Services Management System For Housing Society

By

Muhammad Taha Amin
(BSCS/F14/0117)

Muhammad Areeb Vohra
(BSCS/F14/0112)

Syed Abdul Moiz Hussain
(BSCS/F14/0128)

2018

Faculty of Engineering Sciences and Technology
Hamdard Institute of Engineering and Technology
Hamdard University, Karachi, Pakistan

Services Management System For Housing Society

By

Muhammad Taha Amin
(BSCS/F14/0117)

Muhammad Areeb Vohra
(BSCS/F14/0112)

Syed Abdul Moiz Hussain
(BSCS/F14/0128)

Under the supervision of
Prof. Adnan Ahmed Siddiqui
(Assistant Professor)

2018

Faculty of Engineering Sciences and Technology
Hamdard Institute of Engineering and Technology
Hamdard University, Karachi, Pakistan

Services Management System For Housing Society

By

Muhammad Taha Amin
(BSCS/F14/0117)

Muhammad Areeb Vohra
(BSCS/F14/0112)

Syed Abdul Moiz Hussain
(BSCS/F14/0128)

A project presented to the
Faculty of Engineering Sciences and Technology
Hamdard Institute of Engineering and Technology

In partial fulfillment of the requirements for the degree

Bachelors of Science
In
Computer Science

Faculty of Engineering Sciences and Technology
Hamdard Institute of Engineering and Technology
Hamdard University, Karachi, Pakistan

CERTIFICATE

This project "Services Management System" is presented by **Syed Abdul Moiz Hussain, Muhammad Areeb Vohra and Muhammad Taha Amin** under the supervision of their project advisors, approved by the project examination committee, and acknowledged by the Hamdard Institute of Engineering and Technology, in the fulfillment of the requirements for the bachelor degree of Computer Science.

_____ _____

Prof Adnan Ahmed Siddiqui (Member)
(Project Supervisor) Name & Signature

_____ _____

Mr. Muhammad Fahad (Member)
(Project Co-Supervisor) Name & Signature

_____ _____

Prof. Dr. Aqeel ur Rehman (Member)
(Chairman, Department of Computing) Name & Signature

_____
Acting Director, HIET
(Prof. Dr. Pervaiz Akhter)

ACKNOWLEDGEMENT

All praises and thanks to Al Mighty **"ALLAH**", the most Merciful, the most Gracious, the source of knowledge and wisdom endowed to mankind, Who conferred us with the power of mind and capability to take this project to the exciting Ocean of knowledge. All respects are for our most beloved Holy Prophet "**Hazrat MUHAMMAD (Peace Be Upon Him)**", whose personality will always be source of guidance for humanity.

Acknowledgement is due to **Hamdard Institute of Engineering and Technology** for support of this Project, a highly appreciated achievement for us in the undergraduate level.

We wish to express our appreciation to our **Supervisor Prof. Adnan Ahmed Siddiqui** who served as our major advisor. We would like to express our heartiest gratitude for their keen guidance, sincere help and friendly manner which inspires us to do well in the project and makes it a reality.

Many people, especially our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us an inspiration to improve our project. We thank **Co-Supervisor Muhammad Fahad**, our faculty teachers and all the people for their help, directly and indirectly, to complete our project.

ABSTRACT

This project **Services Management System for Housing Society** is a service you can interact via both Android application and Website to avail and get facilitate with the household services. Although there are a few projects publicly available resembling to this but none of them are made for any specific or particular housing society.

This project provides access to members of the housing society to just request for service in a very easy and productive manner and it also provides workers the opportunity to earn. The workers working for the society can provide multiple services as professionals.

Services Management System provides their customers the impeccable way to request for service and a simple cash on delivery payment method with the 100% customer satisfaction guarantee.

The motivation behind this project was the problems people faces for finding any professional workers. Customers have to find the service men by making phone calls or find one in the hardware shop nearby and the customer has to face the lame excuses of the workers. More over workers are not often available or good enough and also ask for more price that does not contemplate the reasonable prices, so these were the issues we got through and found a solution.

Table of Contents

List of Figures

CHAPTER 1: INTRODUCTION

This project is for the citizens of a society to be able to hire workers like carpenters, plumbers, electricians, sweepers, line-men etc., without even going out of the residential area they are in. Users can either open website or mobile application to request for a service, define their problem and on submission a worker will be en-route to their location.

The project provides easy and a friendly user interface so that the user has no hurdles in solving their household problems.

1.1 Motivation

The proposed project introduces the application which will integrate society's every problem on a platform, and the problem will be on few clicks away to be solved.

1.2 Problem Statement

In housing societies, problems regarding any household issues like electricity, plumbing, gas leakage, carpentry, fumigation, etc. the person himself has to go and bring the repair man, which takes a lot of time and effort.

1.3 Aims & Objectives

i. The main objective of this project is to develop a web and application-based software that is used to inform about the problem of the person is having, concerning any household problems like leakage of gas, water seepage, electricity issues, plumber required for wrecked furniture etc.

ii. As Android based smart phones have become very popular and common devices for innovations, so we are using this platform as a user interfacing medium.

iii. The salient objective to develop a mobile application is for the reduction of complexity in the system and essentially to save people's time.

iv. And definitely a web-based service control manager where all the required services of the users will be entertained and responded.

CHAPTER 2: REQUIREMENT SPECIFICATIONS

2.1 Literature Review

2.1.1 Study of Implementation of Society Management System

The concept of this paper is that to keep the people of a society up to date with their daily activities, by notifying the activities on the application. So majority will go through with every problem or issue, along with, in the societies where people don't interact with each other much, will be able to strengthen their neighborhood. (Gavhane, Vatharkar et al. 2015)

2.1.2 Ubiquitous Smart Home System Using Android Application

The paper defines an android based application which uses an Arduino as a server to control the appliances of a home. This can help us in using the methodology to control not any appliances but the complaints of the society, by using internet server/domain instead of Arduino, as a connecting medium. (Kumar 2014)

2.1.3 Implementation of Society Management System: SOCIETALES

The paper focuses on the online posting of the complaints using android applications for any residential society. A society has to deal with many functions like day to day complaints, involving water supply, maintenance of electricity. So to manage and integrate everything on a single platform would help to maintain a well-defined environment within the society. This project is implemented on the housing society but we thought that it should be applied for every single apartment in the society. Every society has different criteria for the maintenance of apartments but we would make this to create a generic

application once useable for almost every society management. (Vatharkar, Patil et al. 2016)

2.1.4 Ubiquitous Home Control and Monitoring System Using Internet of Things

This project uses the Internet of Things (IoT) to monitor houses in the society and monitor and control appliances of the house too. It uses wireless network or 3G/4G to connect 8051 microcontrollers as a hosting controller.

A flowchart in this project is used to check the online activity of the user that whether the user is connected or not. (Kumbhar and Dilip 2016)

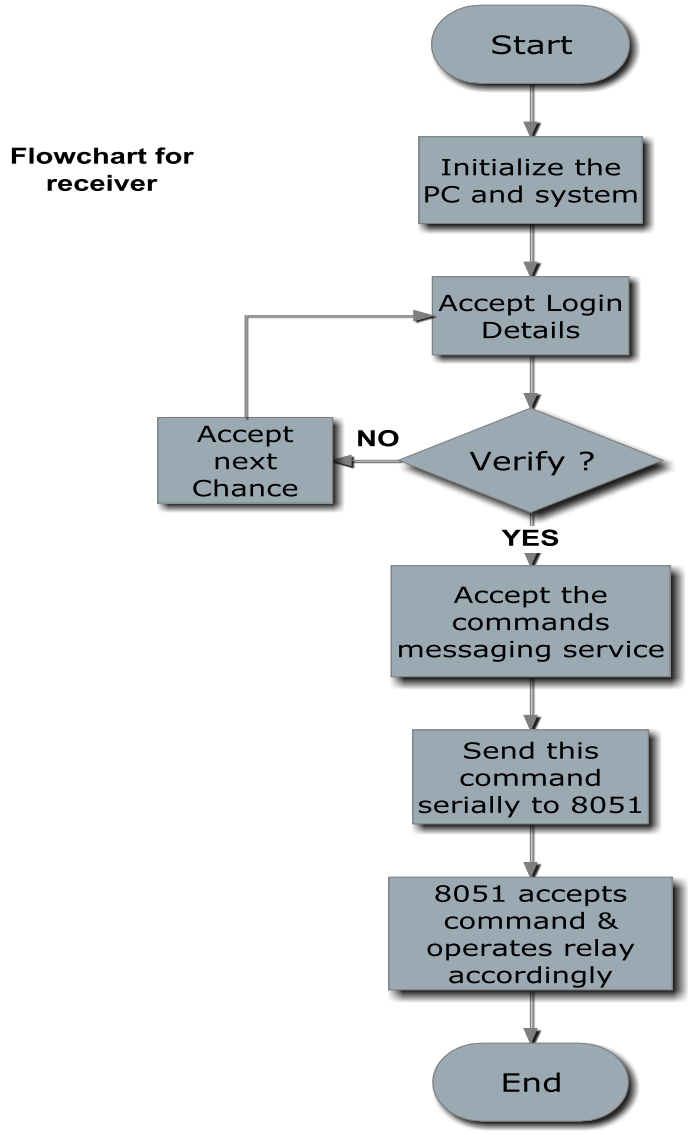


Figure 1

2.1.5 Implementation of Facility Maintenance Management System using Smart Phones

This paper conveys the message that the use of smart phones is increasing gradually and the whole world is in the palm of our hands through high speed internet and cellular internet connectivity, due to which the use of web-based applications on personal computer is decreasing. This paper proposes an integrated android based mobile application and a web-based system that facilitates maintenance management in apartment buildings that saves unnecessary man power that can be used in other aspects.

The methodology of creating an integrated mobile application and web based system is useful to consult the idea of our project. (Joo 2013)

2.2 Project Scope

The proposed project includes a website and an Android application through which user can submit a request for services we have provided which includes electrician for any kind of electronic repairs, carpenter for any furniture damage, fire brigade, ambulance, doctor and plumber for any plumbing issues.

2.2.1 Usage Scenario

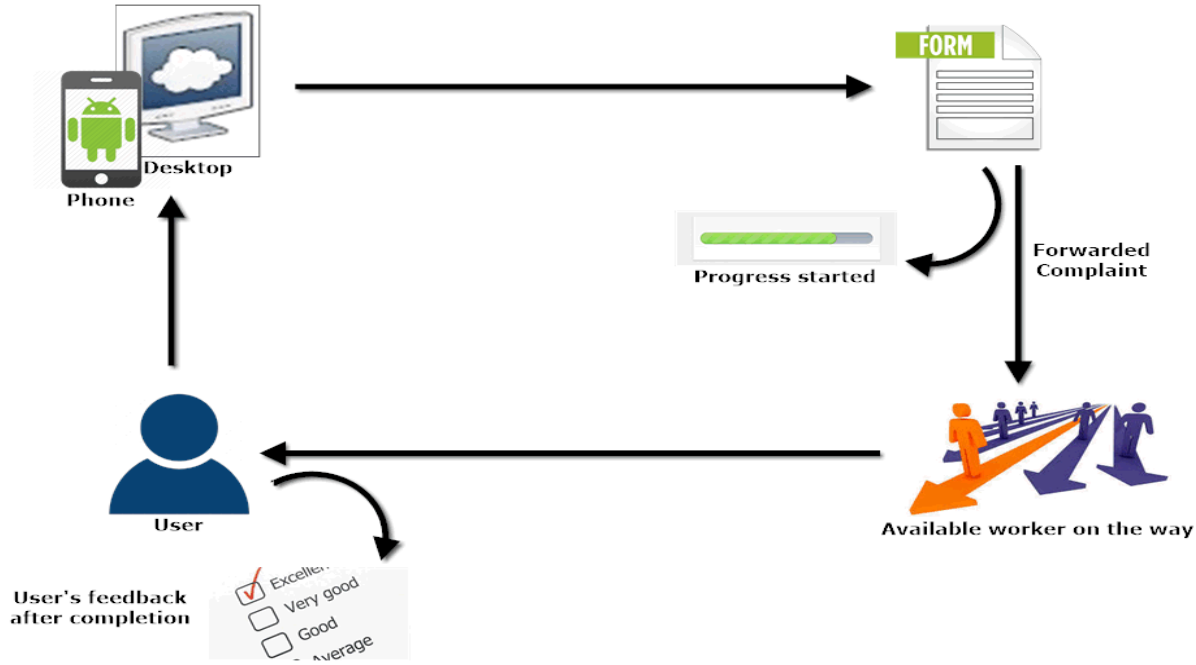


Figure 2

2.4 Functional Requirements

The project has a total of five modules, three on the web application side and two modules on the android application side.

Users are able to give ratings and reviews about the workers they hire, and are notified about any new information of the society from the admins.

2.4.1 Admin Panel:

Through this panel, administrators are able to register new service providers and monitor their activities, view sales reports and go through any technical issues faced by citizens. The admin can delete any user, worker or modify their roles as per the needs.

2.4.2 User Panel (Web App):

This panel allows users to first login to their assigned accounts then search for the service provider of their need, define their problem and get the service at a given time. Users can also check the history of their orders. They can edit their profile name, address, phone etc.

2.4.3 User Panel (Android App):

Android application works the same way as the web app, instead the user doesn't need a desktop or laptop for accessing the web portal. User can also check the history of their submitted orders.

2.4.4 Worker Panel (Web App):

Web portal for workers allows them to add the service they want to provide, receive and view order description and modify status, review sales reports, check their commissions and edit their profile.

2.4.5 Worker Panel (Android App):

Through the Android application of the worker panel, workers can do all the activities that they are allowed to do on the web application, view order description and modify status etc.

2.5 Non-Functional Requirements

2.5.1 Usability

I. A friendly Graphical User Interface.
II. An upgradable system, in future if needed.
III. Availability of android and web application.

2.5.2 Security

I. Every individual user's data is only accessible by that user only.
II. User data will be privacy protected.
III. Request are required to be submitted by the authorized users only so that spams are avoided for workers.

2.5.3 Performance

I. This project is a multiplatform (web/mobile) application.
II. Multiple users can be logged in at a same time, either customers or workers.
III. The application will work only within the premises of the housing society resulting quick response from the network.

2.6 Software Quality Attributes

2.6.1 Availability

If the internet services gets disrupted while sending information to the server. The information can be send again.

2.6.2 Development Platforms

For Website:

XAMP v7.2.3

MySQL Database

Apache Web Server

HTML 5, CSS-3

Wordpress v4.9

JWT-Authorization

WooCommerce Multivendor Plugin

WooCommerce REST-API

For Android:

Android Studio v3.1.3

Java SE Development Kit 8 update 101

Java 8 update 181

Postman (for REST-API testing)

Insomnia (REST-API testing)

Android Studio Libraries:

com.squareup.picasso:picasso:2.71828 (for rendering Images)

com.google.code.gson:gson:2.8.0 (for handling JSON parsing)

com.android.volley:volley:1.0.0 (for handling Network Requests)

CHAPTER 3: SOFTWARE DEVELOPMENT AND PROCESS MODEL

3.1 Methodology - Agile:

We have used sprints of agile methodology during the development of web and android apps.

Firstly the web application was developed according to the requirements and improvements were made time to time recommended by the supervisors. In the initial stages the approach was totally using the plugins and layout customization tools to develop the web application, but when we started to implement the functionality we realized that changes in the plugins files and custom coding was required to deliver the functionality defined by the project scope.

Using sprints of agile methodology we were able to revert back to the initial design and make the necessary changes needed, so we could move forward with ease.

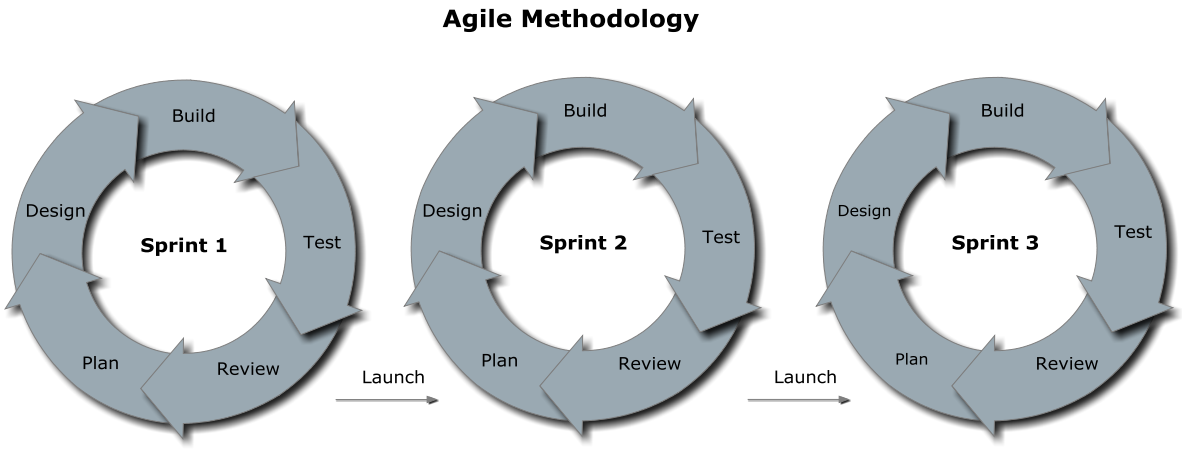


Figure 3

3.2 Team Role & Responsibilities (RACI Matrix)

Responsibility, **A**ccountability, **C**onsultant, **I**nformed.

	Taha Amin	**Areeb Vohra**	**Abdul Moiz**	**Supervisor/Co-supervisor**
Design	R	C	A	C,I
Analysis	R	R	A,C	C,I
Development	C	R,A	R,C	C,I
Testing	A	R	A,C	C,I

3.3 Requirement Phase

In the requirement phase we gathered requirements from both literature reviews and the improvements which we studied to be required in ongoing projects in this domain.

The main requirements are as follows:

i. The project is a web/android based application used for by the citizens of a housing society.
ii. Admin panel is required to create accounts and specify roles to workers and customers, manage and review order details and check for any queries asked by the users.
iii. An android application is required for the workers from which they can view order description, modify status and respond to the request from customers.
iv. Users should be able to interact from both web and android application to request any service, view order history and check status of pending orders.

3.4 Development Phase

3.4.1 Code sample of Connection URLs

```
public static String LOCALHOST_PHY = "192.168.1.100";
public static String LOCALHOST_VIR = "10.0.2.2";
public static String LOCALHOST;
public static String URL_AUTH = "http://" + All.LOCALHOST+"/fixmyhome/wp-json/jwt-
auth/v1/token";
public static String WP_JSON_URL = "http://" + All.LOCALHOST+"/fixmyhome/wp-json/wp/v2";
public static String WC_JSON_URL = "http://" + All.LOCALHOST+"/fixmyhome/wp-json/wc/v2";
public static String URL_PAGES = "/fixmyhome/wp-json/wp/v2/pages";
```

Description: In the above code sample all the necessary connections between the website and android application are made.

3.4.2 Code Sample of Retrieving Substrings

```
public static List<String> getStringList(String sourceSTR, String
startOfSearch, String endOfSearch) {
  List<String> stringList = new ArrayList<>();
  if (sourceSTR == null || sourceSTR.isEmpty()) {
   stringList.add("");
   return stringList;
  }

  Pattern p = Pattern.compile(startOfSearch);
  Matcher m = p.matcher(sourceSTR);
  while (m.find()) {
//   String temp = sourceSTR.substring( isFromStart? m.start(): m.end() );
   String temp = sourceSTR.substring(m.end());

   Pattern pp = Pattern.compile(endOfSearch);
   Matcher mm = pp.matcher(temp);

   String temp2 = "";

   if (mm.find()) {
    temp2 = temp.substring(0,
//      isTillEnd? mm.end() : mm.start()
      mm.start()
    );
   }
   stringList.add(temp2);

   if (stringList.size() < 1) { // if no match was found.
    stringList.add("");
   }
  }
  return stringList;
 }
```

Description: In the above code the functions “getStringList()” returns multiple occurrences at specific points of the source string. The specific points are provided by using the arguments, “startOfSearch” and “endOfSearch”.

3.4.3 Code Sample of Handling a Network Error

```
public static void handleVolleyError(Context context, VolleyError error) {
 error.printStackTrace();
 if (error instanceof NoConnectionError) {
  Toast.makeText(context, "No Connection Found !", Toast.LENGTH_SHORT).show();
 }
 else if (error instanceof NetworkError) {
  Toast.makeText(context, "Network Error !", Toast.LENGTH_SHORT).show();
 }
 else if (error instanceof ServerError) {
  Toast.makeText(context, "Server Error !", Toast.LENGTH_SHORT).show();
 }
 else if (error instanceof AuthFailureError) {
  Toast.makeText(context, "Authentication Failure !",
Toast.LENGTH_SHORT).show();
 }
 else if (error instanceof TimeoutError) {
  Toast.makeText(context, "Timeout !", Toast.LENGTH_LONG).show();
 }
 else if (error instanceof ParseError) {
  Toast.makeText(context, "Parse Error !", Toast.LENGTH_SHORT).show();
 }
}
```

Description: The above code responds with the type of error from a Volley network request.

3.4.4 Code Sample of Network Request

```
public class CustomRequest {
 private Map<String, String> mParams;
 private String responseString;
 private String mURL;
 private int mMETHOD;
 private Context mContext;
 private Map<String, String> mHeaders;
 private IResult mResultCallback = null;
 private StringRequest request;
 CustomRequest(IResult resultCallback, Context context, String url, int method) {
  mResultCallback = resultCallback; mContext = context; mURL = url;
  mMETHOD = method; mParams = new HashMap<>(); mHeaders = new HashMap<>(); }
 public void cancelRequest() {
  if (request != null) {
   request.cancel();
   Toast.makeText(mContext, "Request cancelled", Toast.LENGTH_SHORT).show();
  } else {
   Toast.makeText(mContext, "Error: empty request.", Toast.LENGTH_SHORT).show(); }
}
 public void executeRequest() {
  Toast.makeText(mContext, ":::" + mURL, Toast.LENGTH_SHORT).show();
  this.request = new StringRequest(this.mMETHOD,
    mURL,
    new Response.Listener<String>() {
     @Override
     public void onResponse(String responseStr) {
      responseString = responseStr;
      if (mResultCallback != null)
       mResultCallback.notifySuccess(responseStr); }},
    new Response.ErrorListener() {
     @Override
     public void onErrorResponse(VolleyError volleyError) {
      volleyError.printStackTrace();
      if (mResultCallback != null)
       mResultCallback.notifyError(volleyError);
     } }) {
   @Override
   protected Map<String, String> getParams() throws AuthFailureError {
    return mParams; }
   @Override
   public Map<String, String> getHeaders() throws AuthFailureError {
    return mHeaders; } };
  RequestQueue rQueue = Volley.newRequestQueue(this.mContext);
  rQueue.add(this.request); }
 public void setParamAndValue(String param, String value) {
  this.mParams.put(param, value); }
 public void setHeaderAndValue(String param, String value) {
  this.mHeaders.put(param, value); } }
```

Description: The above code is used to customize every Volley network request throughout the application. This custom class is also useful for notifying URLs with a Toast message whenever a request is executed.

3.5 Testing Phase

We started testing each module of the project as they came into development.

i. Responsiveness of website.
ii. Created users validation.
iii. Login validation of users and workers.
iv. Created services testing.
v. Add to cart and checkout.
vi. Billing details.
vii. Retrieval of every order through REST-API in android application.
viii. Retrieval of images in android app.
ix. Status update of orders in web/android synchronously.
x. Forgot password.

3.5.1 Black Box Testing:

In software testing, blackbox testing plays a vital role in testing of the system. The procedure of the blackbox testing is to check the performance and ensur that the functionality of the application is working fine. If it's software testing or user acceptance testing or integration testing, blackbox testing is eligible to be implemented on each level. It can certainly dominate unit testing but surely is not that much of a higher level testing.

This method attempts to find errors in the following categories:

i. Incorrect or missing functions
ii. Interface errors
iii. Behavior or performance errors
iv. Initialization and termination errors

We took all the requirements of this project to test its functionality requirements.

v. Users and workers must have login in portal by using their usernames and passwords.
vi. Only Registered users and workers can access the portal.

3.5.2 System Testing

System testing is an operation that includes system testing, trainings of users and proper working of the developed system. The person from the user side test the changes developed in the system and checks if the system has changed according to their needs. This testing phase includes the testing of system using different kinds of data. A detailed testing of data is initialized and by the use of the test data, system is tested. The errors are noticed during testing and improvements are made. Training sessions are conducted for the staff so that they can operate the system that is developed. In this phase of testing, the main objective of the testing team is to make sure that the system works precisely and is error-free.

Major types of system testing are:

i. Unit Testing
ii. Integration Testing
iii. User acceptance Testing
iv. Functional Testing

3.5.3 Unit Testing

User acceptance testing is the final process of software testing. The users of the software test's the software during user acceptance test and ensure the system

is enabled to handle the tasks required as per the requirements. For the success of any system, user acceptance is the key factor. During the testing, developers remain in contact with the users of the system so if they want to make changes that are required.

3.5.4 Integration Testing

In the integration testing phase, modules of the software are joined to perform testing as a group. This phase comes before validation testing and after unit testing. Unit tested modules are taken as an input and are grouped together so that the plans of the integration test plan as planned can be applied to the joint groups, after which when the output is delivered, system testing is prepared to be implemented by the integration system. It probably works as a systematic testing that constructs the structure of a program during the error checking initialized by conducting tests. In this phase correction is impossible because it could cause you time, money and effort.

3.5.5 Functional Testing

This phase of testing is a very important phase and is implemented within the software development that helps ensure that the system is working as per the specifications and requirements. Basically, this phase is based on the testing of the functionality that is required and specified by the user and the whole system is check at different peak to test the functionality. Functional testing is kind of a blackbox testing in which requirements of the software parts are put under test which has the bases in test cases. Required inputs are entered and outputs are examined. Functional testing is all about what the system functions.

3.5.6 User Acceptance Testing

User acceptance testing is the final process of software testing. The users of the software test's the software during user acceptance test and ensure the system is enabled to handle the tasks required as per the requirements. For the success of any system, user acceptance is the key factor. During the testing, developers remain in contact with the users of the system so if they want to make changes that are required.

3.5.7 Finalized Testing

We encountered bugs and errors that were fixed at the same time. The website/application was tested by creating users, workers, adding demo details and uploading demo services, and generating orders by multiple users requesting for multiple services.

The android application for workers was also tested by responding to generated orders, updating the status, and viewing order details.

3.6 Test Cases:

We have highlighted all the test cases, where we were having most of the bugs and errors in the application during its development to make sure that the application pass each and every scenario which could be a cause of application crash while its usage.

3.6.1 Test Case # 1

Testing the account creation for customers on the website in the MY ACCOUNT page

Preconditions	Go to the MY ACCOUNTS page and create a new account.
Actions	Enter Username, Email, and Password and click Register to create your account.
Expected Results	The system has created a new account for the user.
Tested By	Muhammad Areeb Vohra and Syed Abdul Moiz Hussain
Result	Pass

3.6.2 Test Case # 2

Browsing through the website as a customer and requesting for a service

Preconditions	Open the SERVICES page and select a worker from the available categories and request a service.
Actions	Enter time, date, contact details and submit the request.
Expected Results	The user will receive a complete summery of his/her order.
Tested By	Muhammad Areeb Vohra and Syed Abdul Moiz Hussain
Result	Pass

3.6.3 Test Case # 3

Responding to a request from the workers dashboard

Preconditions	In the workers dashboard the worker will navigate to the orders tab and view his orders list.
Actions	Worker will open a received order view the order details, contact the customer, and change status from pending to processing
Expected Results	After the customer is contacted his status will change from pending to processing
Tested By	Muhammad Areeb Vohra and Muhammad Taha Amin
Result	Pass

3.6.4 Test Case # 4

Registering a new worker and adding his services for the customers to view

Preconditions	The worker will fill out a form adding his complete details and skills.
Actions	The admin will receive the form and after analyzing the individuals details admin will create an account for the worker and publish his service.
Expected Results	The worker can open his new dashboard and his service is published on the website
Tested By	Muhammad Areeb Vohra and Muhammad Taha Amin
Result	Pass

3.6.5 Test Case # 5

Requesting for a service on the Customer Mobile Application

Preconditions	Open the SERVICES screen and select a worker from the available categories and request a service.
Actions	Enter time, date, contact details and submit the request.
Expected Results	The user will receive a complete summery of his/her order.
Tested By	Muhammad Areeb Vohra and Syed Abdul Moiz Hussain
Result	Pass

3.6.6 Test Case # 6

Responding to a request from the workers Mobile Application

Preconditions	Open my orders from the menu and view order details for the requested service
Actions	View the order details, contact the customer, and change status from pending to processing
Expected Results	After the customer is contacted his status will change from pending to processing
Tested By	Muhammad Areeb Vohra
Result	Pass

CHAPTER 4: PROJECT PLANNING

4.1 First Evaluation Gantt chart

Task Name	Duration	Start	Finish
FYP First Evaluation	**150 days**	**Tue 8/1/17**	**Wed 1/31/18**
Requirement Analysis	**40 days**	**Tue 8/1/17**	**Tue 10/10/17**
Elicitation of Requirement		Tue 8/1/17	Tue 10/10/17
Analysis of Requirement		Tue 8/1/17	Tue 10/10/17
Requirement Validation		Tue 8/1/17	Tue 10/10/17
Designing	**30 days**	**Mon 9/11/17**	**Fri 12/1/17**
Web Application Design		Wed 9/27/17	Sun 11/19/17
Implementation	**40 days**	**Wed 11/1/17**	**Tue 12/26/17**
Web Development		Wed 11/1/17	Tue 12/26/17
Testing	**50 days**	**Mon 9/11/17**	**Wed 1/31/18**
Black Box Testing		Wed 11/1/17	Wed 1/31/18
White Box Testing		Wed 9/27/17	Wed 1/31/18

Timeline

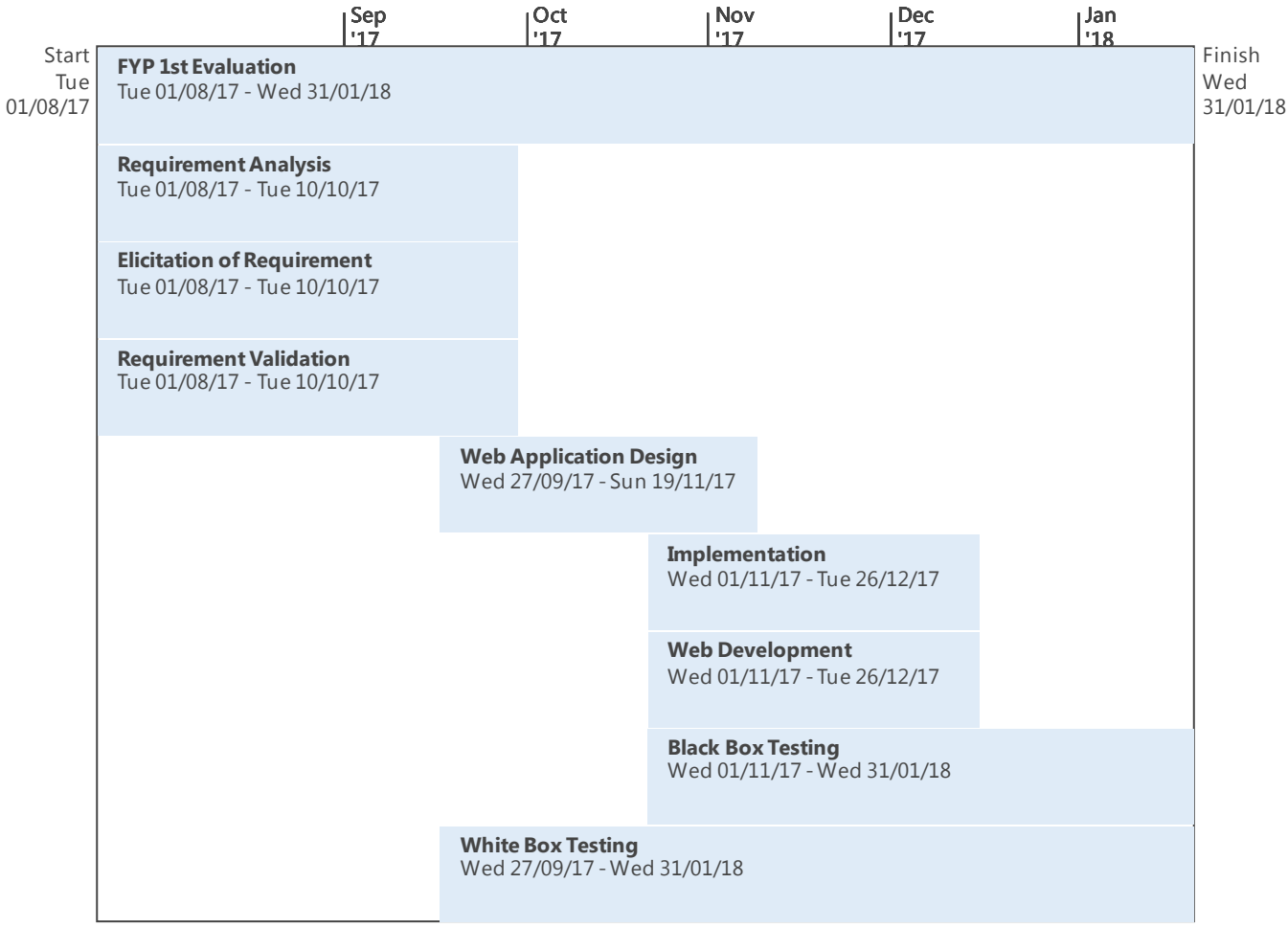


Figure 4

4.2 Final Evaluation Gantt chart

Task Name	Duration	Start	Finish
FYP Second Evaluation	**131 days**	**Mon 01/01/18**	**Sat 30/06/18**
Requirement Analysis	**51 days**	**Mon 01/01/18**	**Mon 12/03/18**
Requirement Elicitation	51 days	Mon 01/01/18	Mon 12/03/18
Analysis of Requirement	51 days	Mon 01/01/18	Mon 12/03/18
Requirement Validation	38 days	Thu 18/01/18	Mon 12/03/18
Designing and Development	**66 days**	**Thu 01/03/18**	**Thu 31/05/18**
Android Designing	57 days	Thu 01/03/18	Sun 20/05/18
Android Development	58 days	Tue 13/03/18	Thu 31/05/18
Testing	**40 days**	**Tue 08/05/18**	**Sat 30/06/18**
Black Box Testing	39 days	Tue 08/05/18	Sat 30/06/18
White Box Testing	39 days	Tue 08/05/18	Sat 30/06/18

Timeline

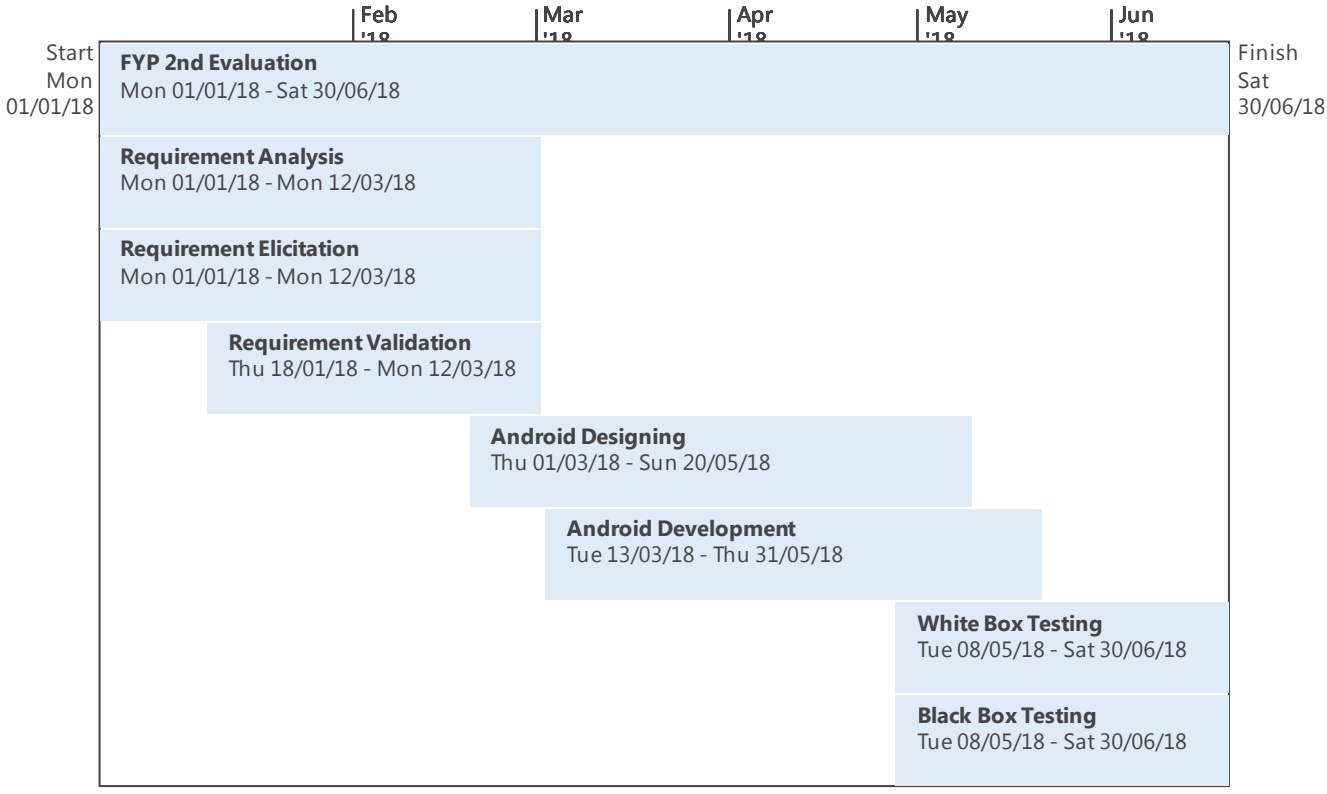


Figure 5

4.3 Website GUI Screenshots

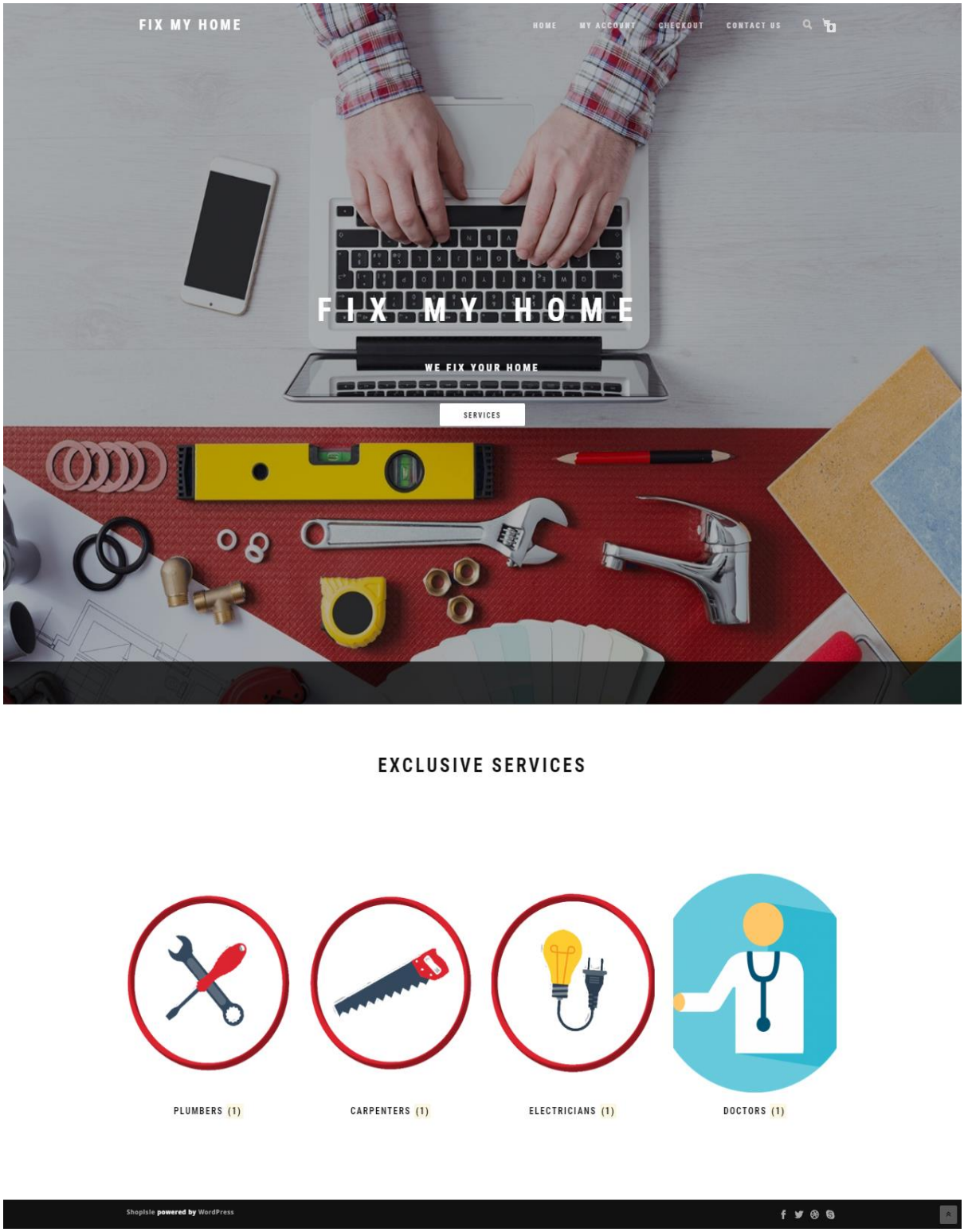


Figure 6

Description: This is the main page for the website, here you can check for available categories of services.

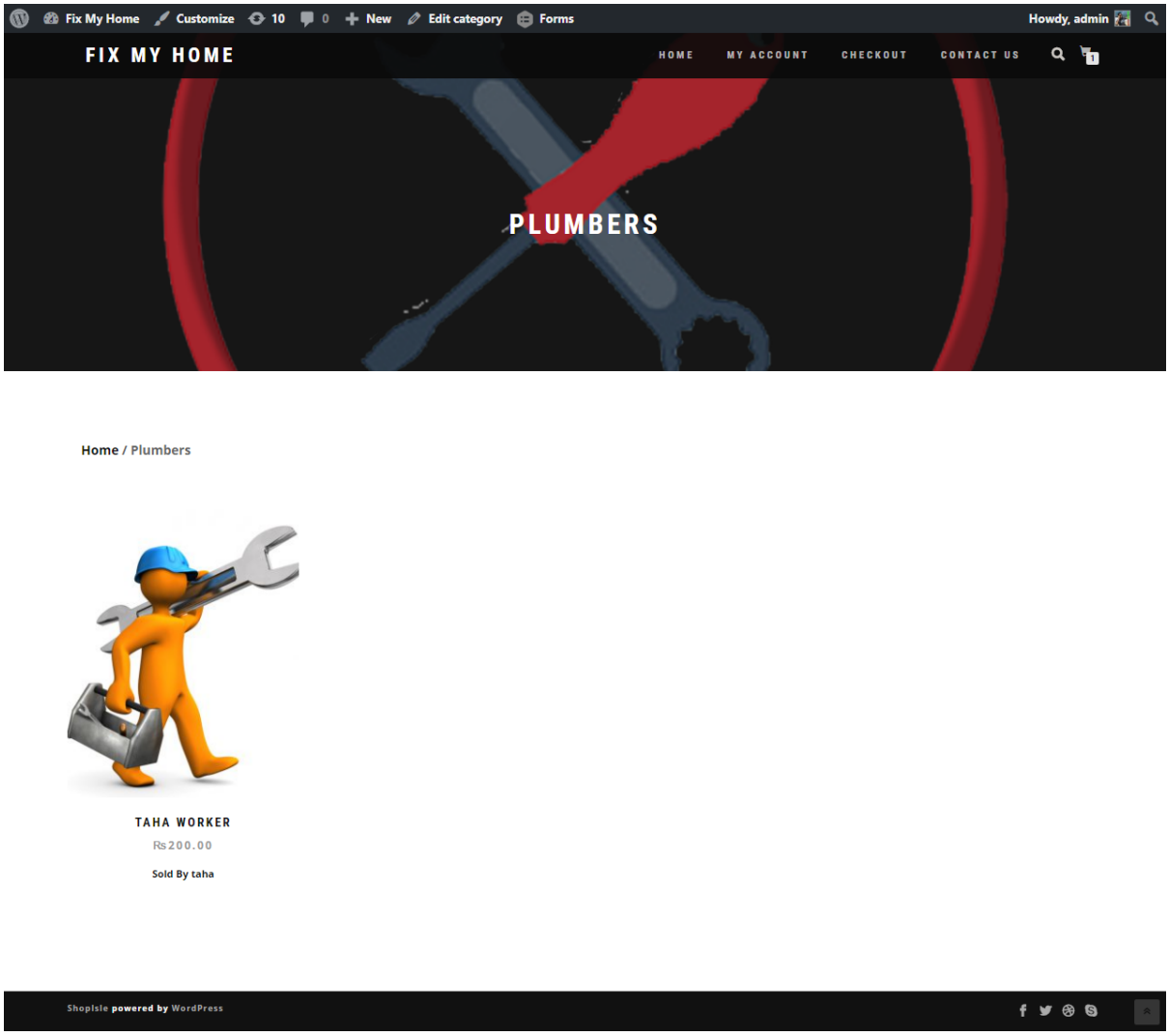


Figure 7

Description: After selecting a category from the services you can select a service available worker in that category.

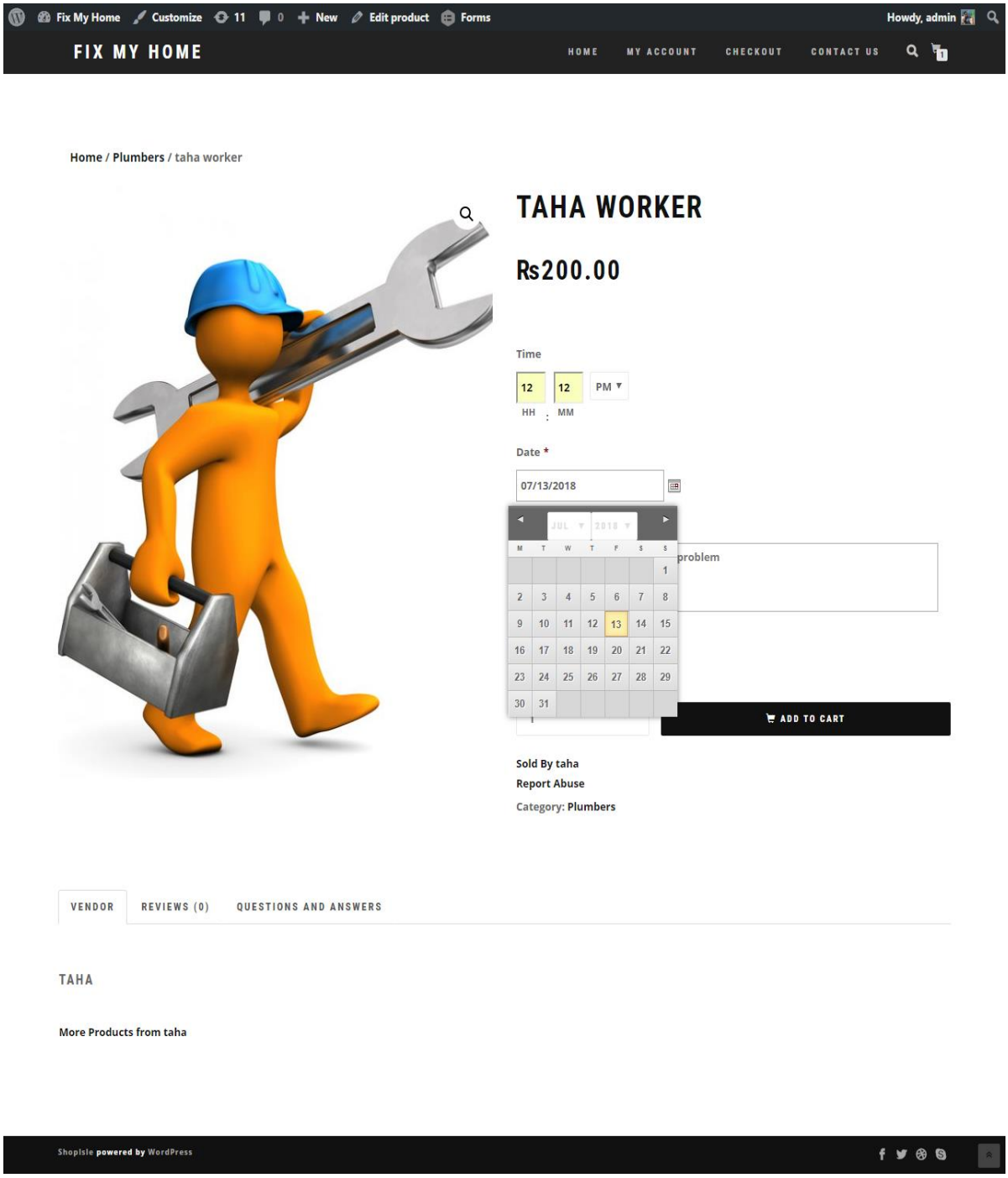


Figure 8

Description: After selection a worker you can select a time and date at of your choosing, at which you want to the worker to come at your home. Also you will send a detailed description telling about the problem in your house.

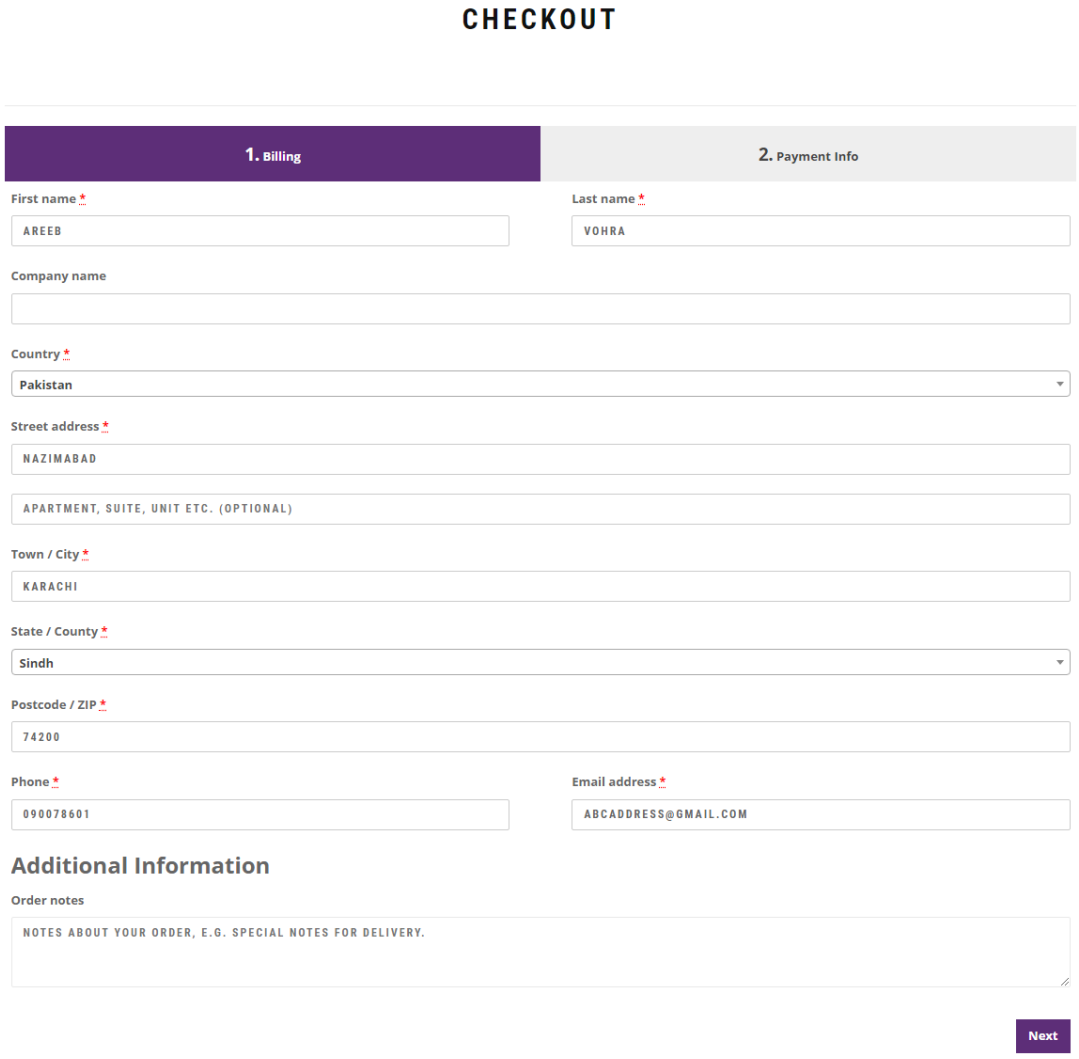


Figure 9

Description: after selection of worker you will proceed to the checkout page where you will enter your contact details.

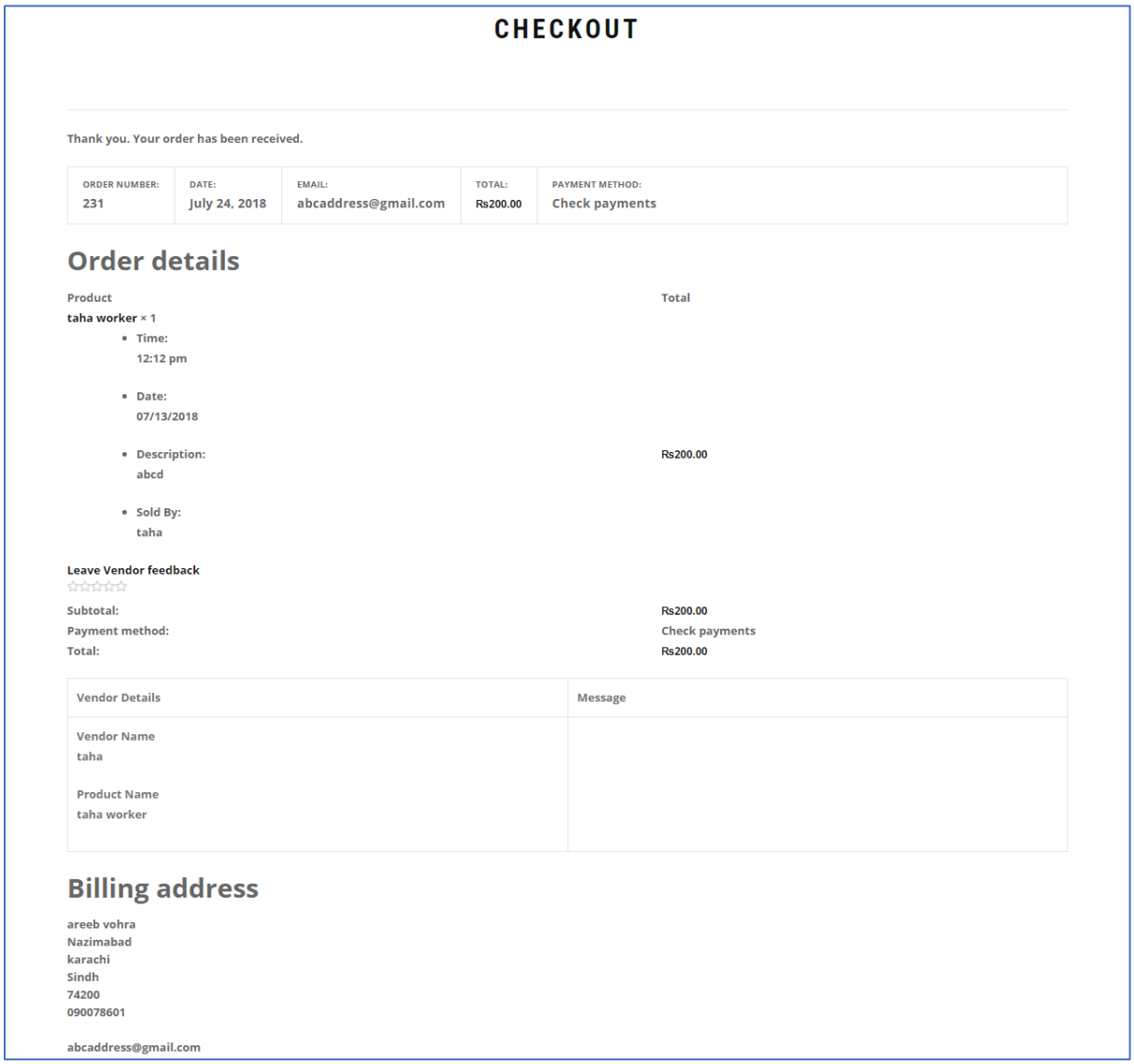


Figure 10

Description: After submitting your contact details you will get a summary of your order.

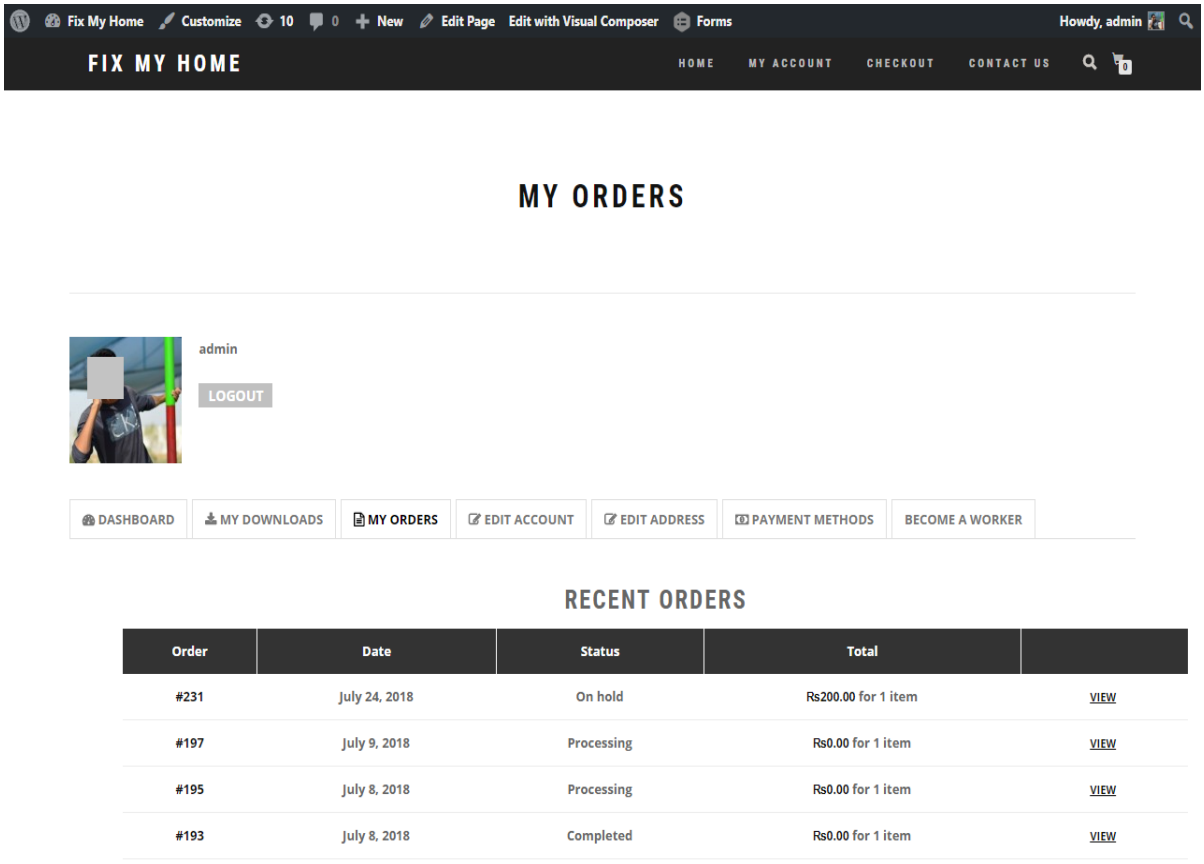


Figure 11

Description: You can check your order status and previous orders in the MY ACCOUNT → MY ORDERS section.

4.4 Admin Dashboard GUI

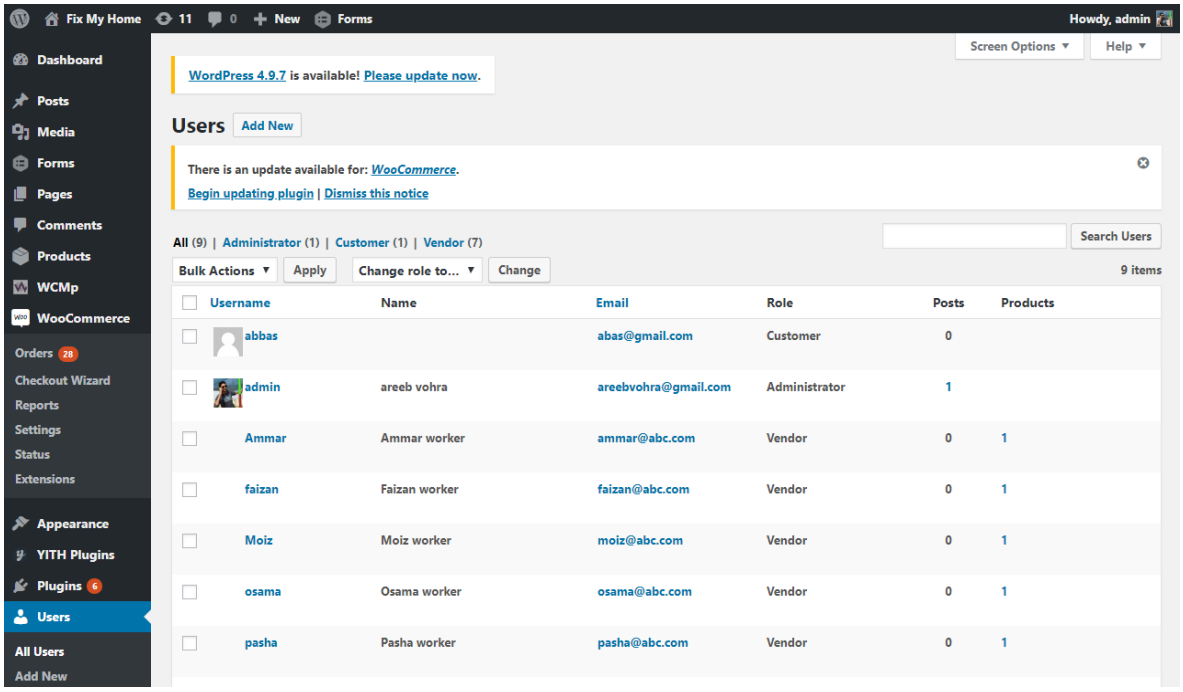


Figure 12

Description: Here is the Admin Dashboard where the admin can view all the Users and their roles connected to the site weather customers or workers.

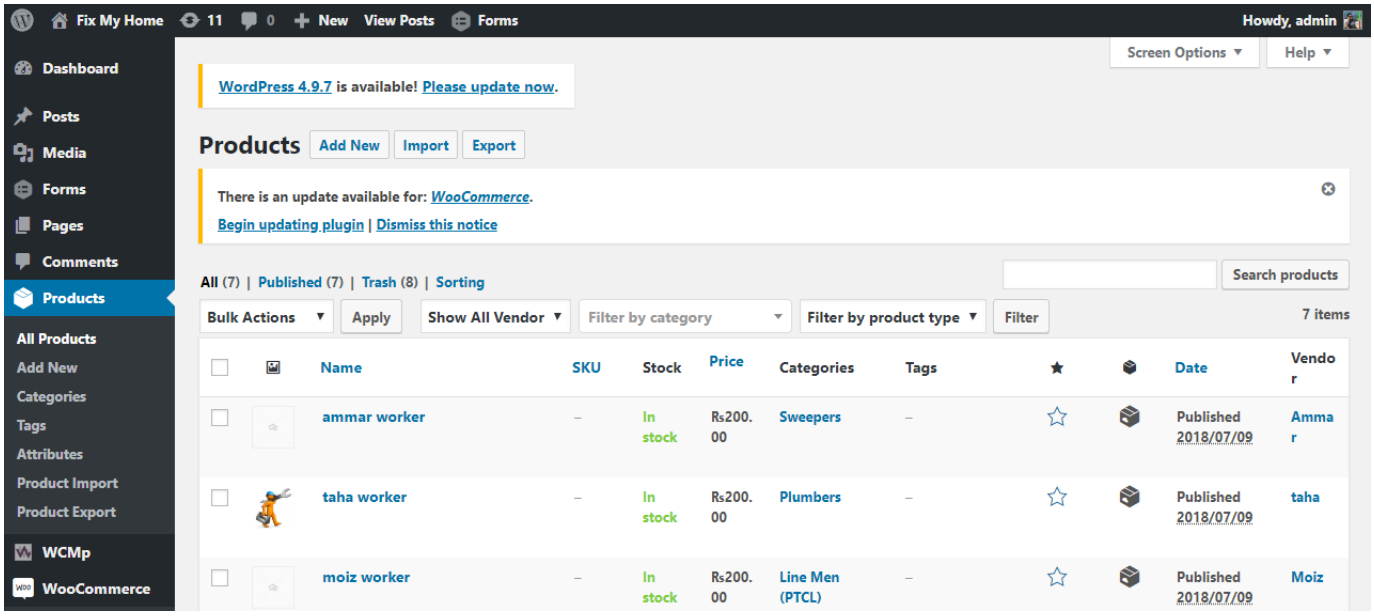


Figure 13

Description: Here the admin the view all the current workers active on the site.

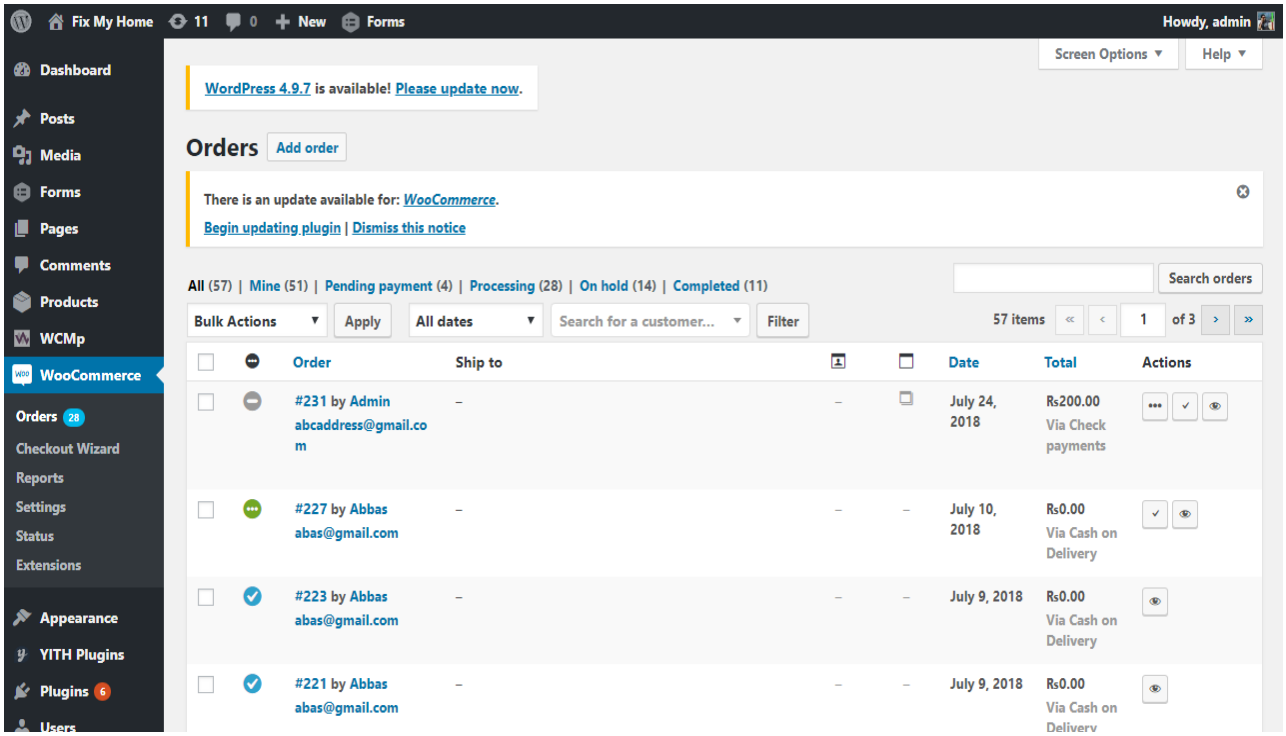


Figure 14

Description: Here the admin can view all the orders and check their status and modify them.

4.5 Android Application GUI (for customers)

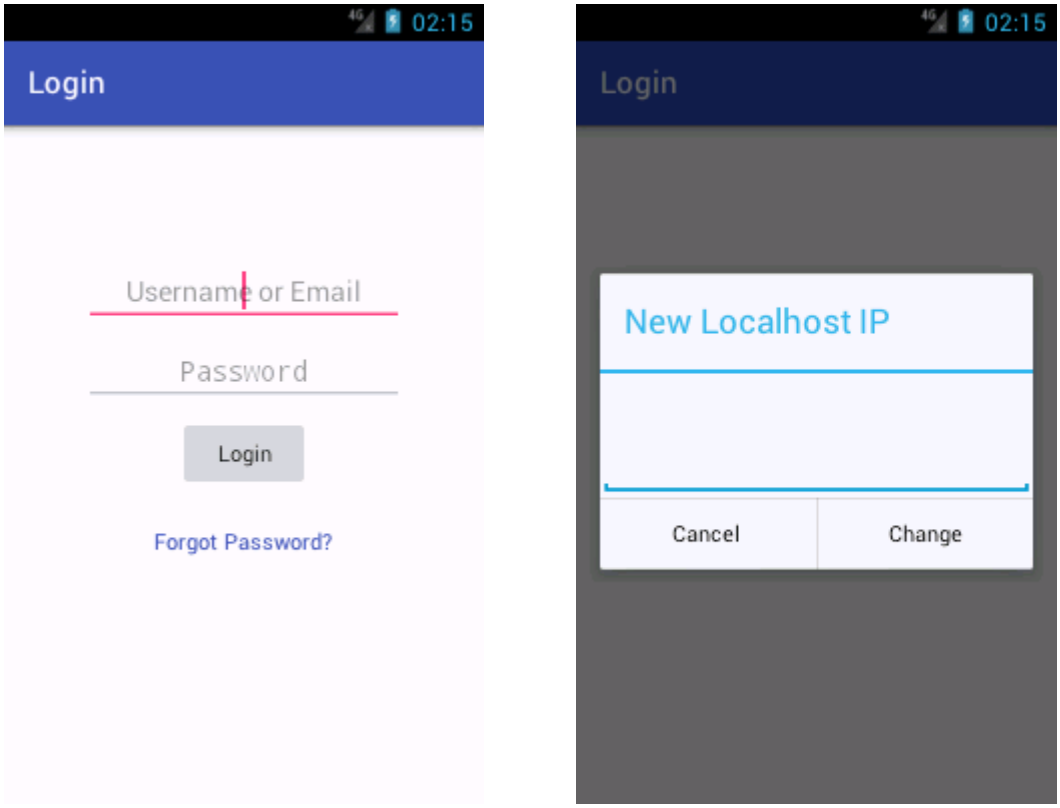


Figure 15

Description: Here the user will login to his/her account. Since the project is currently running on localhost the user has to enter his router IP for the application to connect with the localhost server, the IP section will not be necessary when the project is on a live domain.

Figure 16

Description: After login here is the main view of the mobile application for the customers. The user can click the services button to check for available services.

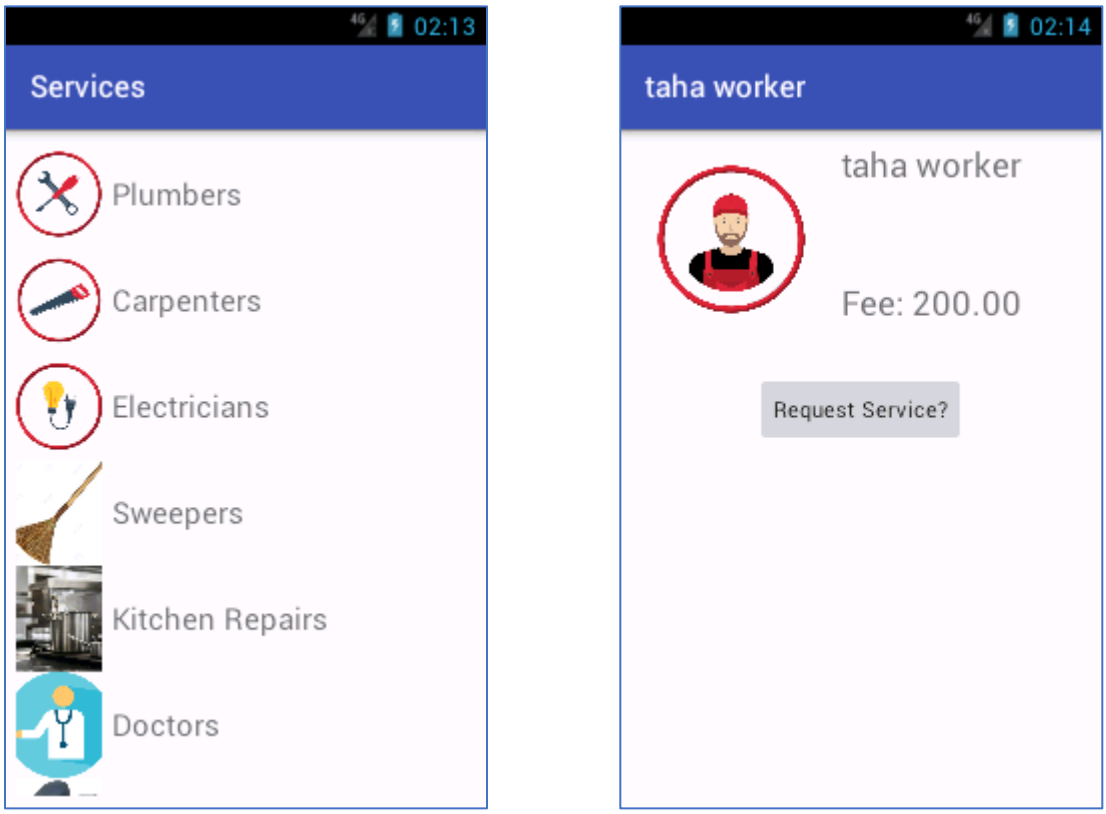


Figure 17

Description: on the left screen the user can view all the available service. On the right screen the user can view an individual worker and request for his service.

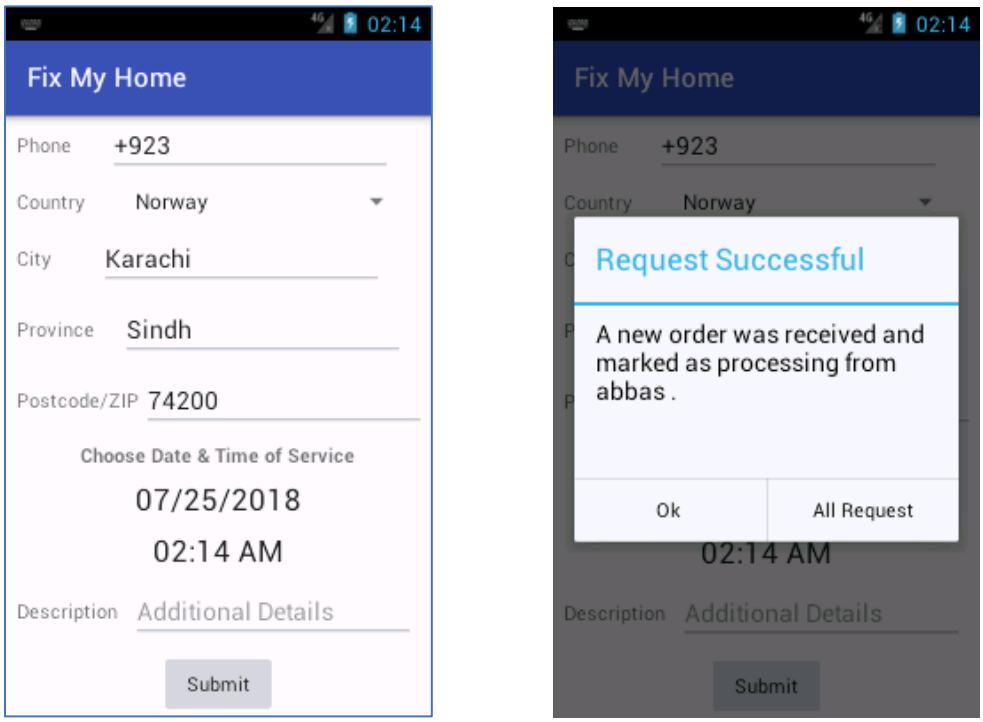


Figure 18

Description: Here the user will enter his contact details and the user can also select date and time on which he would like worker to come at the house, uthe user will also enter description about the problem he would to fix.

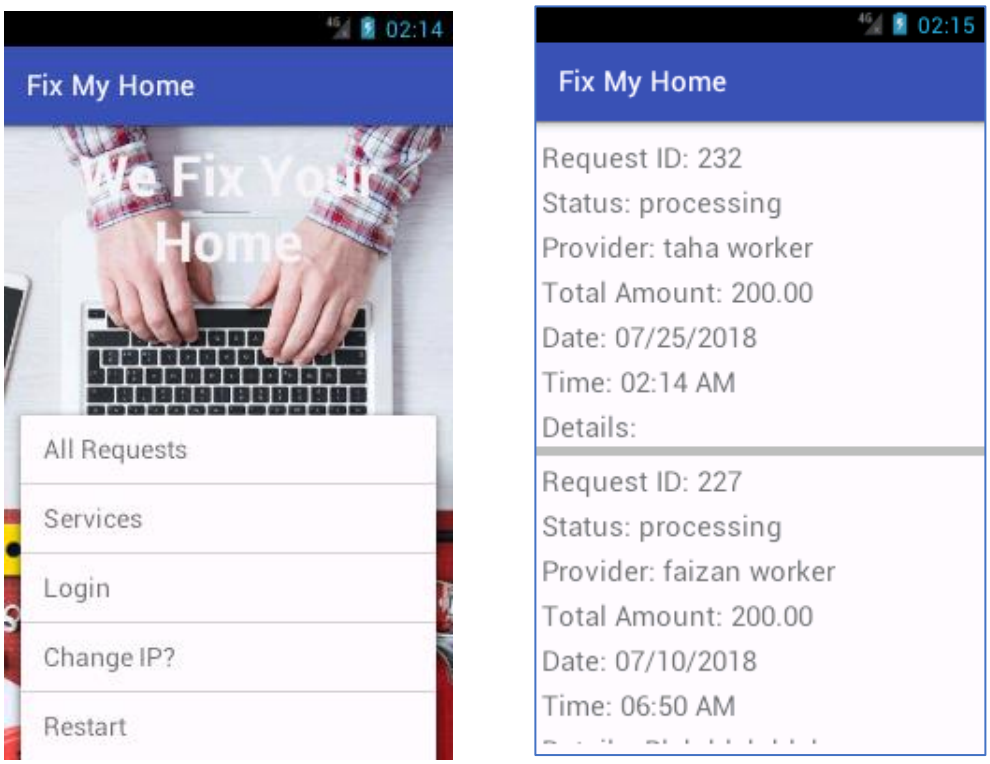


Figure 19

Description: Here the user can open the menu and he/she can view his order in the All Requests section. The user can also view previous order.

4.6 Mobile Application GUI (for Workers)

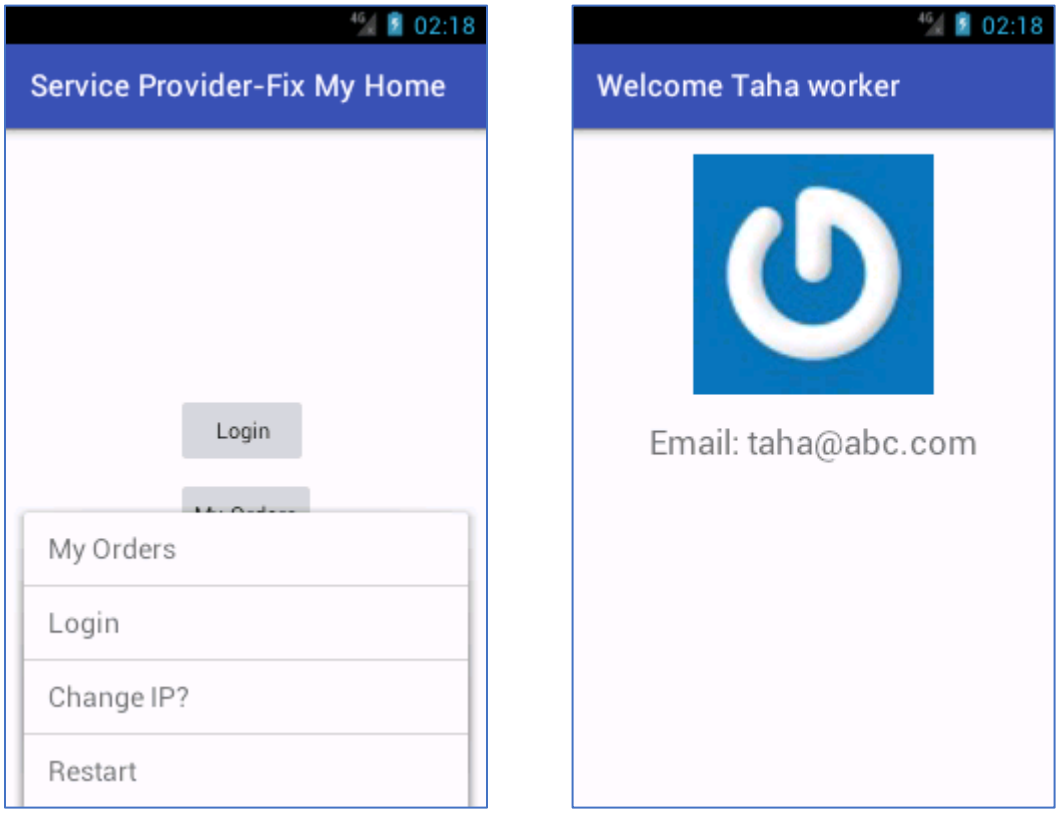
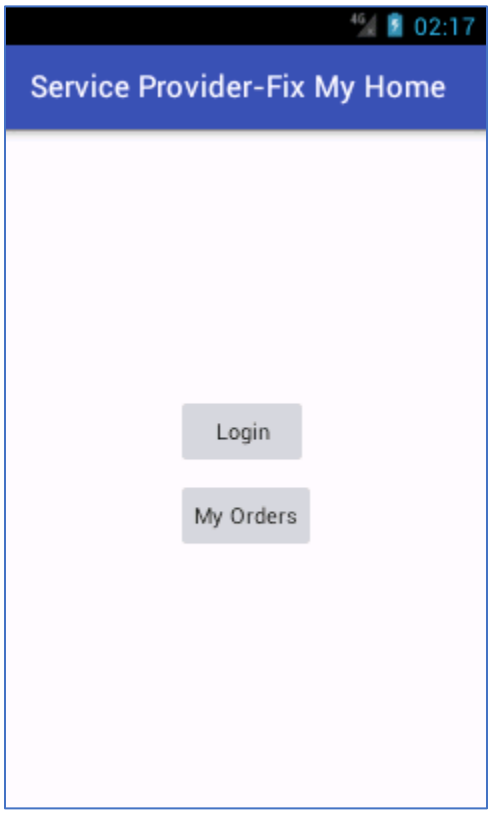
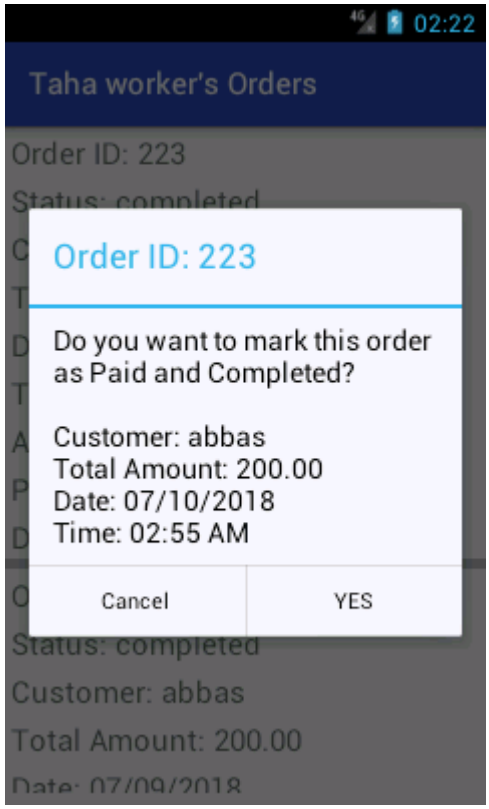


Figure 20

Description: Here is the login screen and welcome screen for the worker. The worker first has to login to his account so that he can view his orders and requests.





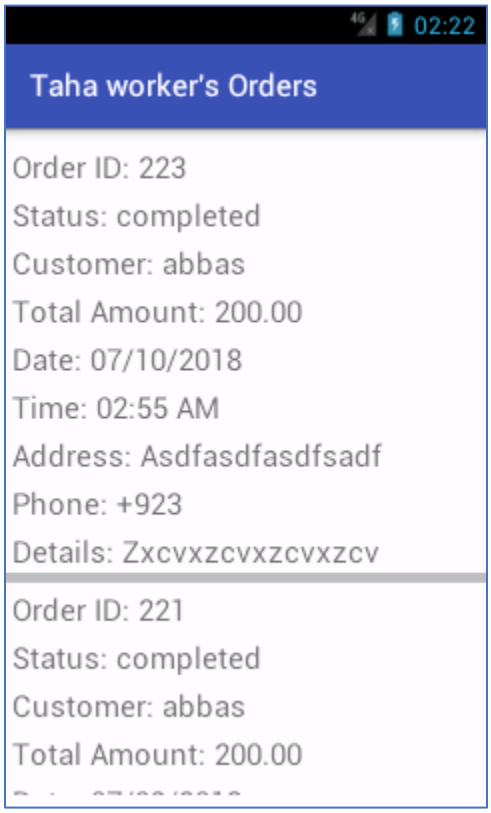


Figure 21

Description: After login the worker check his orders received. He can also change status of his order from pending to complete.

CHAPTER 5: PROJECT DIAGRAMS

5.1 Use Case Diagram

This diagram is used to represent the interaction between system and user. User in any case involved, the relationship between user and system will be showed. Use case diagram helps us to determine different types of user and how they can interact with the system which helps generate different use cases.

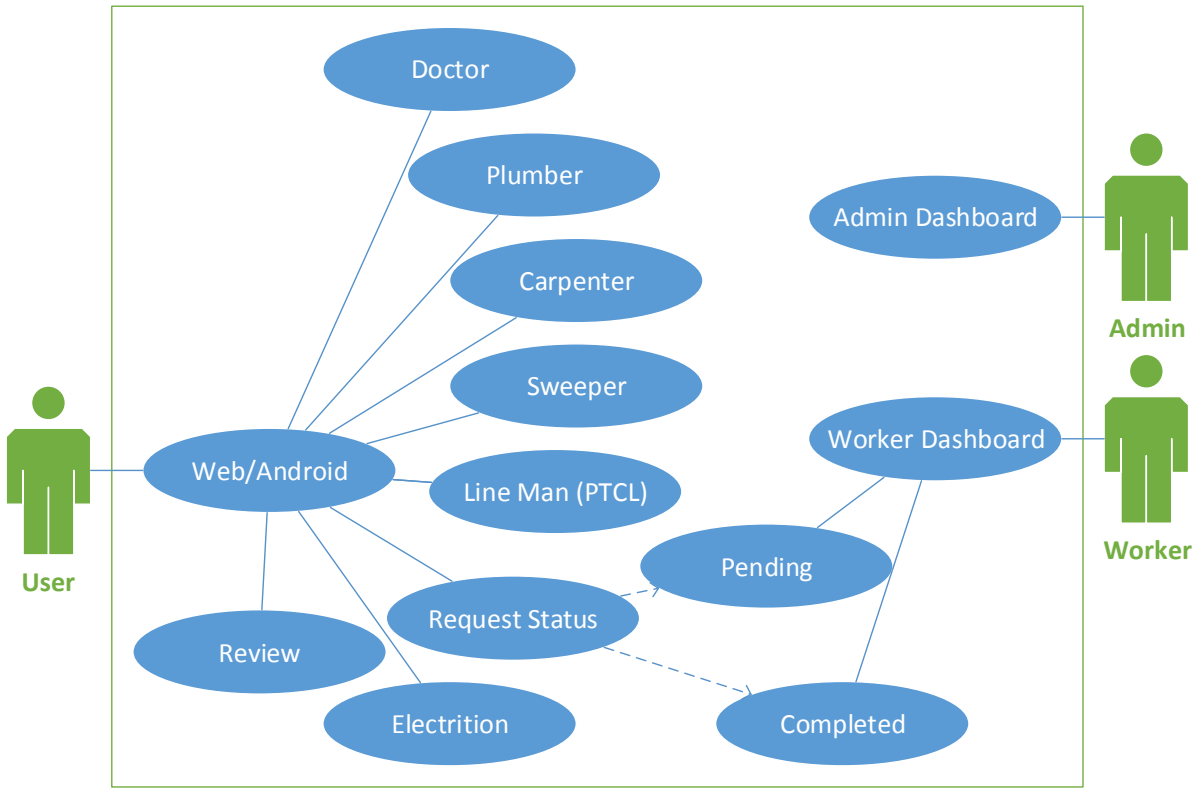


Figure 22

5.3 System Block Diagram

A block diagram is all about functions or principal parts and they are all interconnected through the lines to represent the workflow and relationship between the blocks. Block diagram is usually made to show the block parts of the project in hardware design, software design and process flow diagrams. The structure of the block diagram helps you to understand the important components and working relationships.

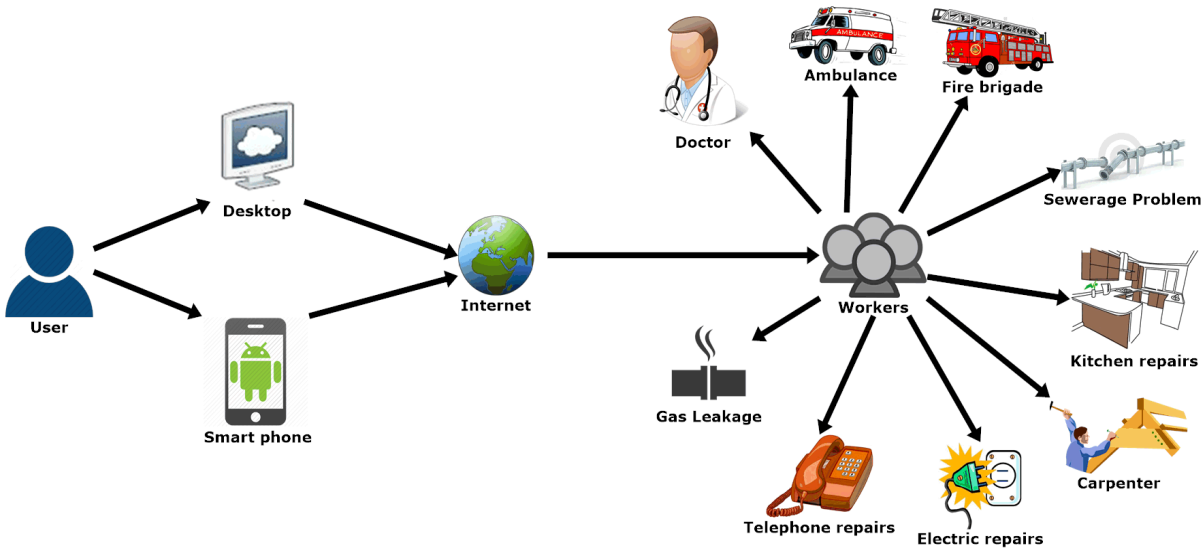


Figure 23

Description: Users are able to select any service by using either website or android application. Requests to every service will be responded by individual workers.

5.2 Activity Diagram

These diagrams are the workflows of stepwise activities, iteration, concurrency, action of support for choice and all these workflows are graphically represented. In UML, computational and organizational both processes are modelled by activity diagram. Activity diagram is all about the flow of control.

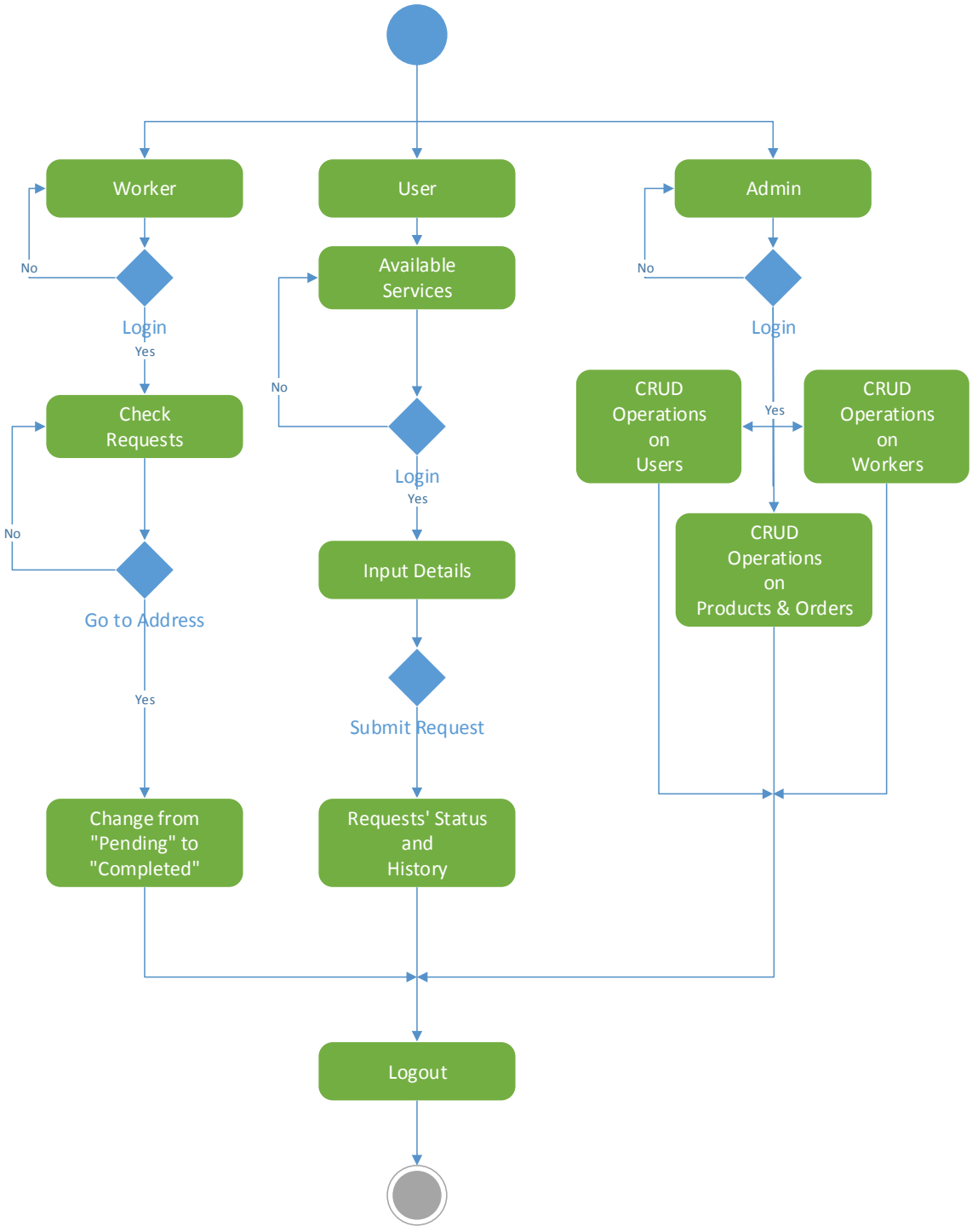


Figure 24

5.4 System Architecture

It is a conceptual model that is based on structure, behavior and to showcase the different angles of the system. It is a representation of the system and formal description that is organized in a way that describes the reasoning of the behaviors and the structures of the system. Architecture basically represents the backbone of the system that is need to be developed. This model describes the concept of structure and behavior of the system, either is it proposed or existing. System Architecture model consist of technical framework, end-user requirements and a list of system components maybe hardware or software.

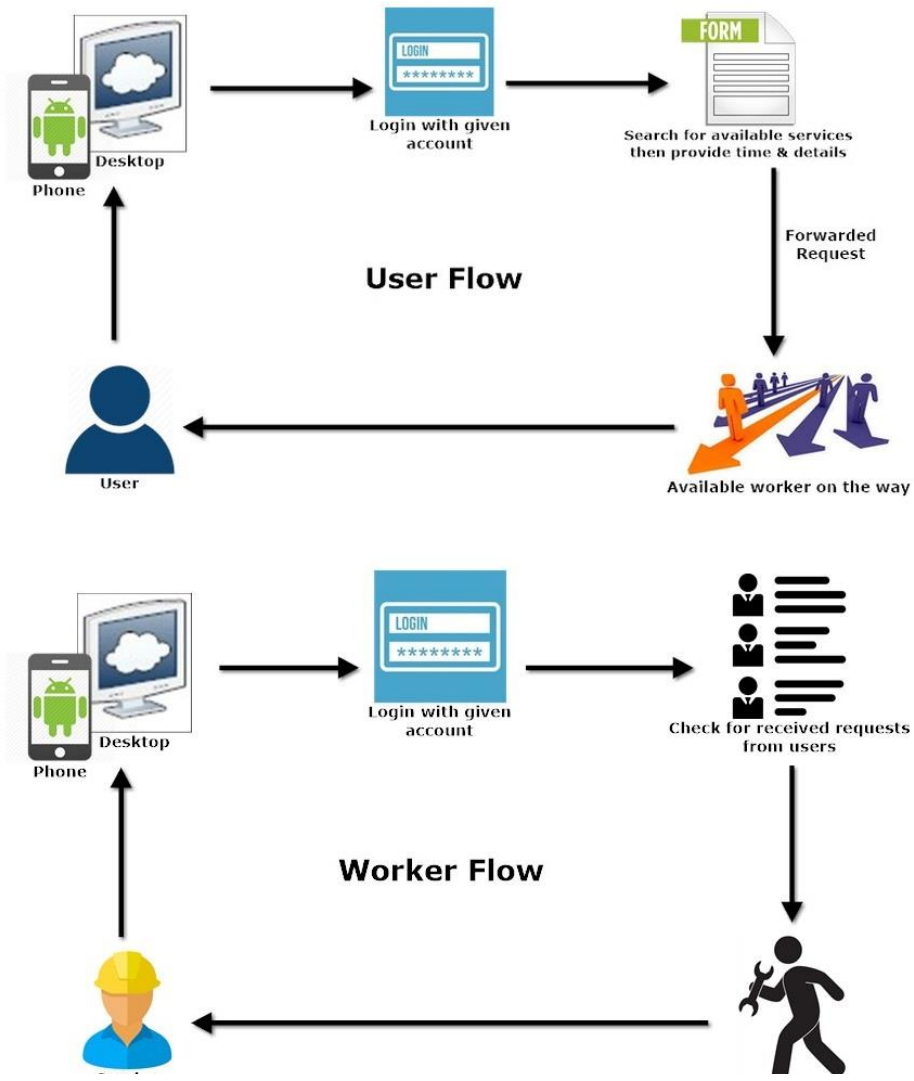


Figure 25

CHAPTER 6: TOOLS AND TECHNIQUES

6.1 Project Techniques

For the Development of **Services Management System for Housing Society,** the tools and techniques we used are as follows:

6.1.1 CSS (Cascading Style Sheet)

We have used CSS3 for Design and it is used to create attractive pages of **Services Management System for Housing Society.** We have used CSS3 in Signup, Login, Front-end and Back-end Pages.

6.1.2 Smart Draw

Smart Draw is a visual processor used to create flowcharts. We have used Smart Draw to create Use-Case Diagram, Data Flow, Sequence, and DFD's Activity ERD.

6.1.3 PHP

We have used PHP to create MYSQL database connection, Form Validation, Session Fetch Data.

6.1.4 Java Script

We have used Java Script for create ON Click function, alert function, delete data function, show and hide function.

6.1.5 HTML

We have used HTML to create Structure of Class Management System such as Division, Header, Footer, and Navigation bar Side Bar Tables.

6.1.6 WordPress

We have used WordPress platform for the development of our website, WordPress offers REST-API through which Android application connects to the localhost server using JSON parsing.

6.2 Project Tools

The following hardware and software so far, we have used in the development of our project **Services Management System of Housing Society** are discussed below.

I. PC, Laptops
II. Web Browser
III. MySQL
IV. XAMPP
V. Android Studio
VI. WordPress
VII. Android Phone
VIII. Router
IX. Visual Studio Code
X. Microsoft Office
XI. Postman (for REST-API testing)
XII. Insomnia (REST-API testing)

6.2.1 Web-Browser

We have used web browser like GOOGLE CHROME and Edge for viewing of **Services Management System for Housing Society** portal screen.

6.2.2 MySQL

Since WordPress offer automatic creation of database so we only used MySQL on PHPMYADMIN to view the database files and navigate to required tables.

6.2.3 XAMPP

We have used XAMPP control panel which offers Apache Web Server for the deployment of our website on the localhost. The Server was also used for the JSON APIs.

6.3 MS-Project

We have used MS-Project to create FYP Timeline.

6.4 MS-Office

6.4.1 MS-Word

We have used MS Word to Create FYP Reports like Proposal, First evaluation, and final evaluation.

6.4.2 MS-PowerPoint

We have used MS PowerPoint to create Presentations like Proposal, First evaluation, and final evaluation.

CHAPTER 7: CONCLUSION AND FUTURE WORK

7.1 Limitations

I. The system is deployed on localhost XAMP Server so the Android Application and Desktop machine has to be connected to the same router in order for the system to work in sync.
II. The Application is limited to Android platform.
III. Worker has to be registered by admin.

7.2 Conclusion

We developed this project **Services Management System for Housing Society** because there was no available system for members of a housing society, although projects are available on a large scale but none of them fulfilled the requirements of a society in closed compounds. Through this project members of a housing society will get their household problems solved without have to step out of the house.

7.3 Future Work

I. SMS Notifications
II. E-mail Notifications
III. Push Notifications

In future we will add these features in our project to make our project more effective. Through Push notifications we can engage users more than a normal usage. The engaging of users will also be done thorough SMS and E-mail notifications.

APPENDICES

Appendix A: Connection Files

All.java

```
package com.example.abdul.servicesmanagementsystem;

import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.NetworkError;
import com.android.volley.NoConnectionError;
import com.android.volley.ParseError;
import com.android.volley.ServerError;
import com.android.volley.TimeoutError;
import com.android.volley.VolleyError;

import java.io.UnsupportedEncodingException;
import java.util.ArrayList;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 * Created by Abdul on 3/24/2018.
 */

public class All {
  public static String LOCALHOST_PHY = "192.168.1.100";
  public static String LOCALHOST_VIR = "10.0.2.2";
  public static String LOCALHOST;
  public static String URL_AUTH = "http://" + All.LOCALHOST+"/fixmyhome/wp-
json/jwt-auth/v1/token";
  public static String WP_JSON_URL = "http://" + All.LOCALHOST+"/fixmyhome/wp-
json/wp/v2";
  public static String WC_JSON_URL = "http://" + All.LOCALHOST+"/fixmyhome/wp-
json/wc/v2";
  public static String URL_PAGES = "/fixmyhome/wp-json/wp/v2/pages";
  public static String SERVICES_STRING = "SERVICES_STRING";
```

```
  public static String TOKEN = "TOKEN";
  static String LOADING_MSG = "Please Wait";
  static String BASIC_TOKEN = "BASIC_TOKEN";
  static String CATEGORIZED_PRODUCTS = "CATEGORIZED_PRODUCTS";
  static String SELECTED_CATEGORY_NICE_NAME = "SELECTED_CATEGORY_NICE_NAME";
  static String SELECTED_CATEGORY = "SELECTED_CATEGORY";
  static String SELECTED_PRODUCT = "SELECTED_PRODUCT";
  static String SELECTED_CATEGORY_LIST = "SELECTED_CATEGORY_LIST";
  static String WORKER_ID = "WORKER_ID";
  static String WORKER_NAME = "WORKER_NAME";
  static String USER_PIC = "USER_PIC";
  static String USER_NAME = "USER_NAME";
  static String USER_ID = "USER_ID";
  static String USER_EMAIL = "USER_EMAIL";
  static String DELIMITER = "DELIMITER";

  public static List<String> getStringList(String sourceSTR, String
startOfSearch, String endOfSearch) {
    List<String> stringList = new ArrayList<>();
    if (sourceSTR == null || sourceSTR.isEmpty()) {
      stringList.add("");
      return stringList;
    }

    Pattern p = Pattern.compile(startOfSearch);
    Matcher m = p.matcher(sourceSTR);
    while (m.find()) {
//      String temp = sourceSTR.substring( isFromStart? m.start(): m.end() );
      String temp = sourceSTR.substring(m.end());

      Pattern pp = Pattern.compile(endOfSearch);
      Matcher mm = pp.matcher(temp);

      String temp2 = "";

      if (mm.find()) {
        temp2 = temp.substring(0,
//            isTillEnd? mm.end() : mm.start()
            mm.start()
        );
      }
      stringList.add(temp2);

      if (stringList.size() < 1) { // if no match was found.
        stringList.add("");
      }
    }
    return stringList;
  }
```

```
  public static void setSharedSTR(String key, String value, Context context) {
    SharedPreferences sharedPreferences =
PreferenceManager.getDefaultSharedPreferences(context);
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putString(key, value).apply();
  }

  public static String getSharedSTR(String key, Context context) {
    SharedPreferences sharedPreferences =
PreferenceManager.getDefaultSharedPreferences(context);
//    Toast.makeText(context,sharedPreferences.getStringList(All.TOKEN, ""),
//        Toast.LENGTH_SHORT).show();
    return sharedPreferences.getString(key, "");
  }

  public static void handleVolleyError(Context context, VolleyError error) {
    error.printStackTrace();
    if (error instanceof NoConnectionError) {
      Toast.makeText(context, "No Connection Found !",
Toast.LENGTH_SHORT).show();
    }
    else if (error instanceof NetworkError) {
      Toast.makeText(context, "Network Error !", Toast.LENGTH_SHORT).show();
    }
    else if (error instanceof ServerError) {
      Toast.makeText(context, "Server Error !", Toast.LENGTH_SHORT).show();
    }
    else if (error instanceof AuthFailureError) {
      Toast.makeText(context, "Authentication Failure !",
Toast.LENGTH_SHORT).show();
    }
    else if (error instanceof TimeoutError) {
      Toast.makeText(context, "Timeout !", Toast.LENGTH_LONG).show();
    }
    else if (error instanceof ParseError) {
      Toast.makeText(context, "Parse Error !", Toast.LENGTH_SHORT).show();
    }
  }

  public static void handleNormalResponse (Context context, VolleyError error)
{
    if (error.networkResponse == null) {
      return;
    }

    try {
      String response = new String(error.networkResponse.data, "UTF8");
      String msg = All.getStringList(response, "<h1>", "</h1>").get(0);
```

```
      Toast.makeText(context, Integer.toString(
          error.networkResponse.statusCode
      ) + "\n" + msg, Toast.LENGTH_LONG).show();
    } catch (UnsupportedEncodingException uee) {
      uee.printStackTrace();
    }
  }
  public static void setLOCALHOST(boolean isVirtual) {
    LOCALHOST = isVirtual? LOCALHOST_VIR: LOCALHOST_PHY;
  }
  public static void changeLocalhost(Context context) {
    final EditText input = new EditText(context);
    LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(
        LinearLayout.LayoutParams.MATCH_PARENT,
        LinearLayout.LayoutParams.MATCH_PARENT);
    input.setLayoutParams(lp);
    AlertDialog alertDialog = new AlertDialog.Builder(context).create();
    alertDialog.setView(input);
    alertDialog.setTitle("New Localhost IP");
    alertDialog.setMessage("");
    alertDialog.setButton(AlertDialog.BUTTON_POSITIVE, "Change",
        new DialogInterface.OnClickListener() {
          public void onClick(DialogInterface dialog, int which) {
            All.LOCALHOST = input.getText().toString().trim();
            dialog.dismiss();
          }
        });
    alertDialog.setButton(AlertDialog.BUTTON_NEGATIVE, "Cancel",
        new DialogInterface.OnClickListener() {
          public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
          }
        });
    alertDialog.show();  }
  public static boolean checkLoginStatus(Context context) {
    String Token = All.getSharedSTR(All.TOKEN, context);
    String Username = All.getSharedSTR(All.USER_NAME, context);
    if (Token.isEmpty() || Username.isEmpty()) {
      Toast.makeText(context, "Session not found, please Login first.",
Toast.LENGTH_LONG).show();
      return false;
    }
    else
      return true;
  } }
```

AllRequests.java

```
package com.example.abdul.servicesmanagementsystem;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.DefaultRetryPolicy;
import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import com.google.gson.Gson;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class AllRequests extends BaseActivity {
  ProgressDialog progressDialog;
  Context context;
  private RecyclerView RVListAllCategories;
  private RVAAllRequests adapter;
  private RecyclerView.LayoutManager layoutManager;
  private List<AllRequestsData> dataList = new ArrayList<>();
  List<String> orderID = new ArrayList<String>();
  List<String> orderStatus = new ArrayList<String>();
  List<String> orderWorkerName = new ArrayList<String>();
  List<String> orderTotal = new ArrayList<String>();
  List<String> orderDate = new ArrayList<String>();
  List<String> orderTime = new ArrayList<String>();
  List<String> orderDescription = new ArrayList<String>();

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_all_requests);
    context = AllRequests.this;

    if ( !All.checkLoginStatus(context) ) {
```

```
      finish();
      startActivity(new Intent(context, Login.class));
      return;
    }

    setRecyclerView();
    progressDialog = new ProgressDialog(this);
    progressDialog.setMessage(All.LOADING_MSG);

    final String basicToken = All.getSharedSTR(All.BASIC_TOKEN, context);
    String userEmail = All.getSharedSTR(All.USER_EMAIL, context);
    String userID = All.getSharedSTR(All.USER_ID, context);

    StringRequest stringRequest = new StringRequest(
        Request.Method.GET,
        "http://" + All.LOCALHOST + "/fixmyhome/wp-json/wc/v2/orders?search="
+ userEmail + "&customer=" + userID,
        new Response.Listener<String>() {
          @Override
          public void onResponse(String response) {
            progressDialog.dismiss();
            Log.v("Allresuests:", response);
            if ( response.length() > 10)
              setResponse(response);
            else
              Toast.makeText(context, response, Toast.LENGTH_LONG).show();
          }
        },
        new Response.ErrorListener() {
          @Override
          public void onErrorResponse(VolleyError error) {
            All.handleVolleyError(context, error);
            progressDialog.dismiss();
          }
        }
    ) {
      @Override
      public Map<String, String> getHeaders() throws AuthFailureError {
        Map<String, String> headers = new HashMap<String, String>();
        headers.put("Authorization", "Bearer " + basicToken);
        return headers;
      }
    };
    int MY_SOCKET_TIMEOUT_MS = 30000;
    stringRequest.setRetryPolicy(new DefaultRetryPolicy(
        MY_SOCKET_TIMEOUT_MS,
        DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
        DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
    progressDialog.show();
```

```
    Volley.newRequestQueue(context).add(stringRequest);
  }

  private void setResponse(String response) {
    if ( !response.isEmpty() ) {
      Gson gson = new Gson();
      List orderObjects = (List) gson.fromJson(response, List.class);
//      String[] postContent = new String[orderObjects.size()];
      for (int i = 0; i < orderObjects.size(); ++i) {
        Map<String, Object> anObject = (Map<String, Object>)
orderObjects.get(i);

        String id = (Double)anObject.get("id") + "";
        id = id.substring(0, id.indexOf('.'));
        orderID.add( id );
        orderStatus.add( (String)anObject.get("status"));

        List line_items = (List) anObject.get("line_items");
        Map<String, Object> line_object = (Map<String,
Object>)line_items.get(0);

        orderWorkerName.add( (String)line_object.get("name") );
        orderTotal.add( (String)line_object.get("total") );

        List meta_data = ((List)line_object.get("meta_data") );

        orderTime.add( (String)((Map<String,
Object>)meta_data.get(0)).get("value") );
        orderDate.add( (String)((Map<String,
Object>)meta_data.get(1)).get("value") );
        orderDescription.add( (String)((Map<String,
Object>)meta_data.get(2)).get("value") );
      }

      for (int i=0; i<orderID.size(); i++ ) {

        dataList.add(
            new AllRequestsData(
                orderID.get(i),
                orderStatus.get(i),
                orderWorkerName.get(i),
                orderTotal.get(i),
                orderTime.get(i),
                orderDate.get(i),
                orderDescription.get(i)
            )
        );
      }
```

```
      setRecyclerView();
    }
    else {
      Toast.makeText(context, "No Response String !",
Toast.LENGTH_SHORT).show();
    }
  }

  private void setRecyclerView() {
    RVListAllCategories = findViewById(R.id.recycler3);
    RVListAllCategories.setHasFixedSize(true);
    layoutManager = new LinearLayoutManager(this);
    RVListAllCategories.setLayoutManager(layoutManager);
    adapter = new RVAAllRequests(dataList);

    RVListAllCategories.setAdapter(adapter);
  }
}
```

AllRequestsData.java

```
package com.example.abdul.servicesmanagementsystem;

/**
 * Created by Abdul on 4/22/2018.
 *
 */

public class AllRequestsData {
  private String orderID;
  private String orderStatus;
  private String orderWorkerName;
  private String orderTotal;
  private String orderTime;  private String orderDate;  private String
orderDescription;
  public AllRequestsData(String orderID, String orderStatus, String
orderWorkerName, String orderTotal, String orderTime, String orderDate, String
orderDescription) {
    this.orderID = orderID;
    this.orderStatus = orderStatus;
    this.orderWorkerName = orderWorkerName;
    this.orderTotal = orderTotal;
    this.orderTime = orderTime;
    this.orderDate = orderDate;
    this.orderDescription = orderDescription;
  }
  public String getOrderID() {
    return orderID;
  }
  public String getOrderStatus() {
    return orderStatus;
  }
  public String getOrderWorkerName() {
    return orderWorkerName;
  }
  public String getOrderTotal() {
    return orderTotal;
  }
  public String getOrderTime() {
    return orderTime;
  }
  public String getOrderDate() {
    return orderDate;
  }
  public String getOrderDescription() {
    return orderDescription;
  } }
```

Appendix B: Main Activity

MainActivity.java

```
package com.example.abdul.servicesmanagementsystem;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Build;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

import com.android.volley.Request;
import com.android.volley.VolleyError;

import org.json.JSONException;
import org.json.JSONObject;

public class MainActivity extends BaseActivity {
  Integer basicTokenCounter;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    oneTimeCode();

    basicTokenCounter = 0;
    getAndSaveBasicTOKEN(); // to check internet connectivity and set basic
working token.
  }


  private void getAndSaveBasicTOKEN() {
    CustomRequest customRequest = new CustomRequest(new IResult() {
      @Override
      public void notifySuccess(String response) {

        try {
          String value;
          JSONObject object = new JSONObject(response);
          value = object.getString("token");
```

```
            All.setSharedSTR(All.BASIC_TOKEN, value, MainActivity.this);
          } catch (JSONException e) {
            e.printStackTrace();
          }
        }

        @Override
        public void notifyError(VolleyError error) {
          All.handleVolleyError(MainActivity.this, error);
          Toast.makeText(MainActivity.this, "Check Internet Connection",
Toast.LENGTH_SHORT).show();
          if (basicTokenCounter < 3) {
            getAndSaveBasicTOKEN();
          }
        }
      },
          MainActivity.this,
          "http://" + All.LOCALHOST + "/fixmyhome/wp-json/jwt-auth/v1/token",
          Request.Method.POST
      );
      customRequest.setParamAndValue("username", "admin");
      customRequest.setParamAndValue("password", "admin");
      customRequest.executeRequest();
      basicTokenCounter++;
    }

    public void onHome(View v) {
      startActivity(new Intent(this, Categories.class));
    }

    private void oneTimeCode() {
      Log.v("MODEL:", Build.MODEL);
      if ((Build.MODEL).equals("Android SDK built for x86")) {
        All.setLOCALHOST(true);
      } else {
        All.setLOCALHOST(false);
      }
    }
}
```

BaseActivity.java

```
package com.example.abdul.servicesmanagementsystem;

import android.content.Context;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;

public class BaseActivity extends AppCompatActivity {

  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.options_menu, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
      case R.id.services:
        startActivity(new Intent(BaseActivity.this, Categories.class));
        return true;
      case R.id.change_ip:
        All.changeLocalhost(BaseActivity.this);
        return true;
      case R.id.login:
        startActivity(new Intent(BaseActivity.this, Login.class));
        return true;
      case R.id.all_requests:
        startActivity(new Intent(BaseActivity.this, AllRequests.class));
        return true;
      case R.id.restart:
        startActivity(new Intent(BaseActivity.this, MainActivity.class));
        return true;
      default:
        return super.onOptionsItemSelected(item);
    }
  }
}
```

Appendix C: Login Page

Login.java

```
package com.example.abdul.servicesmanagementsystem;

import android.app.ProgressDialog;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

import com.android.volley.Request;
import com.android.volley.VolleyError;

import org.json.JSONException;
import org.json.JSONObject;

public class Login extends BaseActivity {
  private EditText user, pass;
  private ProgressDialog progressDialog;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login);
    if (getSupportActionBar() != null)
      getSupportActionBar().setTitle("Login");

    user = findViewById(R.id.username);
    pass = findViewById(R.id.password);
    progressDialog = new ProgressDialog(Login.this);
    progressDialog.setMessage(All.LOADING_MSG);
  }

  public void onSubmit(View v) {
    String username = user.getText().toString().trim();
    String password = pass.getText().toString();
    if (username.isEmpty()) {
      Toast.makeText(this, "Username is empty", Toast.LENGTH_SHORT).show();
      return;
    }
    if (password.isEmpty()) {
      Toast.makeText(this, "Password is empty", Toast.LENGTH_SHORT).show();
      return;
```

```
    }

    CustomRequest customRequest = new CustomRequest(
        new IResult() {
          @Override
          public void notifySuccess(String response) {

            if (response.length() < 10) {
              progressDialog.dismiss();
              Toast.makeText(Login.this, response, Toast.LENGTH_SHORT).show();
              return;
            }
//            String[] props = {"token", "user_email", "user_nicename",
"user_display_name"};

            All.setSharedSTR(All.TOKEN, getProperty(response, "token"),
Login.this);

            String nice_name = getProperty(response, "user_nicename");

            // user email will be used later.
            All.setSharedSTR(All.USER_EMAIL, getProperty(response,
"user_email"), Login.this);

            String url = "http://" + All.LOCALHOST+"/fixmyhome/wp-
json/wp/v2/users?slug=" + nice_name;

            CustomRequest customRequest1 = new CustomRequest(new IResult() {
              @Override
              public void notifySuccess(String response) {
                progressDialog.dismiss();
                if (response.length() < 6 ) {
                  Toast.makeText(Login.this, "User not recognized.",
Toast.LENGTH_SHORT).show();
                  return;
                }

                All.setSharedSTR(All.USER_PIC,
                    All.getStringList(response, "\"96\":\"", "\"").get(0),
                    Login.this
                );
                All.setSharedSTR(All.USER_NAME,
                    All.getStringList(response, "\"name\":\"", "\"").get(0),
Login.this);

                All.setSharedSTR(All.USER_ID,
                    All.getStringList(response, "\"id\":", ",").get(0),
Login.this);
                Log.v("response", response);
```

```
                startActivity(new Intent(Login.this, UserProfile.class));
              }

              @Override
              public void notifyError(VolleyError error) {
                progressDialog.dismiss();
                All.handleVolleyError(Login.this, error);
              }
            },
                Login.this,
                url,
                Request.Method.GET
            );
            customRequest1.setHeaderAndValue("Authorization", "Bearer " +
                All.getSharedSTR(All.BASIC_TOKEN, Login.this));
            customRequest1.executeRequest();
          }
          @Override
          public void notifyError(VolleyError error) {
            progressDialog.dismiss();
            All.handleVolleyError(Login.this, error);
          }
        },
        this,
        "http://" + All.LOCALHOST+"/fixmyhome/wp-json/jwt-auth/v1/token",
        Request.Method.POST);
    customRequest.setParamAndValue("username", username);
    customRequest.setParamAndValue("password", password);
    progressDialog.show();
    customRequest.executeRequest();
  }
  public void forgotPassword(View v) {
    startActivity(new Intent(Login.this, ForgotPassword.class));
  }
  private String getProperty(String source, String key) {
    String value = "";
    try {
      JSONObject object = new JSONObject(source);
      value = object.getString(key);
    } catch (JSONException e) {
      e.printStackTrace(); }
    if (value != null) {
      return value;
    }
    return ""; } }
```

Appendix D: Request Submission

UserProfile.java

```
package com.example.abdul.servicesmanagementsystem;

import android.content.Context;
import android.os.Bundle;
import android.util.DisplayMetrics;
import android.view.Display;
import android.view.WindowManager;
import android.widget.ImageView;
import android.widget.TextView;
import com.squareup.picasso.Picasso;
public class UserProfile extends BaseActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_user_profile);
    String userName;    String userPic;    String email;
    userName = All.getSharedSTR(All.USER_NAME, UserProfile.this);
    if (getSupportActionBar() != null)
      getSupportActionBar().setTitle("Welcome " + userName);
    ImageView profilePic = findViewById(R.id.userPic);
    userPic = All.getSharedSTR(All.USER_PIC, UserProfile.this);
    Picasso.get().load(userPic).into(profilePic);

    WindowManager wm = (WindowManager)
this.getSystemService(Context.WINDOW_SERVICE);
    DisplayMetrics metrics = new DisplayMetrics();

    Display display = null;
    if (wm != null) {
      display = wm.getDefaultDisplay();
    }
    if (display != null) {
      display.getMetrics(metrics);
    }
    Integer width = metrics.widthPixels / 2; // 50% width
    Integer height = metrics.heightPixels / 3; // 33% height
    profilePic.getLayoutParams().width = width;
    profilePic.getLayoutParams().height = height;
//    profilePic.requestLayout();
    email = All.getSharedSTR(All.USER_EMAIL, UserProfile.this);
    TextView userEmail = findViewById(R.id.textViewEmail);
    userEmail.setText("Email: ");
    userEmail.append(email);
  } }
```

Appendix E: Worker Application

RVAMyOrders.java

```
package com.example.abdul.fmhserviceprovider;

import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;

import java.lang.reflect.Method;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class RVAMyOrders extends
RecyclerView.Adapter<RecyclerViewHolderMyOrders> {
  Context context;
  private List<MyOrdersData> allRequestsData = new ArrayList<MyOrdersData>();

  public RVAMyOrders(List<MyOrdersData> allRequestsData) {
    this.allRequestsData = allRequestsData;
  }

  @NonNull
  @Override
  public RecyclerViewHolderMyOrders onCreateViewHolder(@NonNull ViewGroup
viewGroup, int i) {

    LayoutInflater inflater = LayoutInflater.from(context =
viewGroup.getContext());
```

```
    View itemView = inflater.inflate(R.layout.my_orders_layout, viewGroup,
false);

    return new RecyclerViewHolderMyOrders(itemView,
        new RecyclerViewHolderMyOrders.ViewHolderClicks() {
          @Override
          public void onOrderClick(final TextView ID, TextView name, TextView
date, TextView time, TextView total) {

            final ProgressDialog progressDialog = new ProgressDialog(context);
            progressDialog.setMessage(All.LOADING_MSG);

            AlertDialog alertDialog = new
AlertDialog.Builder(context).create();
            alertDialog.setTitle("Order ID: "+ID.getText().toString());
            alertDialog.setMessage("Do you want to mark this order as Paid and
Completed?\n\n" +
                name.getText().toString() +"\n"+
                total.getText().toString() + "\n"+
                date.getText().toString() + "\n" + time.getText().toString()
            );
            alertDialog.setButton(AlertDialog.BUTTON_POSITIVE, "YES",
                new DialogInterface.OnClickListener() {
                  public void onClick(DialogInterface dialog, int which) {

                    StringRequest stringRequest = new StringRequest(
                        Request.Method.POST,
                        "http://" + All.LOCALHOST + "/fixmyhome/wp-
json/wc/v2/orders/" + ID.getText().toString(),
                        new Response.Listener<String>() {
                          @Override
                          public void onResponse(String response) {
                            progressDialog.dismiss();
                            context.startActivity(new Intent(context,
MyOrders.class));
                          }
                        },
                        new Response.ErrorListener() {
                          @Override
                          public void onErrorResponse(VolleyError error) {
                            progressDialog.dismiss();
                            All.handleVolleyError(context, error);
                          }
                        }
                    ){
                      @Override
                      public Map<String, String> getHeaders() throws
AuthFailureError {
```

```
                        String basicToken = All.getSharedSTR(All.BASIC_TOKEN,
context);
                        Map<String, String> headers = new HashMap<String,
String>();
                        headers.put("Authorization", "Bearer " + basicToken);
                        return headers;
                      }

                      @Override
                      public byte[] getBody() throws AuthFailureError {
                        String body =
                            "{\n" +
                                "\t\"status\": \"completed\"\n" +
                                "}";
                        return body.getBytes();
                      }

                      @Override
                      public String getBodyContentType() {
                        return "application/json; charset=utf-8";
                      }

                    };
                    All.executeRequest(context, stringRequest);
                    progressDialog.show();

                    dialog.dismiss();
                  }
                });
            alertDialog.setButton(AlertDialog.BUTTON_NEGATIVE, "Cancel",
                new DialogInterface.OnClickListener() {
                  public void onClick(DialogInterface dialog, int which) {
                    dialog.dismiss();
                  }
                });
            alertDialog.show();
          }
        });
  }

  @Override
  public void onBindViewHolder(@NonNull RecyclerViewHolderMyOrders
recyclerViewHolderMyOrders, int i) {

    recyclerViewHolderMyOrders.goneID.setText(allRequestsData.get(i).getOrderI
D());

    recyclerViewHolderMyOrders.tvID.setText("Order ID: " +
allRequestsData.get(i).getOrderID());
```

```
    recyclerViewHolderMyOrders.tvDate.setText("Date: " +
allRequestsData.get(i).getOrderDate());
    recyclerViewHolderMyOrders.tvDescription.setText("Details: " +
allRequestsData.get(i).getOrderDescription());
    recyclerViewHolderMyOrders.tvStatus.setText("Status: " +
allRequestsData.get(i).getOrderStatus());
    recyclerViewHolderMyOrders.tvTime.setText("Time: " +
allRequestsData.get(i).getOrderTime());
    recyclerViewHolderMyOrders.tvTotal.setText("Total Amount: " +
allRequestsData.get(i).getOrderTotal());
    recyclerViewHolderMyOrders.tvCustomerName.setText("Customer: " +
allRequestsData.get(i).getCustomerName());
    recyclerViewHolderMyOrders.tvCustomerAddress.setText("Address: " +
allRequestsData.get(i).getCustomerAddress());
    recyclerViewHolderMyOrders.tvCustomerPhone.setText("Phone: " +
allRequestsData.get(i).getCustomerPhone());

  }

  @Override
  public int getItemCount() {
    return allRequestsData.size();
  }
}

class RecyclerViewHolderMyOrders extends RecyclerView.ViewHolder implements
View.OnClickListener {

  public TextView tvID;
  public TextView goneID;
  public TextView tvStatus;
  public TextView tvCustomerName;
  public TextView tvCustomerAddress;
  public TextView tvCustomerPhone;
  public TextView tvTotal;
  public TextView tvDate;
  public TextView tvTime;
  public TextView tvDescription;

  public ViewHolderClicks mListener;

  public RecyclerViewHolderMyOrders(View itemView, ViewHolderClicks listener)
{
    super(itemView);
    this.mListener = listener;

    tvID = itemView.findViewById(R.id.all_requests_id);
    goneID = itemView.findViewById(R.id.all_requests_gone_id);
    tvStatus = itemView.findViewById(R.id.all_requests_status);
```

```
    tvCustomerName = itemView.findViewById(R.id.all_requests_customer_name);
    tvTotal = itemView.findViewById(R.id.all_requests_total);
    tvDate = itemView.findViewById(R.id.all_requests_date);
    tvTime = itemView.findViewById(R.id.all_requests_time);
    tvDescription = itemView.findViewById(R.id.all_requests_details);
    tvCustomerAddress = itemView.findViewById(R.id.all_requests_address);
    tvCustomerPhone = itemView.findViewById(R.id.all_requests_phone);

    itemView.setOnClickListener(this);
  }

  @Override
  public void onClick(View view) {
    mListener.onOrderClick(goneID, tvCustomerName, tvDate, tvTime, tvTotal);
  }

  public static interface ViewHolderClicks {
    void onOrderClick(TextView ID, TextView name, TextView date, TextView
time, TextView total);
  }
}
```

REFERENCES

Gavhane, S., et al. (2015). "Study of Implementation of Society Management System." International Journal of Computer Applications **132**(1): 34-36.

Joo, Y.-D. (2013). "Implementation of Facility Maintenance Management System using Smart Phones." The Journal of The Institute of Internet, Broadcasting and Communication **13**(1): 191-197.

Kumar, S. (2014). "Ubiquitous smart home system using android application." arXiv preprint arXiv:1402.2114.

Kumbhar, M. D. M. R. M. and A. Dilip (2016). "UBIQUITOUS HOME CONTROL AND MONITORING SYSTEM USING INTERNET OF THINGS."

Vatharkar, R., et al. (2016). "Implementation Of Society Management System: Societales." International Journal of Science and Technology.