# Home Automation Server Architecture

V1

Flow Diagram:

C++ interface to Sql

PHP Interface to Sql

Sql Database

Web

C++ interface to Arduino

Android App

C++ Command processor

Functional Description:

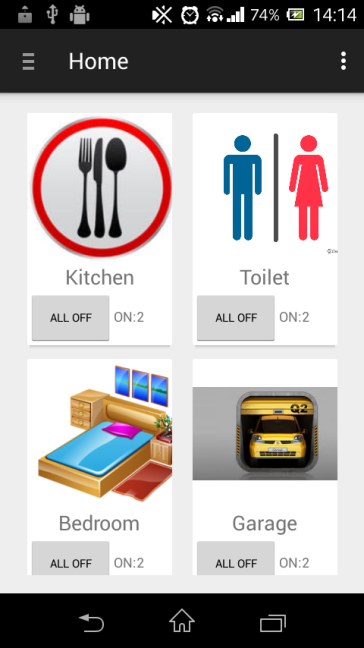
1. **Android App:** This will be a Java program which will interface the user to various settings and actuations of the home automation system. It will talk to the php interface by HTTP Post method.
2. **Php Interface :** This will interface incoming http post commands to the mysql database resident on the Pi. Its job is to only translate commands and store them into the database.
3. **MySql Database:** This will store all data of the home automation system as well as user login data and logs. Commands issued by various apps will be stored to be processed and their respective results also will be updated here.
4. **C++ Sql Interface:** This is a sql listener which listens for commands and translates the same to the command processor.
5. **C++ Command Processor:** This will receive any commands with arguments and process them and take necessary actions and may also invoke the arduino interface if required.
6. **C++ Arduino Interfacer:** This will translate the processed information to arduino and act as an interface between the arduino and the pi.

Detailed Description:

# Android App

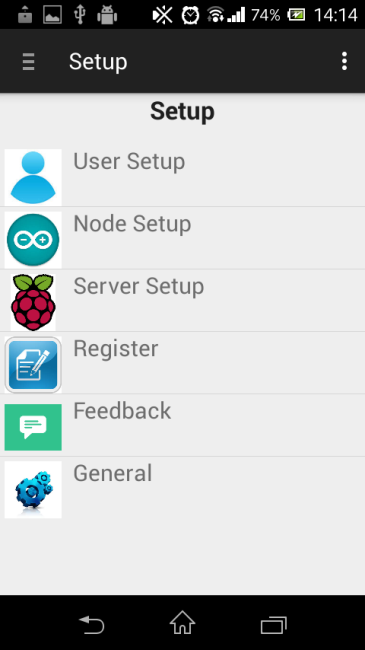
The app is the main interface area between the user and the home automation system. The app will have the following options:

1. **Main Screen :**



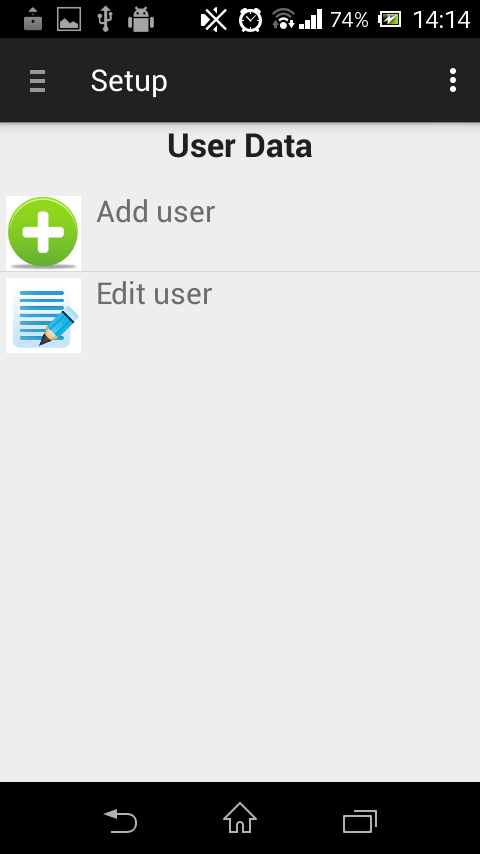
The main screen will open up showing all the nodes and their respective switches. Listeners are incorporated to catch user activity on a particular node and switch and commands are generated.

1. **Setup Screen**

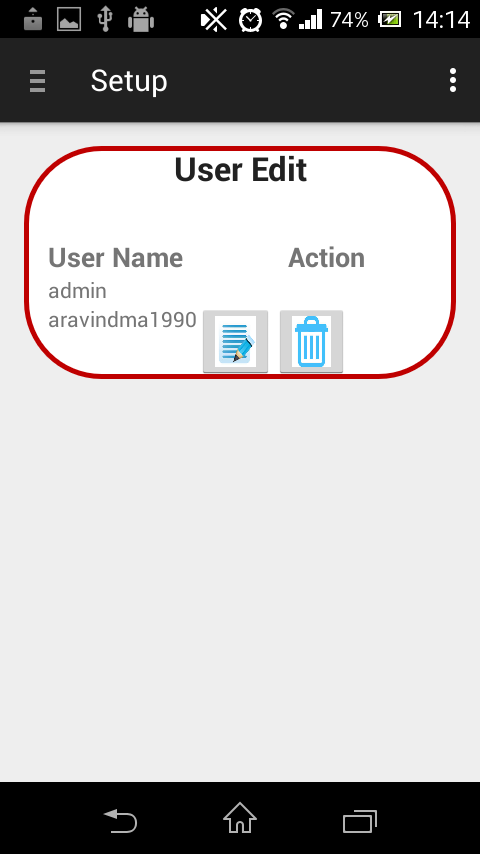


The setup section will have the above options. This is accessible by only the admin who has to login previous to this screen. The login credentials are stored on the server(raspberry pi).

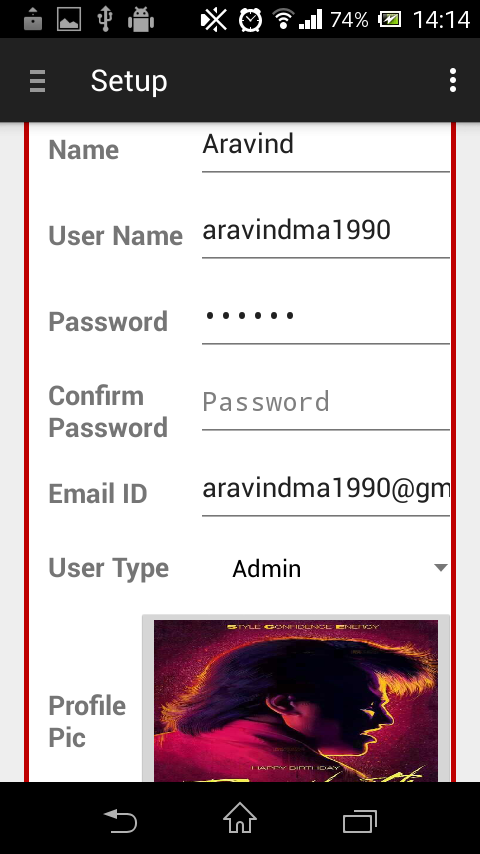
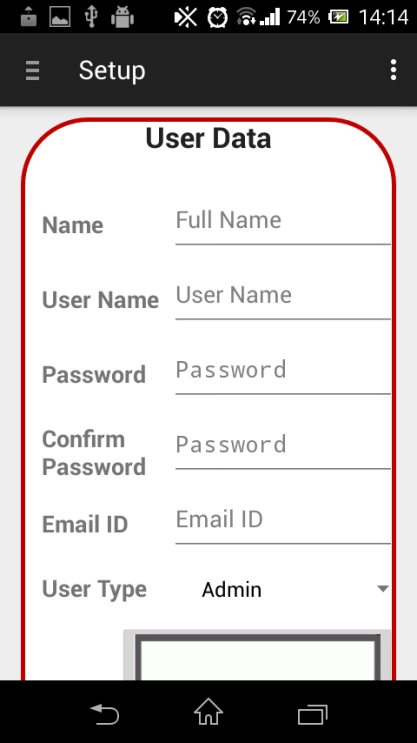
* 1. **User Setup**



User setup will have add and edit user functionalities. This will affect directly at the master database stored on the server(Pi).

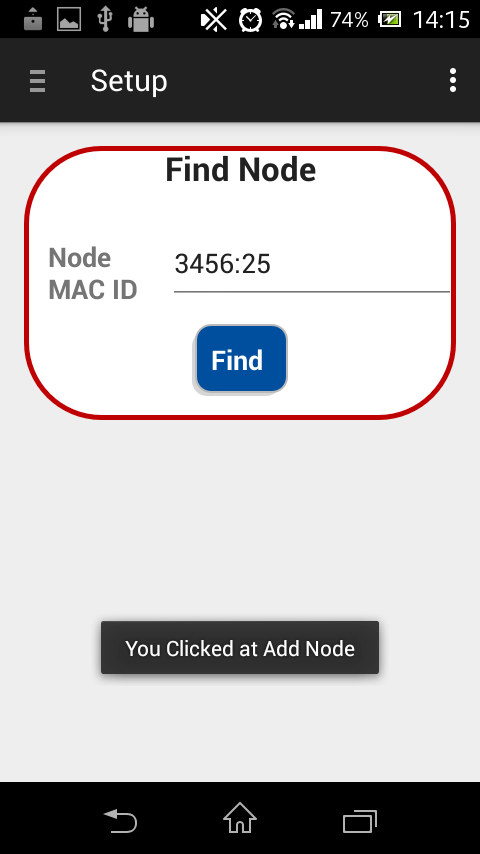


User Edit will show available users from the master database and allows user to delete or edit the same. The main root user admin will not be allowed to edit or delete as it is the technician’s login.

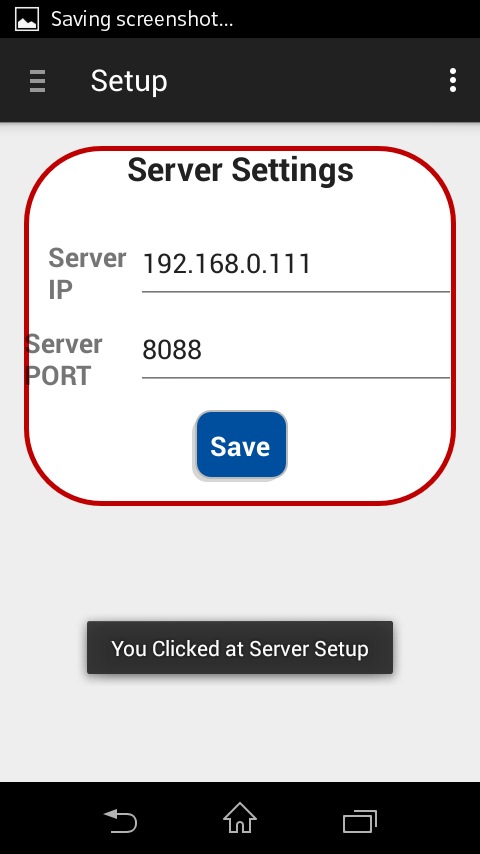


Admin users can add a new user with the above screen. The new user can be of type admin or user. User will not have admin rights and can only control existing nodes. Provision will also be made to let the users only access a few nodes as set by the admin.

* 1. **Node setup**



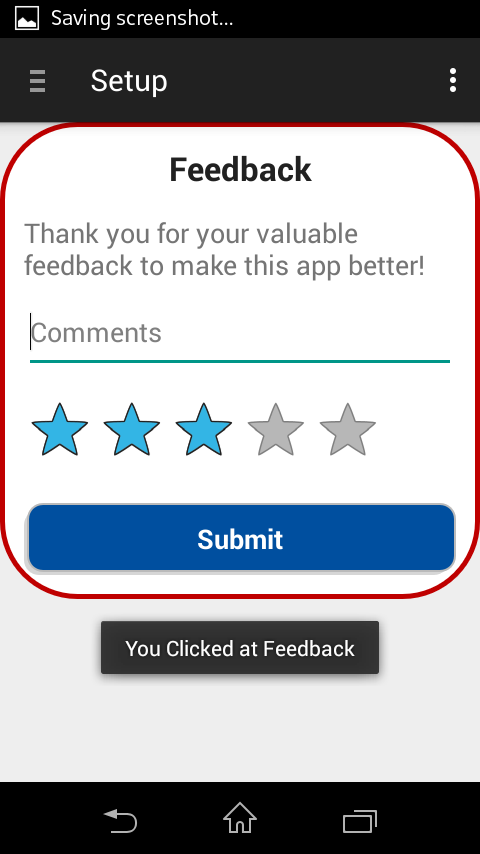
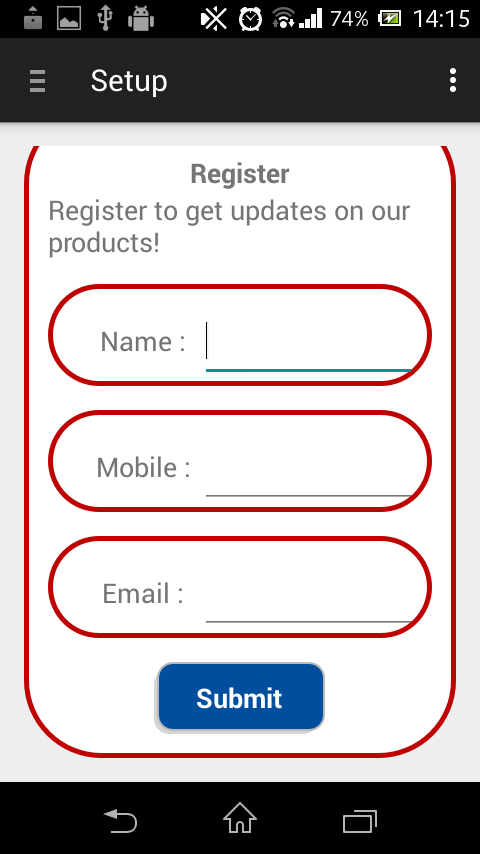
A new node can be added with the add node setup. Users can type in the node mac id printed in the background. The app will then contact the server and try to get the node activated. Once activated the node name and picture can be setup.



* 1. **Server setup**

The server setup page is used to set the IP address and port of the raspberry pi server to access the sql and php files. Once saved, this information is stored locally on the app’s db.

* 1. **Register and feedback**



Register and Feedback options are for users to reach us, The PhotoElectricChefs on our webmail for any reviews/issues/product updates etc.

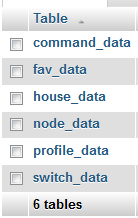
# PHP Interface

The php interface to mysql is a set of php files running on the server(raspberry pi). These are static pages which run a session on invokement from the android,iOs,windows,web apps. The main category of files created are:

1. **CRUD(create read update and delete) operations on User database.**
2. **CRUD operations on Command database.**
3. **CRUD operations on Node database.**
4. **CRUD operations on Switch database.**
5. **CRUD operations on Favorities database.**
6. **CRUD operations on profile database.**

Once invoked, the corresponding files will talk to the database through the user login which has invoked it and make necessary changes in the database. The PHP interface will not stay alive once the necessary task is done. It is up to the app to invoke the interface again to check if any status or entry in the database has changed.

# MySql Database

This database residing on the raspberry pi server has all the information of the home automation system. The contents can be broadly classified as the following tables.

-Table to store all incoming commands and their status

- Table to store favorities

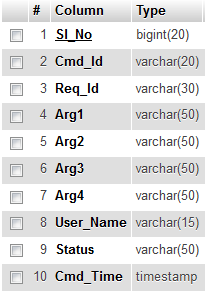
- Table to store user information

-Table to store node information

-Table to store various profiles

- Table to store switch information

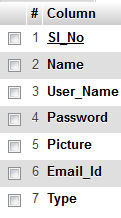
1. **Command\_data**

d

e

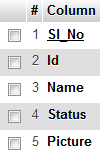
The command table will take input commands from the app through the php interface and populate this table. The command IDs are specific to different scenarios like turning on a switch,adding a node etc. The Request IDs are generated by the app and is used to keep track of the status in relation to the command id. Arg1-4 are arguments that need to be passed along with cmd\_Id to the C processor. For example, in the case of turning on a switch, Node MAC ID,switch ID and switch status would be the arguments. User\_Name is the user\_name from which the command has been issued. The C processor can cross check the credentials with the user database. Status denotes the current status of the issued command. This can range from command-in waiting to error statuses. Cmd\_Time is an auto updater of a timestamp on when the entries were updated.

1. **House\_data**



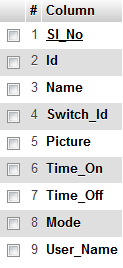
This will contain the various details of the users. Users can be of two types –Admin and User. Admin will have rights to change any of the database and their settings whereas User can only issue command requests to change status of the nodes and switches. User can also be restricted from issuing commands to a few nodes.

1. **Node and Switch Data**



Node and switch data tables will have information of the respective nodes and switches. Nodes can be given a particular name and picture by the user and switches can be activated depending on how many are required. Additionally switch ids are locked to their specific nodes. Switch status can also be locked so as to not allow manual intervention locally at the node or unauthorized usage by other users.

1. **Profile data**



Profile data table lets users and admins create profiles which operate automatically.

Profiles are generated by the user and input into this column by php interface. This will then be polled by the command processor and necessary actions will be taken on conditions being met.

# C Command Processor