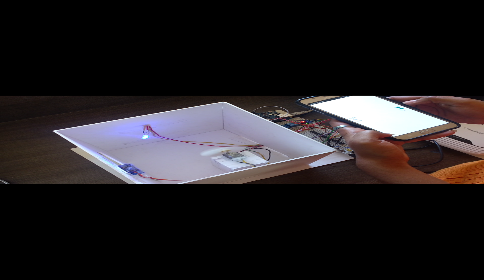
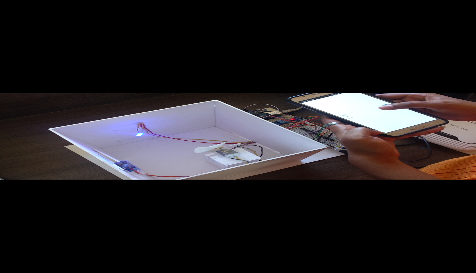
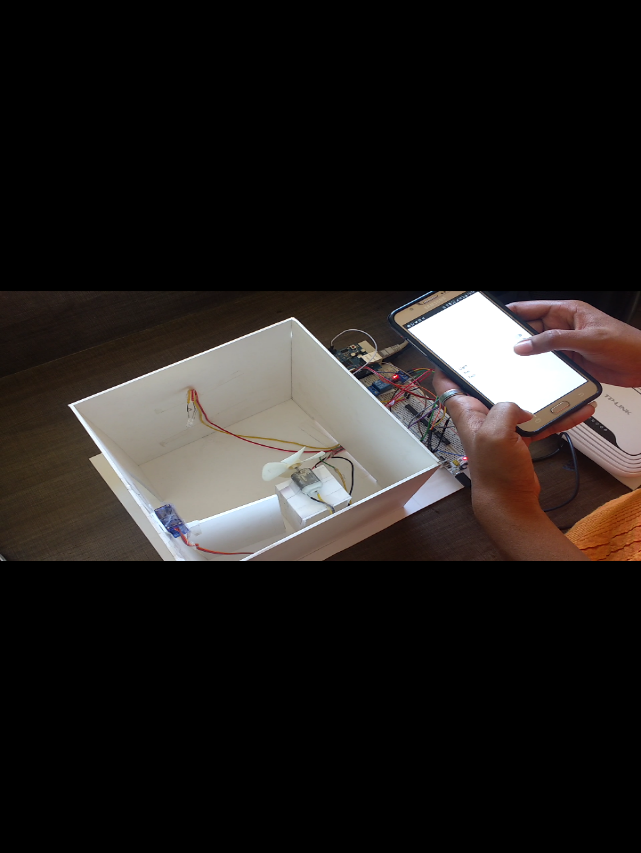
* 1. **Usability testing**

The brief analysis of the feedback got by the users of the app we developed is (the app was given to use for the people before us to test the hardware working functionalities like, turning on or off light, fan, door and other functionalities like login and register)

****

1. **Results**





1. **Summary of learnings**

Problem statement Analysis:

The survey was conducted by us in order to find the existing apps for the same application and to decide of doing something new.

Requirement Analysis:

The use of components to be made was decided and later few modifications were also made and the total cost was decided.

Design phase:

The exterior design was decided by all of us keeping in mind our project requirements and also feasibility of the user.

Coding: As we had decided to do an app we choose android studio as the medium for coding and learnt the new features of java coding for building an app.

Testing: We were exposed to various testing tools, we choose JUNIT for testing our app.

1. **Conclusion and future scope**

As per defined problem statement we have completed the project on implementing IOT based SMART HOME SECURITY which includes all the features like controlling light, fan and door through an android application. At the beginning of problem statement analysis we came with the solution of making automated door using electromagnet, later due the fact that we couldn’t generate such a strong electromagnetic field we ended up making an automated door using servomotor. Future scope is to extend the application not only for controlling lights, fans and door but also for other purposes like gas leakage detection etc.

1. **References/Bibliography**

[1] <http://www.androidhive.info/>

[2] <https://developer.android.com/reference/android/content/ContentProvider.html>

[3] <https://www.youtube.com/watch?v=-feVvHv-vpE>

1. **Appendix**
   1. **Explanation on JUNIT tool**
      1. Introduction**:** JUnit is a [unit testing](https://en.wikipedia.org/wiki/Unit_testing) [framework](https://en.wikipedia.org/wiki/Software_framework) for the [Java programming language](https://en.wikipedia.org/wiki/Java_%28programming_language%29). JUnit has been important in the development of [test-driven development](https://en.wikipedia.org/wiki/Test-driven_development), and is one of a family of [unit testing](https://en.wikipedia.org/wiki/Unit_testing) frameworks which is collectively known as [xUnit](https://en.wikipedia.org/wiki/XUnit) that originated with [SUnit](https://en.wikipedia.org/wiki/SUnit).
      2. What it does: JUnit is an open source framework, which is used for writing and running tests. Provides annotations to identify test methods. Provides test runners for running tests. JUnit tests allow you to test codes faster, which increases quality.
      3. Install Java: JUnit is a testing framework used to test Java based application. So before installing JUnit, you need to configure or verify java development kit (JDK) in your machine.
      4. Download JUnit

Visit <http://junit.org/junit4/> and click Download and Install

* + 1. Download hamcrest core

Visit https://github.com/junit-team/junit4/wiki/Download-and-Install again. Click hamcrest-core.jar