Project Paper
On
The Mechanics : A person who repairs and Maintain machinery

A project submitted in partial fulfilment of the requirement for the degree of B.Sc (Hon’s) in Computer Science and Engineering

Department of Computer Science and Engineering

Dhaka City College

National University, Bangladesh

April 7, 2022

Project Paper On

The Mechanics : A person who repairs and Maintain machinery

Supervised By

Farjana Akter Lina

Lecturer of Dhaka City College

Depertment of CSE.

Submitted By

Afia Tabassum Shamonty

Reg. No: 16502000513

Sadia Islam Nowrin

Reg:16502000406

Session: 2016-2017

Department of Computer Science and Engineering
Dhaka City College
National University, Bangladesh
April 7, 2022

Declaration

I here by declare that the project work entitled The Mechanics **is** record of work done by me under the supervision of **Farjana Akter Lina**, lecturer, Department of Computer Science and Engineering, Dhaka City College. I also declare that the results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma**.**

Countersigned

Farjana Akter lina
Supervisor

Candidate 1:

Candidate 2:

APPROVAL

This project report entitled “**The Mechanics**” was prepared and submitted by Reg.No: 16502000513to the Department of Computer Science and Engineering, the National University of Bangladesh in the partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Engineering, has been examined and recommended for approval and acceptance as to its style and contents.

Supervisor

Farjana Akter Lina

Lecturer of Dhaka City College

Head of the department

Shamima Akter

Associate Professor and Head

Department of CSE

Dhaka City College

ACKNOWLEDGEMENTS

First of all, we would like to thank Almighty Allah, the most merciful for keeping his bleesings on us in every shape of life.

We would like to express our deep gratitudeto **Farjana Akter Lina**, Lecturer, Department of Computer Science & Engineering, and Dhaka City College. Her valuable ideas, thoughts provided very welcome depth and providing valuable feedback to us to develop this project. We are grateful, but we think we will remember her for her humor, kindness, constant encouragement, and the freedom she gave to us to develop intellectually while making sure we do not go too far off track.

We would like to thank **Shamima Sultana,** Associate & Head of the Department of Computer Science and Engineering ,Dhaka City College for inspiring us and giving us the opportunity to participate in this project.

We also acknowledge with a deep sense of reverence, our gratitude towards our guide teacher **Shirin Akter**, Lecturer, Depertment of Computer Science and Engineering, Dhaka City College for her cordial support, guidance, which helps us in completing this task through various stages.

Gratitude goes to our teachers who directly or indirectly helped us to complete this project

At last, we would like to thank all the people who have over the years provided feedback in various forms that have helped ours intellectual and personal growth, which, we believe and hope, finds a manifestation through this dissertate.

Thanking you
Afia Tabassum Shamonty
Sadia Islam Nowrin

ABSTRACT

In today’s world, technology plays a significant role in every industry as well as in our personal lives. Smartphones can potentially play an important role to make our life easier. In this proposal, we are presenting a smartphone app that will help to establish a connection between the customer and the mechanics. This will make the easier way to find out in every area. This app will help every mechanics to work and deal with good customers. There could be many apps related to our app, which are available online. Unfortunately, most of them don’t run across the user’s demand. The purpose of our project is to develop a smartphone-based hiring mechanics app that will provide a better opportunity for an mechanics. This app will also be profitable for the customer. This app will help the people to get the mechanics faster when they will be needed. This smartphone app will help people to find mechanics in their location. People can get good services from mechanics as they can see the details of every mechanics by using this smartphone app. In a huge market, there can be several mechanics and by using this app customers can deal with the better mechanics. Our app will help the customer to find the mechanics easily and to deal with the mechanics in their budgets. We wish that our app will be helpful for every people and it will also be helpful for the technology and it will be helpful for the people who want services and get them, mechanics, easily by using our smartphone-based hiring mechanics app. We will present the paper that provides an idea and outline its advantages and disadvantages. Then we discuss our current work proposing a way to use purposing simple way.

TABLE OF CONTENTS

Content

List of figures

Chapter -1

Introduction

CHAPTER -1
INTRODUCTION

INTRODUCTION

This chapter covers an introduction to the project including the context, a description of aims and objectives, a description of what has been achieved, contributions, and the structure of the report.

1.1 Context

Smartphones can potentially play an important role to make our life easier. In this proposal, we are presenting a smartphone app that will help to establish a connection between the customer and the mechanics. We will present the paper that provides an idea and outline its advantages and disadvantages. Then we discuss our current work proposing a way to use model life simple.

1.2 Motivation

As wireless technology improves, devices with wireless capability have become a regular part of our life. Though people lead a very busy life, these smartphones making our life easier. So when it comes to my mind that smartphones becoming a regular part of our life day by day, I am doing something that will help to make a connection between the customer & the mechanics.

1.3 Objectives

The objective of this project is to develop an android app that links the customer and mechanics. It provides a better opportunity for mechanics in a competitive market. This project will also be profitable for the customers as they can get their desired deal at the best price.

1.4 Methodology

The design and development of the system followed the waterfall model. The waterfall development model is a product of the manufacturing and development industries where making after-the-fact changes is often prohibitively costly. A development project using this model follows a fixed, linear sequence. Each phase must be completed in its entirety before the next phase can begin. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue the project. An advantage of the waterfall method is that it promotes strong documentation of each step of the development process.

The waterfall model emphasizes that a logical progression of steps be taken throughout the software development life cycle (SDLC), much like the cascading steps down an incremental waterfall. While the popularity of the waterfall model has waned over recent years in favor of more agile methodologies, the logical nature of the sequential process used in the waterfall method cannot be denied, and it remains a common design process in the industry. The waterfall model follows a series of processes, which are used during development. Usually, the stages will require the gathering of requirements and their analysis. The design of the system is the next stage, followed by coding the actual system.
Evaluation, testing, and debugging, if necessary, is the next step. Finally, the system will either be accepted and therefore maintained or rejected. It is vital to move to the next process of the waterfall model if the previous step has been completed.

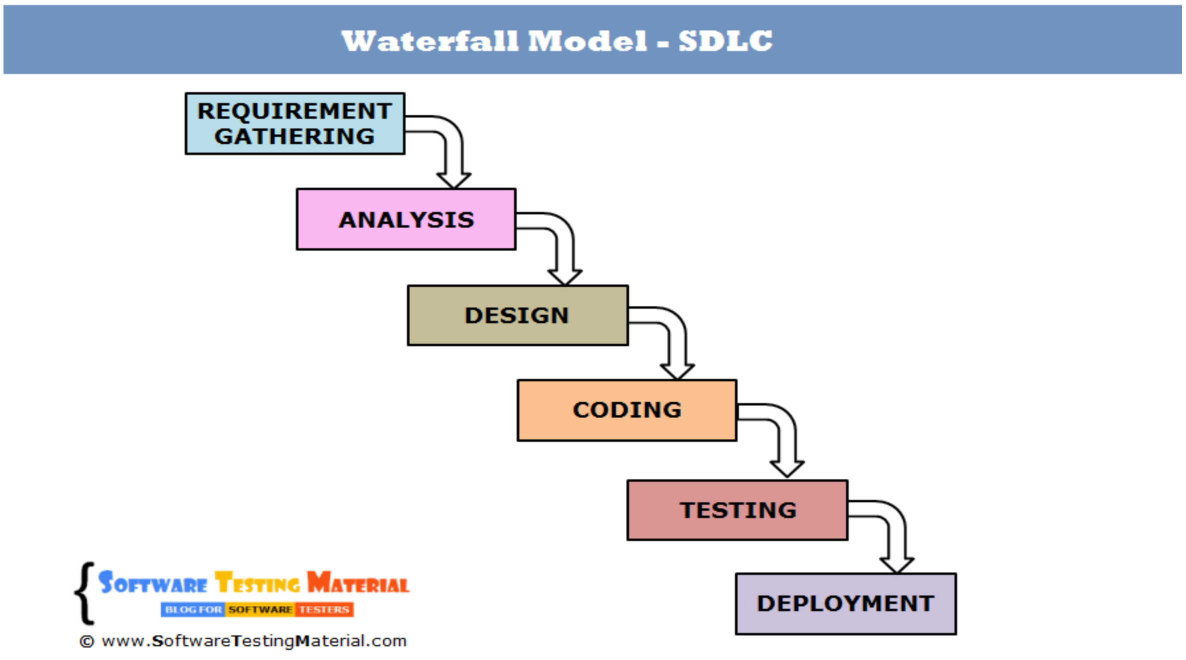

Waterfall Model - SDLC
REQUIREMENT GATHERING
ANALYSIS
DESIGN
CODING
TESTING
DEPLOYMENT
Software Testing Material
BLOG FOR SOFTWARE TESTERS
© www.SoftwareTestingMaterial.com

Figure 1.4: Waterfall Model

1.5 Outline of Dissertation

This report is divided into several chapters.

Chapter two covers analysis and requirement which gives an overview of existing applications which are developed for making the interconnection between customer and service provider. Chapter three gives an overview of the proposal methodologies of the system. Chapter four covers the design and data flow diagram of the system. It includes the database design, an explanation of the algorithms and how they work. Chapter five covers the implementation of the system. Chapter six is about future work. It contains the drawbacks or limitations and plan of this project.

Chapter -2

Analysis & Requirement

CHAPTER -2
ANALYSIS & REQUIREMENTS

2.1 Introduction

Requirements analysis or requirements engineering is a process used to determine the needs and expectations of a new product .It involves frequent communication with the stakeholders and end users of the product to define expectations, involves conflicts and document all the key requirements.

The UR phase can be called the ‘concept' or ‘problem definition' phase of the ESA PSS-05-0 life cycle. User requirements often follow directly from a spontaneous idea or thought. Even so, wide agreement and understanding of the user requirements is more likely if these guidelines are applied. The definition of user requirements is an iterative process. User requirements are documented in the User Requirements Document (URD). The URD gives the user's view of the problem, not the developer's. A URD may have to go through several revisions before it is acceptable to everyone. The main outputs of the UR phase are the:

i. User Requirements Document (URD);
ii. Software Project Management Plan for the SR phase (SPMP/SR);
iii. Software Configuration Management Plan for the SR phase (SCMP/SR);
iv. Software Verification and Validation Plan for the SR Phase (SVVP/SR);
v. Software Quality Assurance Plan for the AD phase (SQAP/SR);
vi. Acceptance Test Plan (SVVP/AT).

2.2 Literature Review

Android has been a major revolution, initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The operating system has since gone through multiple major releases, with the current version being 10 “Android Q”, released in November 2019.

Its impact is enormous; starting with the fact it is open-source, which makes it a playground to innovate new OS for new devices and so many changes and flavors for so things. It sculptures the ground for any app that is possible by imagination.

Android is a mobile operating system developed by Google, based on a modified version of the Linux kernel and other open-source software, and designed primarily for touchscreen mobile devices such as smartphones and tablets. Besides, Google has further developed Android TV for televisions, Android Auto for cars, and Android Wear for wrist watches, each with a specialized user interface. Variants of Android are also used on game consoles, digital cameras, PCs, and other electronics.

2.3 Existing Systems & Applications

There is a hiring app like this in Bangladesh which is Sheba xyz. This app also establishes a connection between a customer and the service provider. But it has a payment gateway through this app. Which makes this a little complex for the mechanics based on our country. Here in our app, we make it simple and easier to use for all the level of the people. This simplicity is our main focus point.

2.4 Software Requirements

As our proposed system is android based, so we need the setup outlined as follows:

- Android Studio IDE (Integrated Development Environment)
- Android Platform
- Android SDK (Software Development Kit)
- Bluestack / Andy / Amiduos
- ADT (Android Development Tool) plug-in
- Java Run timeEnvironment7.0(SE)
- Android Smartphone(Android Kit Kat 4.4.2 or Higher)

2.5 User Requirements & Guidance

- As a customer, I need to register so that I can log in.
- As a customer, I want to search for Mechanics so that I can get expected results.
- As a customer, I want to see the search list.

- As a customer, I want to contact you through call.
- As a customer, after completion of the work customer can give feedback to that individual service provider.
- As a service provider, I need to register so that I can log in.
- As a service provider, He has to enlist himself in an individual location.
- As a service provider, He has to accept the task and complete it.
- As a service provider, after completion of the task can give feedback to the customer.

2.6 Conclusion

This is a minimum cost app. To develop the android app the cost will be on average. The requirements of an android development are also cost-effective. Necessary software and hardware are also available. User requirements for this android app are also low cost.

Chapter -3

PROPOSED METHODOLOGIES

CHAPTER -3
PROPOSED METHODOLOGIES

3.1 Methodology

The methodology is the systematic, theoretical analysis of the methods applied to a field of study. It comprises the theoretical analysis of the body of methods and principles associated with a branch of knowledge. Typically, it encompasses concepts such as paradigm, theoretical model, phases, and quantitative or qualitative techniques, here our “Duties of Mechanics”, we used a few methods and technique which we are going to
describe below.

3.2 Proposal Methodology

The research problem in this paper requires the use of both qualitative and quantitative methods. For instance, the conceptualisation of a construct entails close examination of meanings, awareness of peoples’ everyday understanding of the concept, and analysis of the literature and relevant knowledge bases to determine the dimensions to be included in the construct. The operationalisation and empirical validation involve assessing validity and reliability through techniques such as exploratory factor analysis and reliability analysis. Keeping in view the objectives and the nature of the research problem, this research methodology draws on the positivist research paradigm and uses techniques involving both qualitative (substantive) and quantitative (structural) aspects (seen in Figure 1 and explained below).

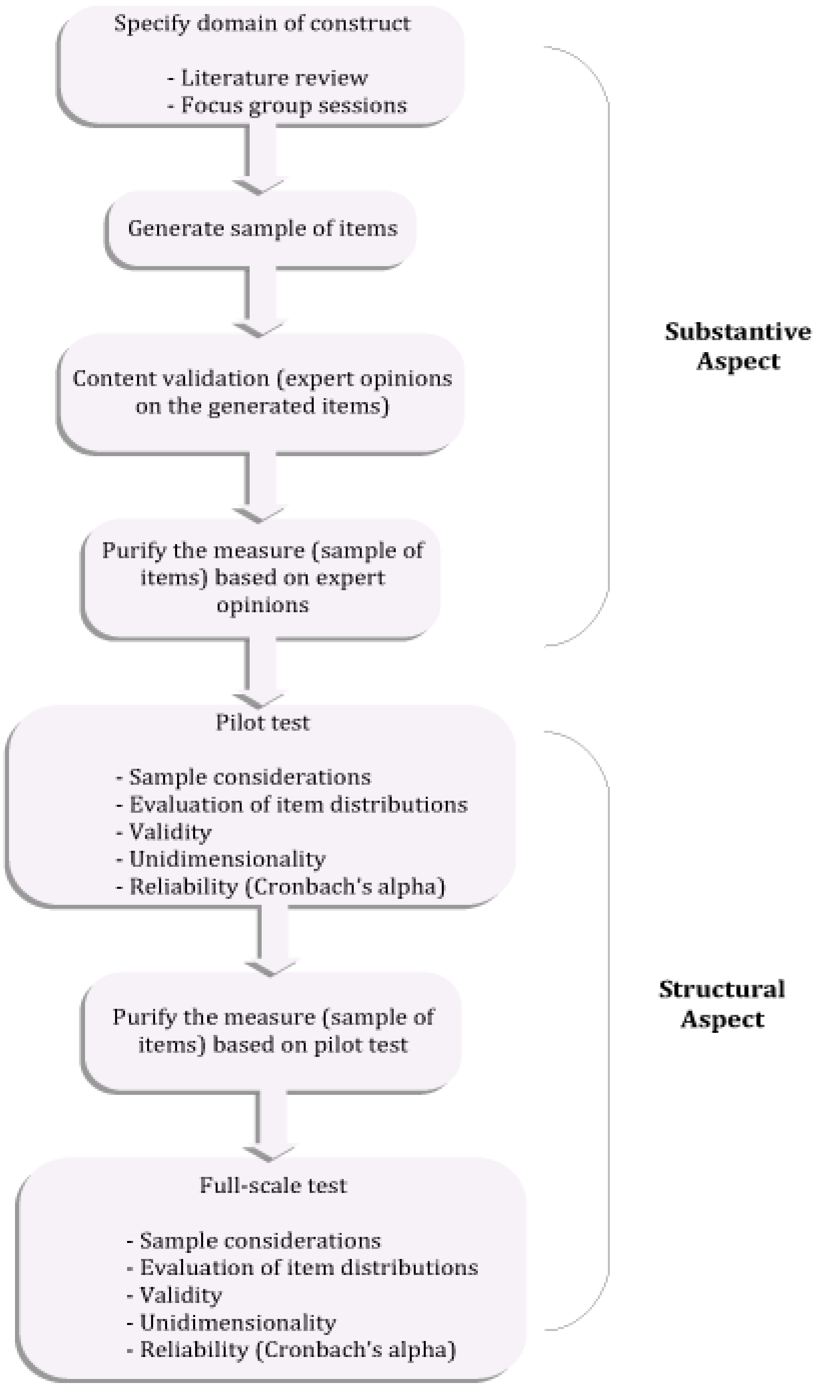

Specify domain of construct
- Literature review
- Focus group sessions
Generate sample of items
Content validation (expert opinions on the generated items)
Purify the measure (sample of items) based on expert opinions
Pilot test
- Sample considerations
- Evaluation of item distributions
- Validity
- Unidimensionality
- Reliability (Cronbach's alpha)
Purify the measure (sample of items) based on pilot test
Full-scale test
- Sample considerations
- Evaluation of item distributions
- Validity
- Unidimensionality
- Reliability (Cronbach's alpha)
Substantive Aspect
Structural Aspect

Figure 1: Proposed methodology

As noted earlier, the proposed methodological approach has been used in other disciplines, including psychology, marketing, and information systems. The work of Moore and Benbasat (1991) is an important example of the application of this approach in the information systems discipline. Their work examined the concepts of relative advantage, compatibility, complexity, observability, and trialability. Those concepts were proposed by Rogers (1983), who argued that they influence the adoption of any innovation. Moore and Benbasat developed a validated instrument to measure perceptions of the concepts. They started with a review of the literature relating to the concepts and examined instruments that had been used in previous attempts to operationalise them. Following this phase, three additional steps were taken: (1) items creation, (2) instrument development, and (3) instrument testing. After the first step, content and construct validity were assessed. Following those assessments, the initial instrument operationalising the concepts was refined and pilot tested. Finally, the instrument was re-tested through a full-scale test that included validity and reliability assessment. Agarwal and Prasad (1997) noted that Moore and Benbasat subjected their instrument '*to an intensive validation procedure to determine reliability and validity*' (p. 567).

We decided to make such an android app that helps to establish a connection between mechanics and customer. There will be mainly three panels. Mechanics, customer, and an administration panel. The core of the system is on android.

There are some things we will have to use:

- The development of this system using Android Studio.
- There will be GPS enabled phone.
- Minimum 1GB ram configured phone.
- There will be using Java language.

Chapter -4

DESIGN

CHAPTER – 4
DESIG

4.1 Design Architecture

The design architecture part consists of the MVVM architecture for android as this is an android application. This architecture is described here.

4.2 MVVM

Model–view–view model (MVVM) is a software architectural pattern. MVVM facilitates a separation of development of the graphical user interface – be it via a markup language or GUI code – from the development of the business logic or back-end logic (the data model). The view model of MVVM is a value converter, meaning the view model is responsible for exposing (converting) the data objects from the model in such a way that objects are easily managed and presented. In this respect, the view model is more model than the view and handles most if not all

of the view's display logic. The view model may implement a mediator pattern, organizing access to the back-end logic around the set of use cases supported by the view.

MVVM is a variation of Martin Fowler's Presentation Model design pattern. MVVM abstracts a view's state and behavior in the same way, but a Presentation Model abstracts a view (creates a view model) in a manner not dependent on a specific user-interface platform. MVVM and Presentation Model both derive from the model–view–controller pattern (MvC) .

The MVVM pattern is used in Windows Presentation Foundation (WPF), which runs on Microsoft’s .NET. Silverlight, a Microsoft WPF internet equivalent multimedia plug-in, also uses MVVM.

The separation of the code in MVVM is divided into View, ViewModel and Model:

- View is the collection of visible elements, which also receives user input. This includes user interfaces (UI), animations and text. The content of View is not interacted with directly to change what is presented.
- ViewModel is located between the View and Model layers. This is where the controls for interacting with View are housed, while binding is used to connect the UI elements in View to the controls in ViewModel.
- Model houses the logic for the program, which is retrieved by the ViewModel upon its own receipt of input from the user through View.

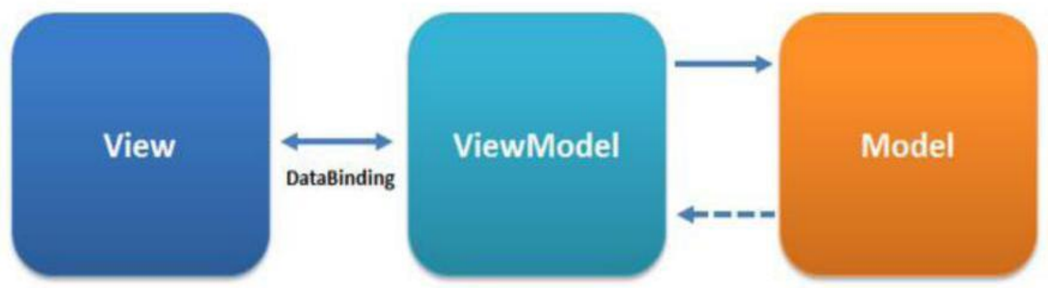


Figure 4.1: MVVM architecture

Now for this application, the MVVM architecture is described by a table shown here:

Model	Med Alert	App_ltem
View Model	Main Activity, Med Alert, Med Alert Add, Med Alert Adapter	Main Activity
View	Activit_Main, Activity_Med_Alert, Med_alert_item	activity_main, app_item

Main Activity, MedAlert activities. The MedAlertAdapter is for controlling the alert system. For the view it has activity_main, activity_med_alert, med_alert_item. App_item has the MainActivity as the viewmodel and app_item as the view.

4.3 Database Design

Database design is the process of producing a detailed data model of a database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity. The term database design can be used to describe many different parts of the design of an overall database system. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application. For this project data storage(static) is used to store the data. The table diagram for Duties of Mechanics is shown here.

Application Data Model

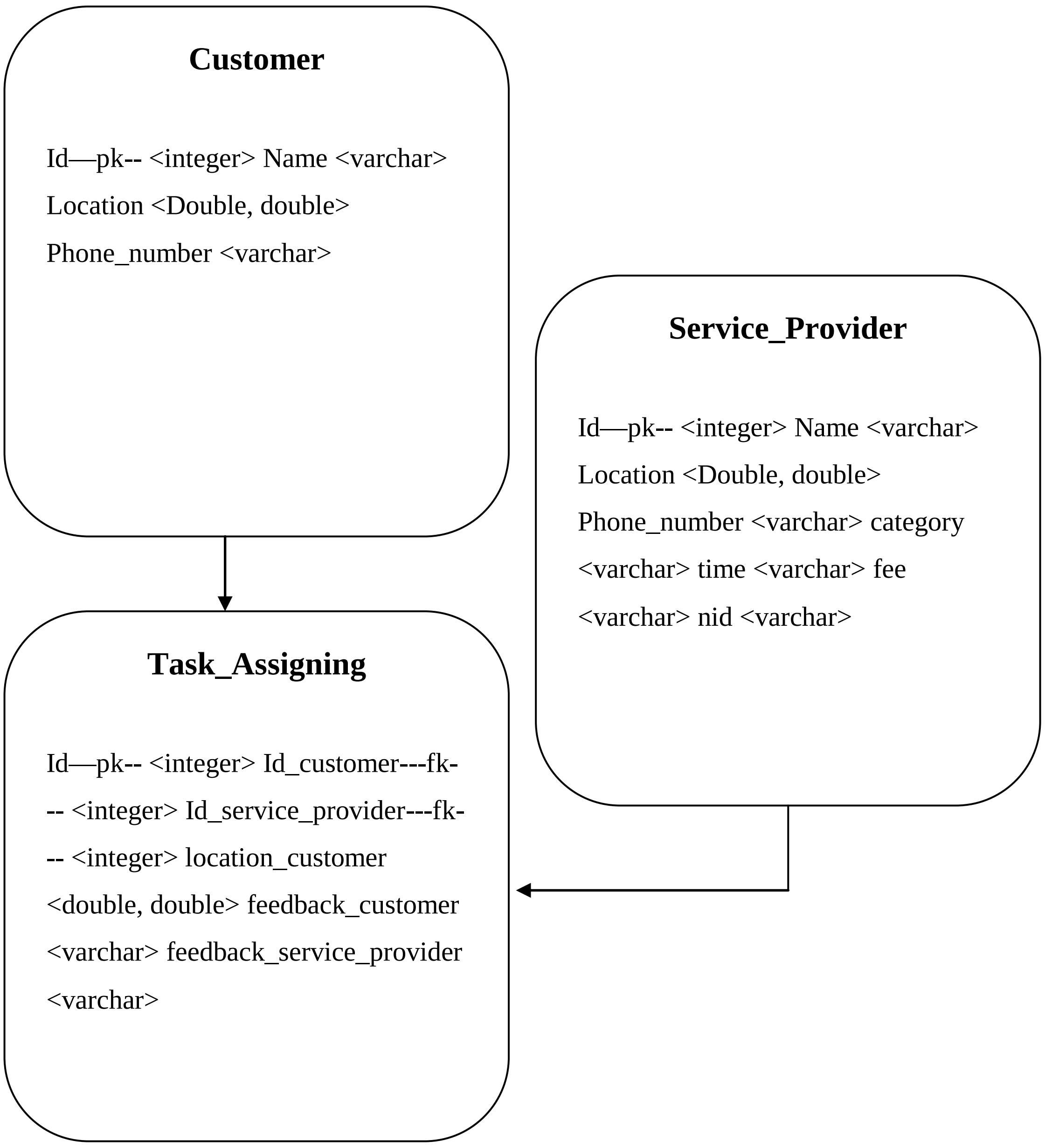

Customer
Id—pk-- <integer> Name <varchar>
Location <Double, double>
Phone_number <varchar>
Service_Provider
Id—pk-- <integer> Name <varchar>
Location <Double, double>
Phone_number <varchar> category
<varchar> time <varchar> fee
<varchar> nid <varchar>
Task_Assigning
Id—pk-- <integer> Id_customer---fk-
-- <integer> Id_service_provider---fk-
-- <integer> location_customer
<double, double> feedback_customer
<varchar> feedback_service_provider
<varchar>

Figure 4.3: Table diagram of Duties of Mechanics

4.4 Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing(structured design). A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or parallel, unlike a traditionally structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data flows as a unified model. The goal of data flow diagramming is to have a commonly understood model of a system. The diagrams are the basis of structured systems analysis. Data flow diagrams are supported by other techniques of structured systems analysis such as data structure diagrams, data dictionaries, and procedure-representing techniques such as decision tables, decision trees, and structured English.

A context diagram is a top-level (also known as "Level 0") data flow diagram. It only contains one process node("Process 0") that generalizes the function of the entire system in relationship to external entities.

DFD Layers:
Draw data flow diagrams can be made in several nested layers. A singleprocess node on a high-level diagram can be expanded to show a moredetailed data flow diagram. Draw the context diagram first, followed by various layers of data flow diagrams.

DFD Levels:
The first level DFD shows the main processes within the system. Each ofthese processes can be broken into further processes until you reach pseudo code.

Data Flow Diagrams Symbols:

Process Notations: A process transforms incoming data flow into outgoing data flow.

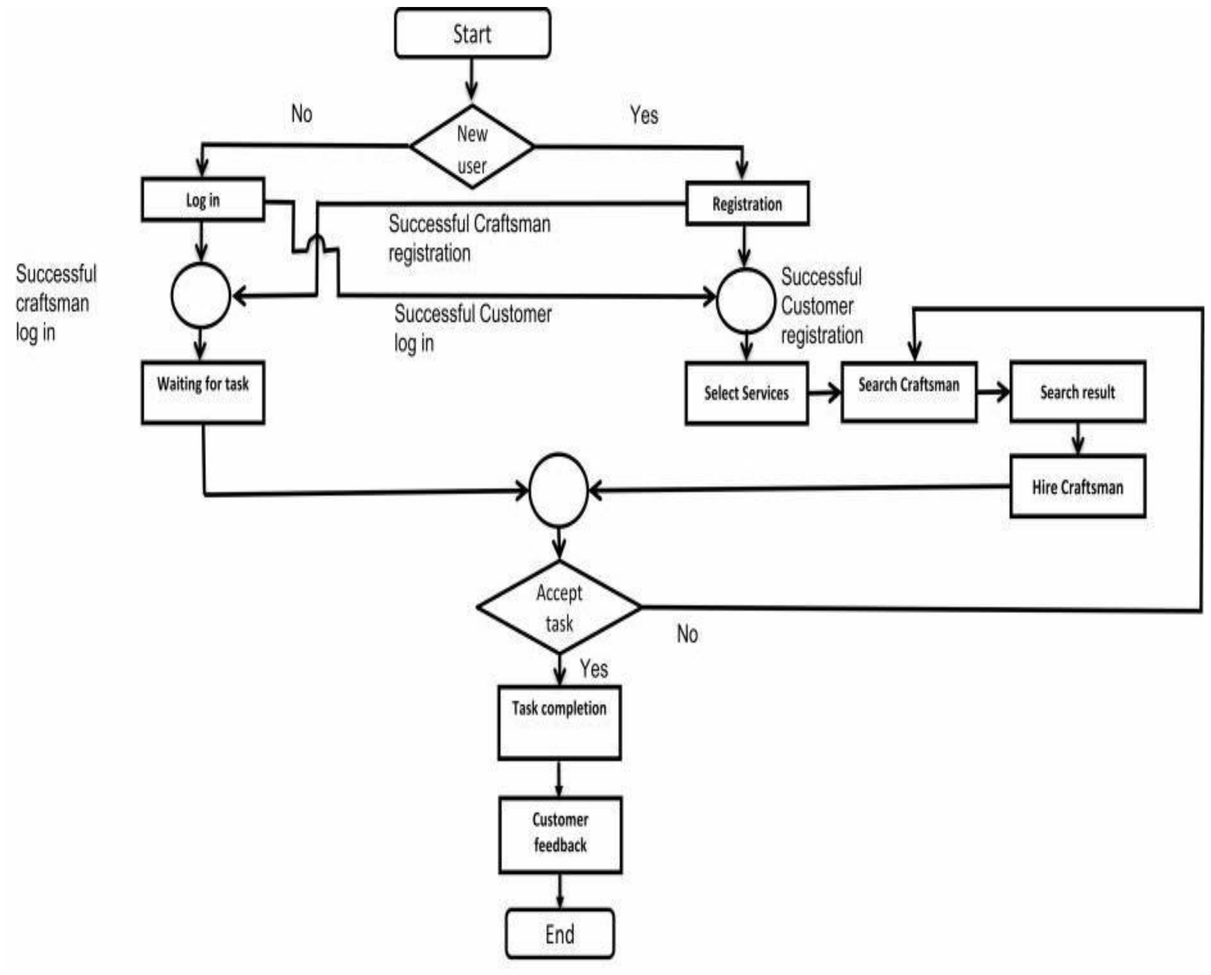
Data-store Notations: Data-stores are repositories of data in the system. They are sometimes also referred to as files.

Data-flow Notations: Data-flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.

External Entity Notations: External entities are objects outside the system, with whichthe system communicates. External entities are sources and destinations of the system's inputs and outputs.

The data flow diagram for the Duties of Mechanics is described here.

4.4: Data Flow Diagram



4.5 Conclusion

The design architecture needs to be so clear and specific. The project should maintain all the models and View Model shown in the MVVM architecture. The data flow diagrams also show the data flow of the data stored in the data storage for the actions.

Chapter -5

Implementation

CHAPTER -5
IMPLEMENTATION

5.1 Structural Design of System Implementation

The overall structure for any android application consists of several portions. These include-

- Android Manifest
- Java Class
- Layout Design
- Resource Directories

Each of these portions contains a different phase of the whole application. It is a must to synchronize all portions together; otherwise, an error message has occurred during the build operation of the application.

5.2 Android Manifest

Every application must have an AndroidManifest.xml file (with precisely that name) in its root directory. The manifest file presents essential information about your app to the Android system, information the system must have before it can run any of the app's code. Among other things, the manifest does the following: [5]

- It names the Java package for the application. The package name serves as a unique identifier for the application.

- It describes the components of the application — the activities, services, broadcast receivers, and content providers that the application is composed of. It names the classes that implement each of the components and publishes their capabilities (for example, which Intent messages they can handle). These declarations let the Android system know what the components are and under what conditions they can be launched.

- It determines which processes will host application components.

- It declares which permissions the application must have in order to access protected parts of the API and interact with other applications.

- It also declares the permissions that others are required to have in order to interact with the application's components.
- It lists the instrumentation classes that provide profiling and other information as the application is running. These declarations are present in the manifest only while the application is being developed and tested; they're removed before the application is published.
- It declares the minimum level of the Android API that the application requires.
- It lists the libraries that the application must be linked against.

5.3 Java Class

Java classes are grouped in different activities. These are -

Activity

- o BasicActivity
- o BottomNavigationActivity
- o EmptyActivity
- o FullscreenActivity
- o LoginActivity
- o Master/Detail Flow
- o NavigationDrawerActivity
- o ScrollingActivity
- o SettingsActivity
- o TabbedActivity
- o MainActivity
- o RegisterActivity
- o UserHomeActivity
- o WorkerHomeActivity

o DB Helper Package

o Adapter

o Model

5.4 Resource Directory

Here is the list of the resource directories:

o Drawable Layout

o Menu

o Mipmap

o Ic_luncherValues

o Google_maps_api

o Strings

o Styles

o Dimens

5.5 Layout

Layouts are intents which flow from one to one containing the elements of the application such as widgets, text fields, containers, elements etc. Here is a list of these layouts-

- activity_main.xml
- activity_register.xml
- activity_user_home.xml
- activity_worker_home.xml
- fragment_detail.xml
- login_layout.xml
- member_item.xml
- password_update_layout.xml
- query_layout.xml
- rating_sheet.xml
- update_info_dialog.xml

5.6 Source Code

```
<?xml version="1.0" encoding="utf-8" ?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.myapp.Mechanics">
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" /><uses-permission
android:name="android.permission.CALL_PHONE" />
<uses-feature
android:glEsVersion="0x00020000"
android:required="true" />
<application
android:allowBackup="true"
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<meta-data
android:name="com.google.android.gms.version"
android:value="@integer/google_play_services_version" />
<meta-data
android:name="com.google.android.geo.API_KEY"
android:value="AIzaSyD2dhuEo55Ex8kHzoHZ0VxmiM0yF2I0Trg" />
<activity android:name="com.myapp.Mechanics.activity.MainActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

```
</activity>

<meta-data
android:name="preloaded_fonts"
android:resource="@array/preloaded_fonts" />

<activity android:name="com.myapp.Mechanics.activity.RegisterActivity"
/><activity android:name="com.myapp.Mechanics.activity.WorkerHomeActivity"
/><activity android:name="com.myapp.Mechanics.activity.UserHomeActivity" />

</application>

</manifest>
```

MainActivity.class:

```
import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import com.google.android.gms.tasks.OnCompleteListener; import
com.google.android.gms.tasks.OnFailureListener; import
com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult; import
com.google.firebase.auth.FirebaseAuth; import
com.google.firebase.auth.FirebaseUser; import
com.google.firebase.database.DataSnapshot; import
com.google.firebase.database.DatabaseError; import
com.google.firebase.database.DatabaseReference; import
com.google.firebase.database.FirebaseDatabase
import com.google.firebase.database.ValueEventListener; import
com.myapp.Mechanics.R;
import com.myapp.Mechanics.utility.SharedPreferenceManager;

public class
MainActivity extends AppCompatActivity {
private Button hireButton;
private Button workButton;
private FirebaseAuthmAuth;
private String usertype;
private ProgressDialogmProgressDialog;
private DatabaseReferencedataRef;
private FirebaseDatabase database;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
//check if user id signed in !
if (SharedPreferenceManager.getInstance(this).isLoggedIn()) { if
(SharedPreferenceManager.getInstance(this).getUserType().equalsIgnoreCase("worker"))
{
startActivity(new Intent(this, WorkerHomeActivity.class)); } else if
(SharedPreferenceManager.getInstance(this).getUserType().equalsIgnoreCase("user")
) { startActivity(new Intent(this, UserHomeActivity.class));
}
} else {
setContentView(R.layout.activity_main);
database = FirebaseDatabase.getInstance();
initializeViews();
}
```

```
}
private void initializeViews
hireButton = findViewById(R.id.btn_hire); workButton =
findViewById(R.id.btn_work); mAuth = FirebaseAuth.getInstance();
mProgressDialog = new ProgressDialog(this); hireButton.setOnClickListener(new
View.OnClickListener() {

@Override
public void onClick(View v) {
usertype = "user";
dataRef = database.getReference().child(usertype).child("information");
showUserLoginDialog("user");
}
});

workButton.setOnClickListener(new View.OnClickListener() { @Override
                                    public void onClick(View v) {
                                          usertype = "worker";
dataRef = database.getReference().child(usertype).child("information");
showUserLoginDialog("worker");
}
});
}

private void showUserLoginDialog(final String user) { AlertDialog.Builder builder;
builder = new AlertDialog.Builder(this);
builder.setTitle("Login");

View viewInflated = LayoutInflater.from(this).inflate(R.layout.login_layout, null,
false);
final EditTextemailInput = (EditText) viewInflated.findViewById(R.id.login_email);
final EditTextpasswordInput = (EditText)
viewInflated.findViewById(R.id.login_password);
Button registerButton = viewInflated.findViewById(R.id.btn_register);
builder.setView(viewInflated);

builder.setPositiveButton(android.R.string.ok, new DialogInterface.OnClickListener()
{
@Override
public void onClick(DialogInterface dialog, int which) {
```

```
String email = emailInput.getText().toString();
String password = passwordInput.getText().toString();

if (email.matches("") || password.matches("")) { Toast.makeText(MainActivity.this, "No
Fields Can Be empty",

Toast.LENGTH_SHORT).show();
} else {
//LOGIN USER
mProgressDialog.setMessage("Logging In...");
mProgressDialog.show();
mAuth.signInWithEmailAndPassword(email, password)
.addOnCompleteListener(MainActivity.this, new
OnCompleteListener<AuthResult>() {
@Override
public void onComplete(@NonNull Task<AuthResult> task) {
mProgressDialog.dismiss();
if (task.isSuccessful()) {
final FirebaseUser user = mAuth.getCurrentUser();
dataRef.addListenerForSingleValueEvent(new ValueEventListener() {
@Override
public void onDataChange(DataSnapshotdataSnapshot) {
if (dataSnapshot.hasChild(user.getUid())) {
checkVerification(user);
} else {
Toast.makeText(MainActivity.this, "Not Registered as " + usertype.toUpperCase(),
Toast.LENGTH_SHORT).show();
mAuth.signOut();
}
}

@Override
public void onCancelled(DatabaseErrordatabaseError) {
}
                                        });
                                      } else {

Toast.makeText(MainActivity.this, task.getException().getLocalizedMessage(),
Toast.LENGTH_SHORT).show();
}
}
})
.addOnFailureListener(new OnFailureListener() { @Override
public void onFailure(@NonNull Exception e) {
mProgressDialog.dismiss();
Toast.makeText(MainActivity.this, e.getLocalizedMessage(),
Toast.LENGTH_SHORT).show();
}
});

}
}
});
builder.setNegativeButton(android.R.string.cancel, new
DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog, int which) { dialog.cancel();
}
});

registerButton.setOnClickListener(new View.OnClickListener() { @Override
public void onClick(View v) {
showUserRegister(user);
}
});

builder.show();
}

private void checkVerification(FirebaseUser user) { //verification turned off for
testing purpose

//                  if (user.isEmailVerified()) { Toast.makeText(MainActivity.this,
"Successfully Signed in.",
Toast.LENGTH_SHORT).show(); showHome(user.getUid());

//          } else {
//                        Toast.makeText(MainActivity.this, "Please Verify Your Email",
//                      Toast.LENGTH_SHORT).show();
//                      user.sendEmailVerification();
//                      mAuth.signOut();
//          }

private void showHome(String uid) {

SharedPreferenceManager.getInstance(this).setUserId(uid); if
(usertype.equals("worker")) {
SharedPreferenceManager.getInstance(this).setUserType("worker");
startActivity(new Intent(MainActivity.this, WorkerHomeActivity.class));
} else {
SharedPreferenceManager.getInstance(this).setUserType("user");
startActivity(new Intent(MainActivity.this, UserHomeActivity.class));
}
}


private void showUserRegister(String user) {
Intent intent = new Intent(getBaseContext(), RegisterActivity.class);
intent.putExtra("USER_TYPE", user); startActivity(intent);
}
}
```

5.7 Execution of application

The execution procedures are screens are given here

Main Menu:

This is the menu bar for our existing application. From this, we can go and use the various the activity of this app.

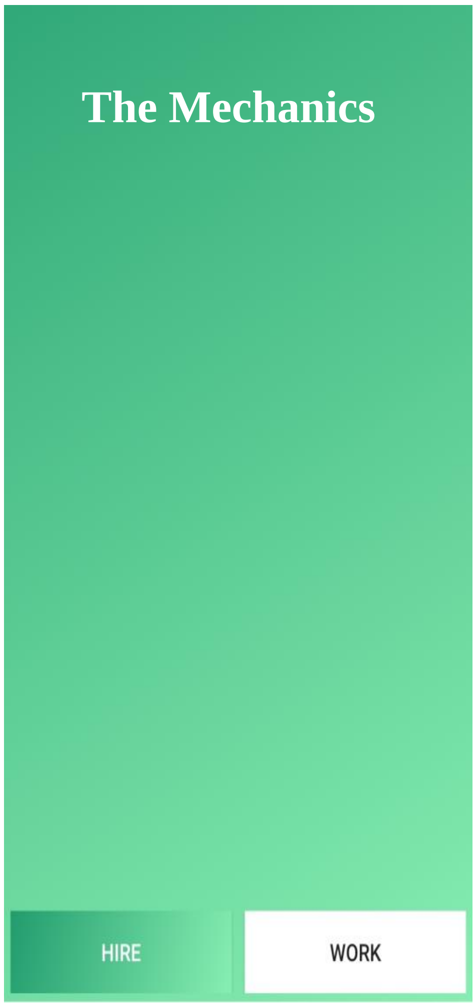


Figure 5.7: Main Menu

User Login Activity:

On this window, we get to see the login activity of users.

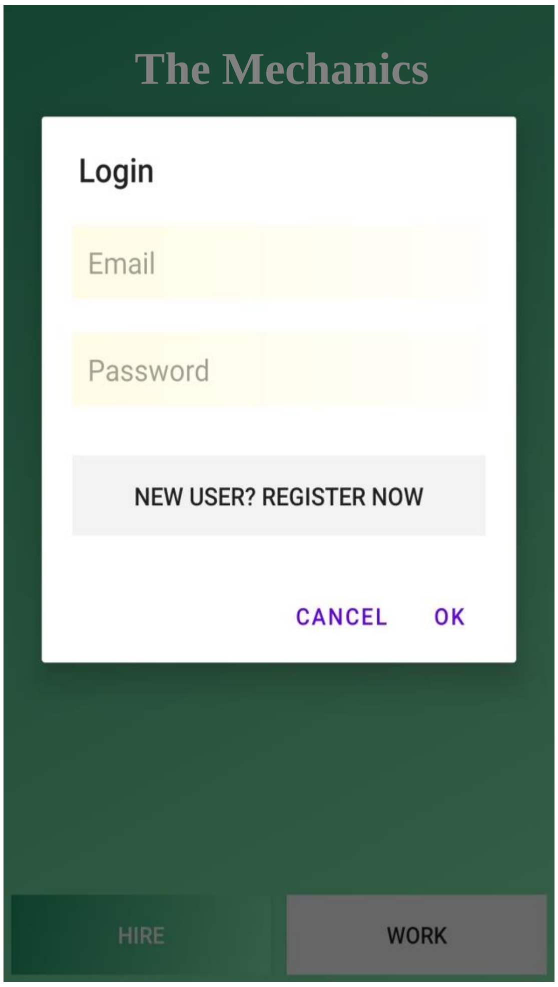


Figure 5.7.1: Login Activity

User Registration Activity:

In this window, we get to see the registration activity of users.

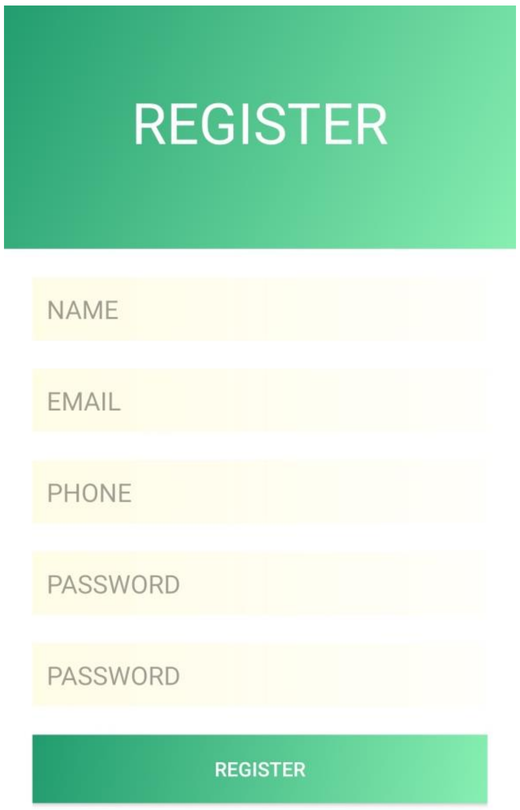


Figure 5.7.2: Registration Activity of User

Mechanics Registration Activity:

On this window, we get to see the registration activity of the Mechanics.

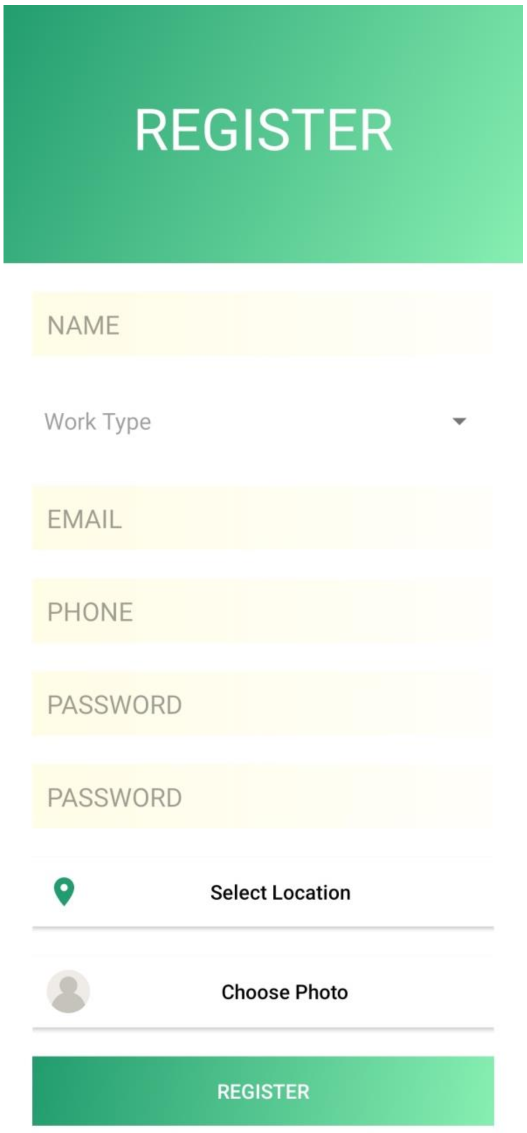


Figure 5.7.3: Registration Activity of mechanics

User Home Activity:

On this window, we get to see the services button and select the location button. By selecting services and selecting locations we get to see the Mechanics lists.

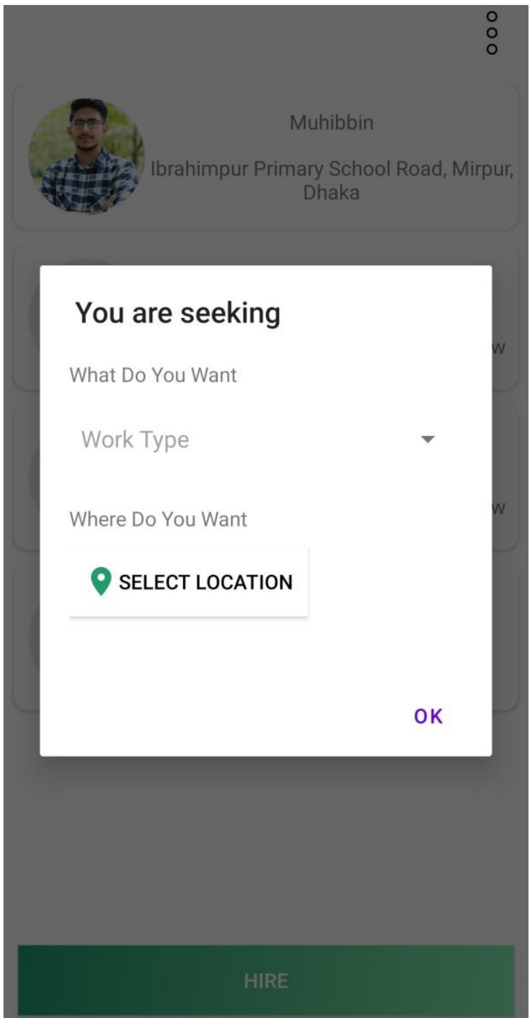
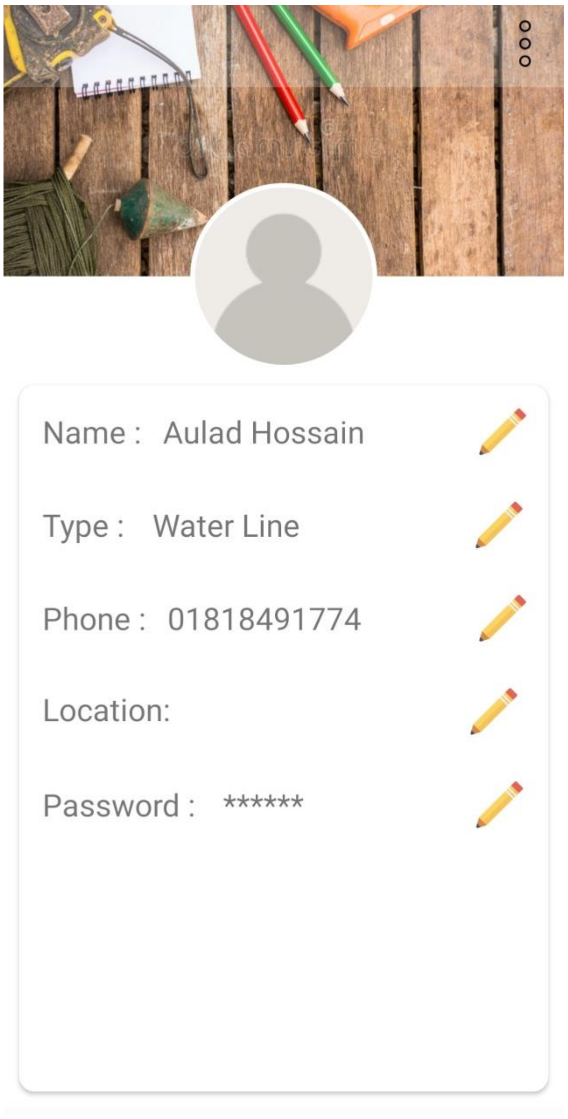


Figure 5.7.4: User Home Activity

Mechanics Home Activity:

On this window, we get to see the Mechanics home activity.



Name : Aulad Hossain

Type : Water Line

Phone : 01818491774

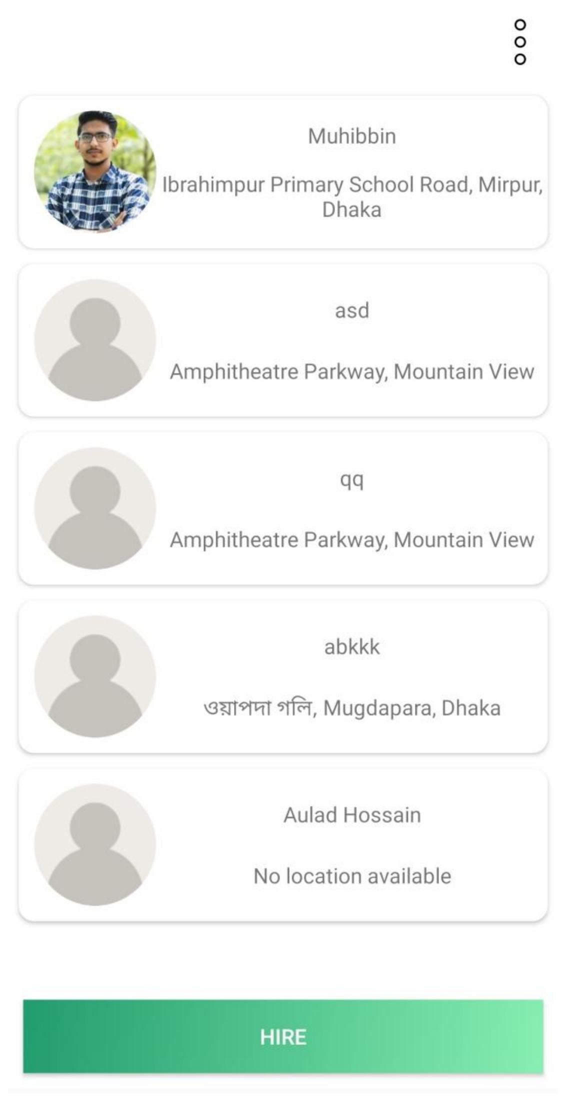
Location:

Password : ******

Figure 5.7.5: Mechanics Home Activity

Mechanics List Activity:

On this window, we get to see the list of the Mechanics. Click the hire button to hire an Mechanics.


Muhibbin

Ibrahimpur Primary School Road, Mirpur, Dhaka

asd

Amphitheatre Parkway, Mountain View

qq

Amphitheatre Parkway, Mountain View

abkkk

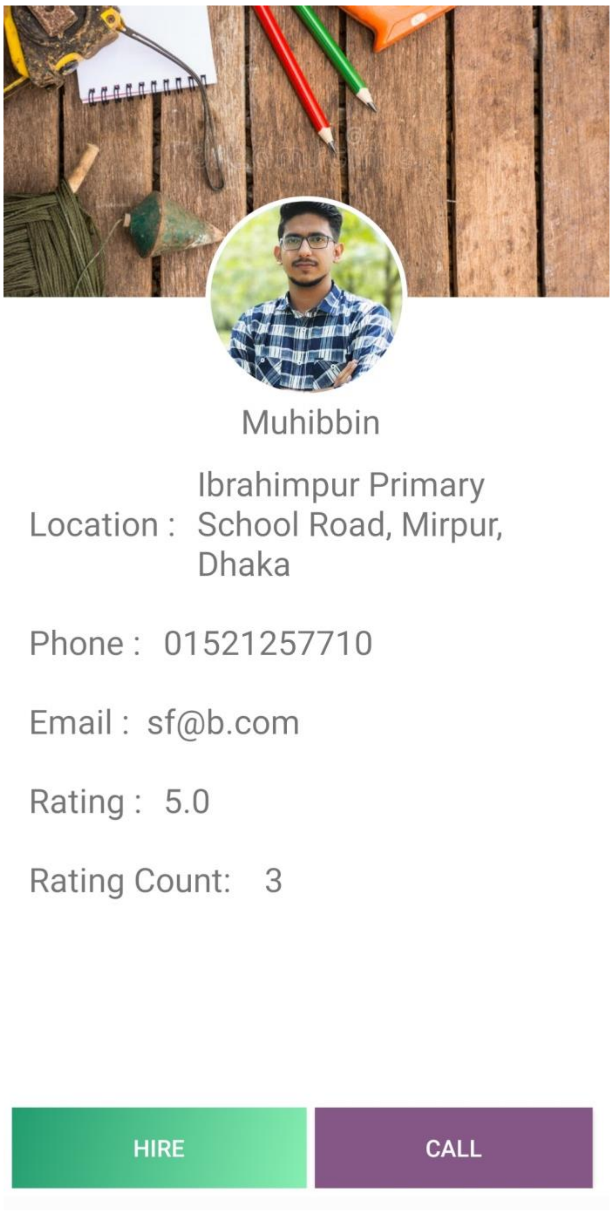
ওয়াপদা গলি, Mugdapara, Dhaka

Aulad Hossain

No location available

Figure 5.7.6: Mechanics List Activity

Mechanics Details Activity:


Muhibbin

Location : Ibrahimpur Primary School Road, Mirpur, Dhaka

Phone : 01521257710

Email : sf@b.com

Rating : 5.0

Rating Count: 3

Figure 5.7.7: Mechanics Details Activity

Chapter -6

Future Work

CHAPTER -6
FUTURE WORK

6.1 Drawback

If we use this application reduces time. Only authentic data is provided. User can access data from one reliable platform. We tried our best for this project by giving maximum effort to help the people. Hopefully, it will be the best supportive app for future use and we will upgrade this app's activity day by day.

Still, it has some drawbacks too. At testing phase, we faced some of problems that we need to discuss here, which should be overcome in the next version. These facts are outlined briefly here-

# Requires an active internet connection.

# System will provide inaccurate results if data not entered properly.

# There needs to add more features to increase the app performance.

6.2 Future Work

- In future we will improve the effectiveness.
- We will add new features as and when it required.
- In future we must overcome this limitations by using modern technologies.

6.3 Conclusion

In this paper, we concentrated on the Mechanics at your Door using the appropriate technique and tools. We have studied the past works and to the best of our knowledge. Hope our efforts will help to take the research on Mechanics at your Door more ahead.

REFERENCES

[1] Project Idea **http://nevonprojects.com/project-ideas/android-project-ideas/**

[2] **http://www.javatpoint.com/android-tutorial**

[3] **http://ieeexplore.ieee.org/document/6104696/**

[4] **https://en.wikipedia.org/wiki/Android(operating system)**

[5] **https://en.wikipedia.org/wiki/Mobile app**

[6] **https://scholar.google.com/scholar?q=android%20applications%20life%20cycle%20 papers&hl=en&as_sdt=0&as_vis=1&oi=scholar**