A

PROJECT REPORT

ON

GYM DATA MANAGER

Submitted in Partial Fulfillment of the Requirements for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

Submitted By:
Name : Abhinav chauhan
Ashutosh Vishwakarma

University Roll Number: 1708551748

And 1708551787

Project Guide:
Mr. Pramod Sharma

Designation
BCA Dept., SRC

DEPARTMENT OF BCA
SHRI RAM COLLEGE
MUZAFFARNAGAR

Batch 2017-2020

Affiliated by Choudhary Charan Singh University , Meerut

SHRI RAM COLLEGE

TABLE OF CONTENT

DECLARATION

We Abhinav Chauhan and Ashutosh Vishwakarma hereby declare that the project report title " furniture management system" is an original work carried out by us under the supervision of Mr. Sanjaykant Tyagi Sir .

I further declare that this work has not been submitted to any other Institute/University

for the award of the degree of Bachelor of Computer Applications.

Student Name:Abhinav chauhan
Roll No.1708551748
And :Ashutosh Vishwakarma
Roll No.1708551787

Date:

FORWARDING LETTER

This is to certify that the project entitled "GYM DATA MANAGER" which is being submitted for the partial fulfillment for the award of Degree of Bachelor of Computer Applications from Ch. Charan Singh University , Meerut i s an authentic work carried out by Mr. Abhinav Chouhan (University Roll No. 1708551748) and Mr. Ashutosh Vishwakarma under the guidance of Project Guide Mr. Pramod Sharma.

The matter embodied in this project work has not been submitted earlier for the award of degree or diploma. We wish him/her all the best for the future.

Internal Guide **Head - BCA Department**

ACKNOWLEDGEMENT

This project report on "GYM DATA MANAGER" is the result of the suggestions to me by Mr. Pramod Sharma

I have received unfailing encouragement of Mr. Pramod Sharma whose exceptional knowledge and unparalleled behavior is full of ardent inspiration in it. However, we can never adequate thank all those who have their assistance, guidance, cooperation, contributed to the improvement of this report.

I am ebullient in expressing my intense in debtless heartiest gratitude to all of them.

Since performance feedback is essential for effective communication, mistakes and creative feedback of the report may be unhesitatingly communicated to me, which will be as far as possible duly acknowledged and most welcome.

In this report, whatever is beneficial comes from almighty, and whatever is faulty is mine.

Date

Submitted By

Abhinav Chouhan

Ashu Vishwakarma

Ch 1 Introduction

1. 1 Scope of System

All organizations rely on computer and information technology to conduct business and operate more efficiently. The rapid spread of technology across all industries has generated a need for highly trained workers to help organizations incorporate new technologies. The tasks performed by workers known as computer systems analysts evolve rapidly, reflecting new areas of specialization or changes in technology, as well as the preferences and practices of employers. Computer systems analysts solve computer problems and apply computer technology to meet the individual needs of an organization. They help an organization to realize the maximum benefit from its investment in equipment, personnel, and business processes.

Systems analysts may plan and develop new computer systems or devise ways to apply existing systems' resources to additional operations. Systems analysts discussed the systems problems with managers and users to determine its exact nature.
Defining the goals of the system and dividing the solutions into individual steps and separate procedures, systems analysts use techniques such as structured analysis, data modeling, sampling, and cost accounting to plan the system.
They specify the inputs to be accessed by the system, design the processing steps, and format the output to meet users' needs. They also may prepare cost-benefit and return-oninvestment analyses to help management decide whether implementing the proposed technology will be financially feasible.

1.2 Project Description

1.2.1 About Existing System

The existing system is a manual in which the Gym owner keeps all the records in a notebook due to which it becomes hard to find members names every time and he has no way of knowing whose fee is due except the intuition or looking it in the notebook every time the process becomes very difficult and time consuming on top of that the notebook may be stolen or lost manual calculations can be wrong also one notebook can't keep the records for years .

At some point notebook is going to be full then he has to buy new one which Is extra cost also the responsibility to keep the old one safe.

1.2 IMPLEMENTATION OF PROPOSED SYSTEM

The proposed system is trying to solve all the problems of existing system by means of automating the most work.

In addition it provides analytics of fees records via a chart also it saves the pictures of members of which is not even possible with the notebook approach.

The proposed system is the implementation of JAVA programming language and android framework which uses firebase firestore realtime database.

With the help of realtime database the data is never lost even though the app may be uninstalled device may be lost but the data is immortal as long the user has credentials to login.

The proposed system will give flexible and user-friendly solution for maintaining all the details of fees and member information with Very nice U.I.

This system simplifies the task of the user by performing calculations automatically.
Moreover the new features can be added any time as per the requirements

1.3 ADVANTAGES OF PROJECT

The proposed system is going to save the Gyms time and money with more accuracy and easy access to the data . system is going to automatically manage all the sales , purchase etc records with the corresponding data.
The user can also search the records for any particular member, we are also providing a very user friendly and vivid user interface which is very easy to learn and work with.

Ch 2 : Project category, tools and Environment

2.1 Project Category

This Project is an android Application

2.2 Front end Coverage

The front end of the system is completely designed with android widgets and external libraries like androidPChart , cardview etc.

2.3 Back end Coverage

The back end of the system is designed with the Firebase Firestore which is a realtime nosql database is used for the storage purpose. Cloud functions are used for deleting account , resizing images , and firebase security rules are used to the security of the database

2.4 Software and Hardware Requirements

The minimal Hardware Requirements for smoothly running the system is as follows

Processor ---------- any processor
RAM ---------- 512 MB
Display Type ---------- Touch

Internet ---------- required

The minimal software requirement for smooth functioning of the system is as follows

Platform ---------- Android
Android version -------- Kitkat
API level ---------- 21

Ch 3 : Project Development States

3.1 Recognition of needs

Technically, data is otherwise known as facts.
The facts or data collected are properly recorded during the analysis phase and are further used in the project.
Thus Fact-finding is the process of collecting the appropriate and needed data for the project.
Fact-finding involves gathering of related information for the proposed system. In general, there are various methods used for collecting data or facts.
Some of them are:

Interviews with the owner and

members Review of written documents.

Questionnaires

Study of the current Manuals

Observation of the functioning of

the Organization Sampling

Record

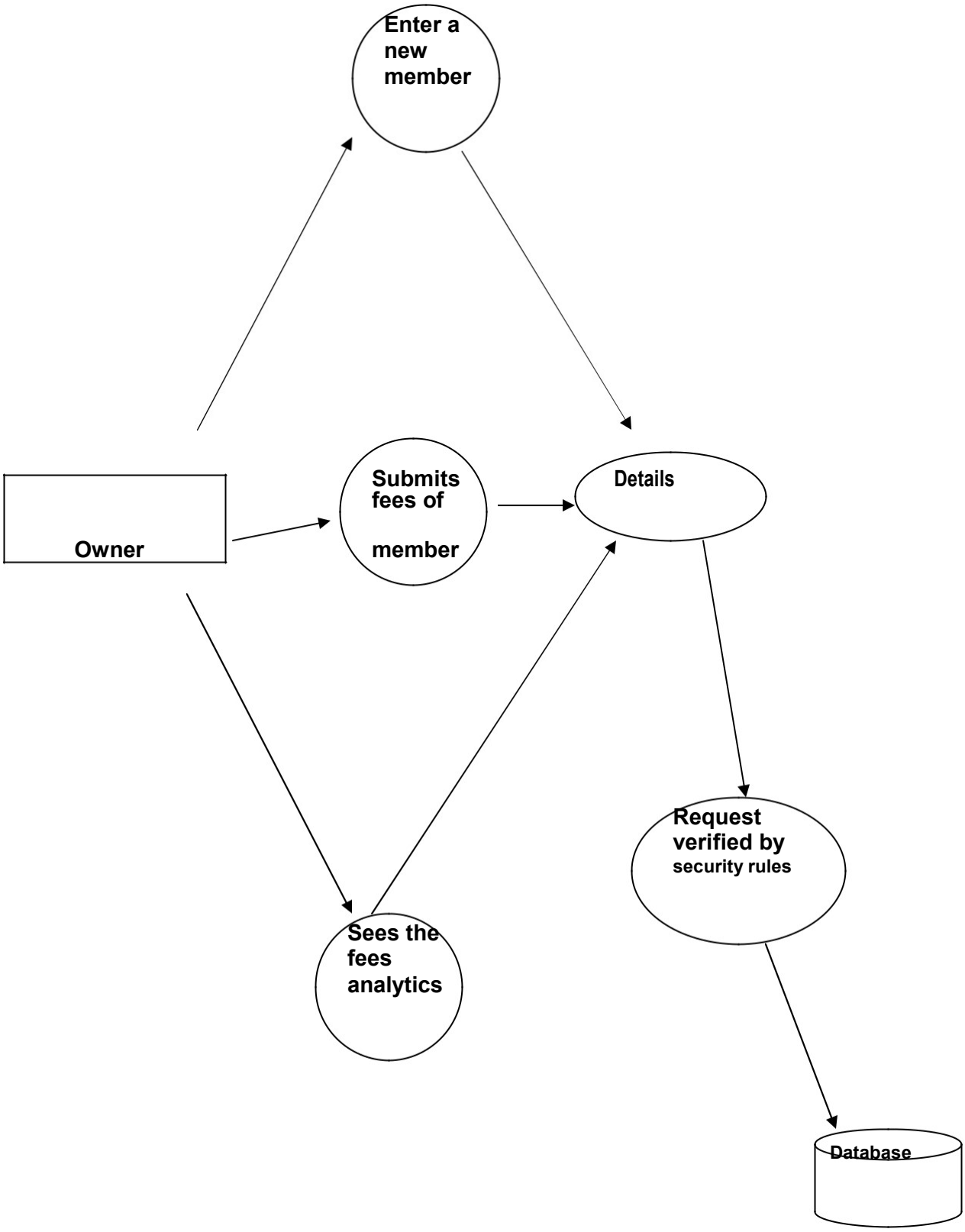
Searching Onsite

Observation

3.2 Feasibility Study

Feasibility study is done by examining the cost of current system and effort going in maintaining the records and the problems arriving in the current scenario and then comparing it to the benefits which the new system can bring in after analyzing everything we came to the conclusion that building and android app is the best solution because and computer is not very portable and in needs space in the gym and everybody doesn't know how to use it on the other hand an android app can be used any time anywhere by anybody

3.3 DFD


Enter a new member
Owner
Submits fees of member
Details
Request verified by security rules
Sees the fees analytics
Database

.5 System Development

System development is done in many different ways. It forms the basis of all methodologies. The approach that is being implemented for this project is structured approach. Structured programming, structured analysis, structured design are the technique for structured approach. This is implemented for this system development.

Structured programming is one that begins with one beginning and one ending, and each step in the program execution consists of one of the three programming constructs. One of the concepts of structured programming is implemented in this project.

(i.e.) top down approach is implemented. Through this complex programming is divided into hierarchy of modules.

Two main principles of structured design are the program module should be designed so they are loosely coupled or highly cohesive out of which highly cohesive is being used.

Along with it the strategy design pattern is used.

Structured analysis defines system-processing requirements by identifying by all of the events that will cause a system to react in someway. Each event leads to a different system activity. These activities are then taken and data flow diagram is created showing the processing details including inputs and outputs. The most important development activity is preparation of computer program needed for the system. The system flowcharts, input charts, output charts, are transferred into program. In each stage of preparation, the program has been tested

and errors are corrected if any. All accuracy measures falls

into account while testing the program.

While preparing the program, to avoid the error message, if one button is functioning for particular record might be formatted, as other has been enabled. The change over method is the process where the existing manual system is to be replaced by the new computerized system.
The following changes are made during the change over plan, Change over plan has to be made carefully, so as to minimize the problem that may arise from human errors.
The activities to be performed during the change over plan have

to be

identified and the responsibilities should be assigned

to individuals in the organizations.

3.6 Testing implementation and maintainance

3.6.1 TESTING

Software testing is the last phase of the software development cycle. Testing is very important for the success of a system.
System testing makes a logical assumption that if all parts of the system are correct, then the goal has been achieved.

The testing should be done at the end of all development steps. Even though the final testing and verification are inevitable for better life and functionality of the software.

The major phases in testing are design of test plan, setting up test case and test candidate and test procedure, testing and correction. This is a cycle process and the software will circulate through all the steps till it attains the required quality.
The testing is carried in the following steps.

1. **Unit Testing**
2. **Validation Testing**
3. **System Testing**
4. **Acceptance Testing**
5. **Regression Testing**
6. **Database Testing**

1.Unit Testing

Unit testing refers testing of all the individual programs. This is sometimes as program testing. This test should be carried out during programming stage in order to find the errors in coding and logic for each program in each module. Unit test focuses verification effort on the smallest unit of software design module. In this project, the user must fill each field otherwise the user to enter values.

2.Validation Testing

Valid and invalid data should be created and the program should be made to process this data to catch errors.
When the user of each module wants to enter some date then validation for correct name , price are validated for example if an alphabet is entered in the price field a error messages pops up saying invalid input ,

pops with saying you didn’t sell anything in other situation user may try to

sell something that is not available in that case also error message shows up.
Here the inputs given by the user are validated.

3 Input Testing

Here system is tested with all variable combination of inputs. User may type data in situations like entering password, numerical details etc. The system is tested with all the cases and it responded with appropriate error messages.

4 .Output Testing

Here the output is tested to view whether that the screen is what which is desired. It is also checked whether it is to the satisfaction of the user. Changes that need to be done can be done after the result is seen.

3. System Testing

System testing is used to test the entire system (Integration of all the modules). It also tests to find the discrepancies between the system and the original objective, current specification and system documentation. The entire system is checked to correct deviation to achieve correctness.

4. Acceptance Teasing

Acceptance testing is performed on a collection of business functions in a production environment and after the completion of functional testing. This is the final stage in the testing process before the system is accepted for operational use. This testing should be done with original data and with the presence of the users. This test confirms the system ready for production.

5. Regression Testing

Regression testing refers to the retesting components / functionality of the system to ensure that they function properly even after and change has been made to parts of the system. As detects are discovered in a component, modification are made to correct them.

6. Database Testing

The purpose of database testing is to determine how well the databases are meets requirements. This is an ongoing process because no database is static. When table is created, a mirror of the same should be created and stored. The original alone and its mirror images go through the various tests.

This process continues until changes can be implemented in the original table

IMPLEMENTATION

Implementation is the phase where the developed component is installed in the working place. The operation of the software was monitored and the results were recorded.

Implementation is the stage of the project where the theoretical design is turned into a working system. This involves careful planning, investigation of the current system and it's constrains on implementation, design of methods to achieve the change over and evaluation of change over methods.

The problems encountered are converting files training users, creating accurate files and verifying print outs for integrity. The objective is to put the tested system into operation while holding costs, risks and personnel irritation to a minimum. It involves creating computer compactable files, training the operational staff and installing terminals and hardware.

Maintenance activities begin where conversion leaves off. Maintenance is handled by the same planning and control used in a project. Maintenance can be classified as corrective, adaptive or perceptive. Corrective measures means repairing process of performance failures or making changes because of previously in corrected problems or false assumption. Adaptive Maintenance means changing the program functions. Perceptive Maintenance means enhancing the performance or modifying the programs to respond to the user's addition or changing needs.

The implementation view of software requirements presents the real world manifestation of processing functions and information structures. In some cases, physical representation is developed as the first step in software design. The analyst must recognize the constraints imposed by the pre defined system elements and consider the implementation view of the function and information when such view is appropriate.

CODING

Code base for this project can be found on the following Github Repository

https://github.com/AbhinavChauhan97/Gym_Data_Manager/tree/Dividing_fee_records_in_months%26years

```
package com.bignerdranch.practice.Activities;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager; import
androidx.fragment.app.FragmentPagerAdapter; import
androidx.recyclerview.widget.RecyclerView; import
androidx.viewpager.widget.ViewPager; import
android.content.Intent;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import com.bignerdranch.practice.Fragments.MembersNamesRecyclerViewFragment;
import com.bignerdranch.practice.Model.Member;
import com.bignerdranch.practice.R;
import com.google.android.material.bottomnavigation.BottomNavigationView; import
com.google.android.material.floatingactionbutton.FloatingActionButton; import
com.bignerdranch.practice.Fragments.AllMembersRecordsFragment; public class
MainActivity extends AppCompatActivity implements
MembersNamesRecyclerViewFragment.CallBacks {
    private int MEMBER_INFO_REQ = 1;
    private boolean changed;
    private ViewPager mViewPager;
    private MembersNamesRecyclerViewFragment membersNamesRecyclerViewFragment;

    private AllMembersRecordsFragment allMembersRecordsFragment;

    @Override
    public void onMemberNamesPressed(Member member) {
        Intent intent = new Intent(this, MemberInfoActivity.class);
        intent.putExtra("Member", member);

    } startActivityForResult(intent,MEMBER_INFO_REQ);

    @Override

    public boolean isDataChanged() {

    }    return true;

    @Override
                                                                    data) {
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent
super.onActivityResult(requestCode, resultCode, data);
        if(requestCode == MEMBER_INFO_REQ)
        {
            changed = true;

    }
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        membersNamesRecyclerViewFragment =
```

```
MembersNamesRecyclerViewFragment.newInstance();
            allMembersRecordsFragment = AllMembersRecordsFragment.newInstance();
            if(savedInstanceState == null)
            {

getSupportFragmentManager().beginTransaction().add(R.id.list,membersNamesRecyclerView
Fragment).commit();
            }
            setContentView(R.layout.entry_screen);
            setBottomNavigationView();

      }     setAddMemberFab();

      private void setAddMemberFab() {
            FloatingActionButton mAddMemberFab;
            mAddMemberFab = findViewById(R.id.add_new_member_fab);
            mAddMemberFab.setOnClickListener(new View.OnClickListener() {
                  @Override
                  public void onClick(View v) {
                        startActivity(new Intent(MainActivity.this,
AddNewMemberActivity.class));

                  }
      }     });

      private void setBottomNavigationView() {
            BottomNavigationView bottomNavigationView =
findViewById(R.id.entry_screen_bottom_navigation);

            bottomNavigationView.setOnNavigationItemSelectedListener(new

BottomNavigationView.OnNavigationItemSelectedListener() {
                  @Override
                  public boolean onNavigationItemSelected(@NonNull MenuItem item) {


                        FragmentManager fragmentManager = getSupportFragmentManager();
                        switch (item.getItemId()) {
                              case R.id.navigate_to_all_members:

if(fragmentManager.findFragmentById(R.id.navigate_to_all_members) == null) {
                                          fragmentManager.beginTransaction()
                                                .replace(R.id.list,
membersNamesRecyclerViewFragment).commit();
                                    }
                                    return true;
                              case R.id.navigate_to_past_records:


if(fragmentManager.findFragmentById(R.id.navigate_to_past_records) == null) {
                                          fragmentManager.beginTransaction()
                                                .replace(R.id.list,
allMembersRecordsFragment).commit();
                                    }
                                    return true;

                        }     default: return false;


                  }
            });
```

```
    }
}
```

```
package com.bignerdranch.practice.Fragments;

import com.bignerdranch.practice.Activities.*;

import android.content.ContentResolver; import

android.content.Intent;

import android.content.pm.PackageManager;
import android.content.pm.ResolveInfo;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.provider.MediaStore;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.core.content.FileProvider;
import androidx.fragment.app.Fragment;

import com.bignerdranch.practice.Model.Member;
import com.bignerdranch.practice.R;
import com.bignerdranch.practice.database.MembersManager; import

com.google.android.material.textfield.TextInputEditText; import

com.squareup.picasso.Picasso;

import java.io.File;
import java.security.Provider;
import java.text.DateFormat;
import java.util.Date;
import java.util.List;

public class AddNewMemberFragment extends Fragment {
    private TextInputEditText mNewMemberName; private
    TextInputEditText mNewMemberPhone; private
    TextInputEditText mNewMemberAddress; private
    TextInputEditText mNewMemberJoiningDate; private
    ImageButton mClickImage;
    private ImageButton mAddThisMember;
    private ImageButton mDontAddThisMember;
    private ImageView mNewMemberPhoto;
    private File mPhotoFile;
    private boolean mImageTaken;
    private int REQ_TAKE_PHOTO = 123;
    private Uri imageFileUri;
    public static AddNewMemberFragment newInstance() {
```

```
        AddNewMemberFragment fragment = new

    AddNewMemberFragment(); } return fragment;

    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup
container, @Nullable Bundle savedInstanceState) {
        super.onCreateView(inflater, container, savedInstanceState);

    } return inflater.inflate(R.layout.add_new_member,container,false); @Override



    public void onViewCreated(@NonNull View view, @Nullable Bundle
savedInstanceState) {
        super.onViewCreated(view, savedInstanceState);
        mNewMemberName = view.findViewById(R.id.member_name_textinput_edittext);
        mNewMemberPhone = view.findViewById(R.id.member_phone_textinput_edittext);
        mNewMemberAddress =
view.findViewById(R.id.member_address_textinput_edittext);
        mNewMemberPhoto = view.findViewById(R.id.new_member_circularimageview);
        mNewMemberJoiningDate =
view.findViewById(R.id.member _joiningdate_textinput_edittext);
        mNewMemberJoiningDate.setText(DateFormat.getInstance().format(new Date()));
        setAddThisMemberListener(view);
        setmDontAddThisMember(view);

    }    setClickImageImageButton(view);

    @Override
    public void onStart() {
        super.onStart();

    }
    private void setClickImageImageButton(View view){
        mPhotoFile = MembersManager.getInstance(getActivity()).getPhotoFile();
        mClickImage = view.findViewById(R.id.click_image);
        mClickImage.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent captureImage = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                imageFileUri =
FileProvider.getUriForFile(getActivity(),"com.bignerdranch.practice.provider",mPhotoF
ile);
                captureImage.putExtra(MediaStore.EXTRA_OUTPUT,imageFileUri);
                List<ResolveInfo> cameraActivities =
getActivity().getPackageManager()
                        .queryIntentActivities(captureImage,
PackageManager.MATCH_DEFAULT_ONLY);
                for(ResolveInfo activity : cameraActivities) {

getActivity().grantUriPermission(activity.activityInfo.packageName,imageFileUri,
Intent.FLAG_GRANT_WRITE_URI_PERMISSION);
                }
                    startActivityForResult(captureImage,REQ_TAKE_PHOTO);

            }
```

```
        });
    }
    private void setAddThisMemberListener(View view){

        class AddMemberTask extends AsyncTask<Member,Void,Void>{

            @Override
            protected Void doInBackground(Member... members) {
                MembersManager.getInstance(getActivity()).addMember(members[0]);

                return null;
            }

            @Override
            protected void onPostExecute(Void aVoid) {
                Toast.makeText(getActivity(), "member added",
Toast.LENGTH_LONG).show();

                AddNewMemberFragment.this.getActivity().finish();
            }
        }
        mAddThisMember = view.findViewById(R.id.add_this_member);
        mAddThisMember.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String name =    mNewMemberName.getText().toString().trim();
                String phone    = mNewMemberPhone.getText().toString().trim();
                String address = mNewMemberAddress.getText().toString().trim();
                if (name.length() > 0 && phone.matches("(0/91)?[7-9][0-9]{9}") &&
address.length()> 0)

                    Member newMember = new Member(name, phone, address);
                    if(mImageTaken){

                    }    newMember.setMemberImageFileUri(imageFileUri.toString());
                    else{

                    }    newMember.setMemberImageFileUri(null);
                    new AddMemberTask().execute(newMember);

                }
                else Toast.makeText(getActivity(),"invalid
input",Toast.LENGTH_LONG).show();

            }
        });
    }
    private void setmDontAddThisMember(View view){
        mDontAddThisMember = view.findViewById(R.id.cancel_adding_this_member);
        mDontAddThisMember.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(mImageTaken) {
                    // Uri uri = FileProvider.getUriForFile(getActivity(),
"com.bignerdranch.practice.provider", mPhotoFile);
                    ContentResolver resolver = getActivity().getContentResolver();

                }    resolver.delete(imageFileUri, null, null);

                AddNewMemberFragment.this.getActivity().finish();
```

```
            }
        }    });


    @Override
    public void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
if(requestCode == REQ_TAKE_PHOTO)

            {
                mImageTaken = true;
                Picasso.with(getActivity()).load(  imageFileUri)

                        .fit().centerInside().into(mNewMemberPhoto);
            }
        }
 }


package com.bignerdranch.practice.Fragments;

import android.content.Context;

import android.content.Intent;
import android.database.Cursor;
import android.graphics.drawable.Drawable;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.view.animation.AlphaAnimation;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.widget.SearchView;
import androidx.core.view.MenuItemCompat;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.DefaultItemAnimator;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.bignerdranch.practice.Activities.MainActivity;
import com.bignerdranch.practice.Activities.PreferenceActivity; import
com.bignerdranch.practice.Dialogs.MemberNotFoundDialog; import
com.bignerdranch.practice.Model.Member;
import com.bignerdranch.practice.R;
import com.bignerdranch.practice.database.MembersManager;

import com.mikhaellopez.circularimageview.CircularImageView;

import com.squareup.picasso.Picasso;
```

```
public class MembersNamesRecyclerViewFragment extends Fragment {


    private RecyclerView mMembersNameRecyclerView;
    private CallBacks mCallBacks;
    private Adapter mAdapter;
    private Member[] mALlMembers;
    private final String MEMBER_NOT_FOUND_DIALOG = "Member Not Found";
    private boolean c ;
    public interface CallBacks{
        void onMemberNamesPressed(Member member);

    }       boolean isDataChanged();
    public static MembersNamesRecyclerViewFragment newInstance() {
        MembersNamesRecyclerViewFragment fragment = new
MembersNamesRecyclerViewFragment();

    }     return fragment;
    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setHasOptionsMenu(true);
    }

    @Override
    public void onAttach(@NonNull Context context) {
        super.onAttach(context);

        mCallBacks = (CallBacks)context;
    }

    @Override
    public void onDetach() {
        super.onDetach();

        mCallBacks = null;
    }
    @Override
    public void onPause() {
        super.onPause();

    }

    @Override
    public void onResume() {
        super.onResume();
        if(mCallBacks.isDataChanged()) {
        mAdapter = new
Adapter(MembersManager.getInstance(getActivity()).getMembers(null));
        mAdapter.notifyDataSetChanged();
          setMembersNameRecyclerView(getView());
          if(mAdapter == null || mAdapter.getItemCount() == 0)

            {
                TextView textView = getView() .findViewById(R.id.empty_message);

                textView.setText("NO MEMBERS TO SHOW");
            }
          }
    }
```

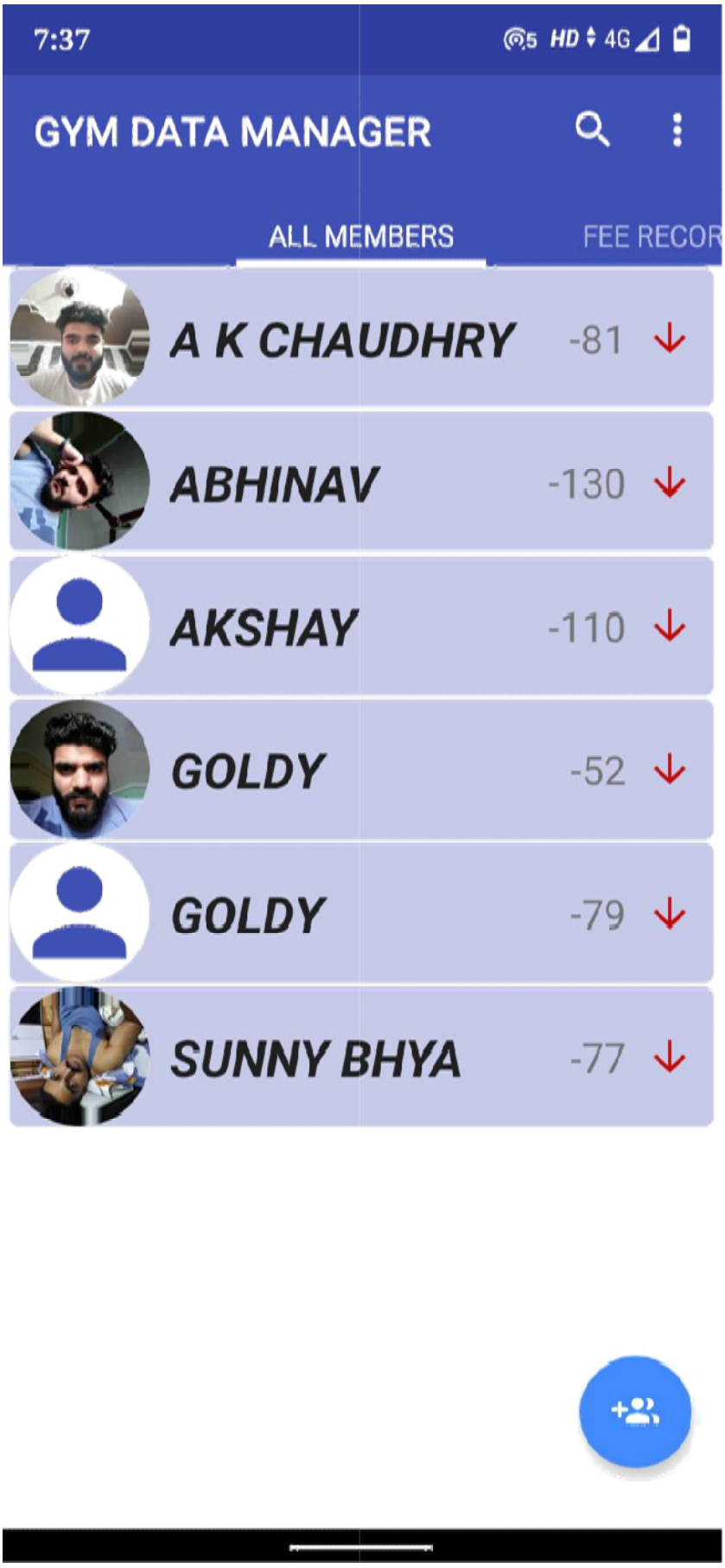
```
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup
container, @Nullable Bundle savedInstanceState) {
            super.onCreateView(inflater, container, savedInstanceState);

    }   return inflater.inflate(R.layout.recycler_view,container,false);

    @Override
    public void onViewCreated(@NonNull View view, @Nullable Bundle
savedInstanceState) {

        super.onViewCreated(view, savedInstanceState); }

    setMembersNameRecyclerView(view);
    @Override
    public void onCreateOptionsMenu(@NonNull Menu menu, @NonNull MenuInflater
inflater) {
        super.onCreateOptionsMenu(menu, inflater);
        inflater.inflate(R.menu.entry_screen_menu,menu);

    }   setSearchView(menu);

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        switch (item.getItemId()){
            case R.id.settings:
                startActivity(new Intent(getActivity(), PreferenceActivity.class)); return true;


        }   default: return     super.onOptionsItemSelected(item);
    }
    private void setMembersNameRecyclerView(View view){
        mMembersNameRecyclerView = view.findViewById(R.id.recycler_view);
        mMembersNameRecyclerView .setLayoutManager(new
LinearLayoutManager(getActivity()));

        new FetchMemberTask().execute();
        mMembersNameRecyclerView.setHasFixedSize(true);

   }

    private void setSearchView(Menu menu){


        MenuItem searchMenuItem = menu.findItem(R.id.searchview);
        final SearchView searchView =
(SearchView)MenuItemCompat.getActionView(searchMenuItem);
        searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {
                for(Member member : mALlMembers){
                    if(member.getMemberName().equalsIgnoreCase(query.trim())){
                        mCallBacks.onMemberNamesPressed(member);

                        return true;
                    }
                }

                new
```

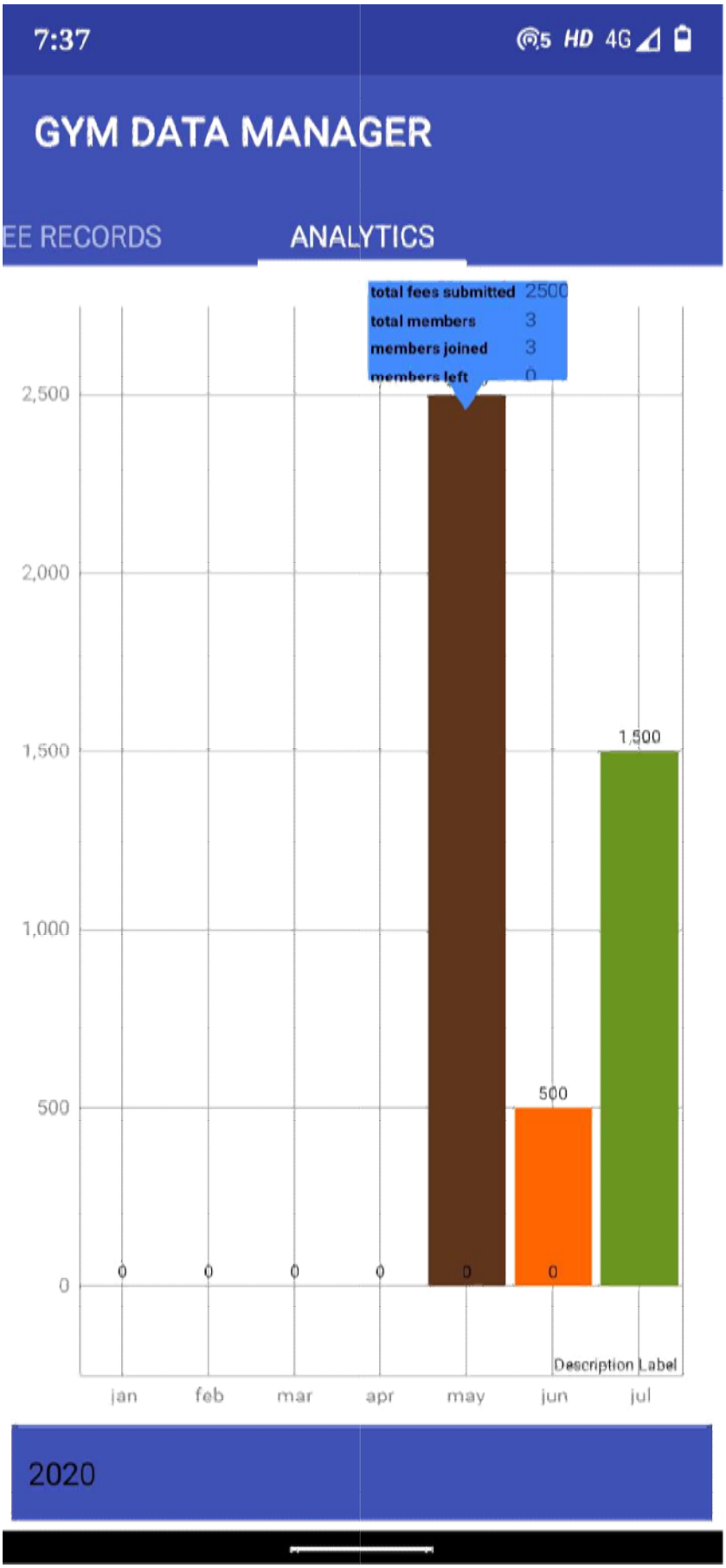
```
MemberNotFoundDialog().show(getFragmentManager(),MEMBER_NOT_FOUND_DIALOG); }

                return false;

                @Override
                public boolean onQueryTextChange(String newText) {

                    return false;
            });   }
            final SearchView.SearchAutoComplete searchAutoComplete =
searchView.findViewById(androidx.appcompat.R.id.search_src_text);
            searchAutoComplete.setDropDownBackgroundResource(android.R.color.white);
            final ArrayAdapter<Member> namesAdapter = new
ArrayAdapter<>(this.getActivity(),android.R.layout.simple_dropdown_item_1line,mALlMem

bers);     searchAutoComplete.setAdapter(namesAdapter);
            searchAutoComplete.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> parent, View view, int position,
long id) {
                    Member member = (Member)parent.getItemAtPosition(position);

                    mCallBacks.onMemberNamesPressed(member);
                }
        }     });
    private class Adapter extends RecyclerView.Adapter<Holder>{

        Adapter(Member[] allMembers){

        }     mALlMembers = allMembers;
        @NonNull
        @Override
        public Holder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
            LayoutInflater inflater = LayoutInflater.from(getActivity());

            return new Holder(inflater,parent);
        }

        @Override
        public void onBindViewHolder(@NonNull Holder holder, int position) {
holder.bind(mALlMembers[position],mALlMembers[position].getDaysLeftForFeeSubmission()

);        }

        @Override
        public int getItemCount() {

        }     return mALlMembers.length;

    }
    private class Holder extends RecyclerView.ViewHolder implements
View.OnClickListener
    {
     private Member mMember; private

        TextView mName; private

        TextView mDaysLeft;
```

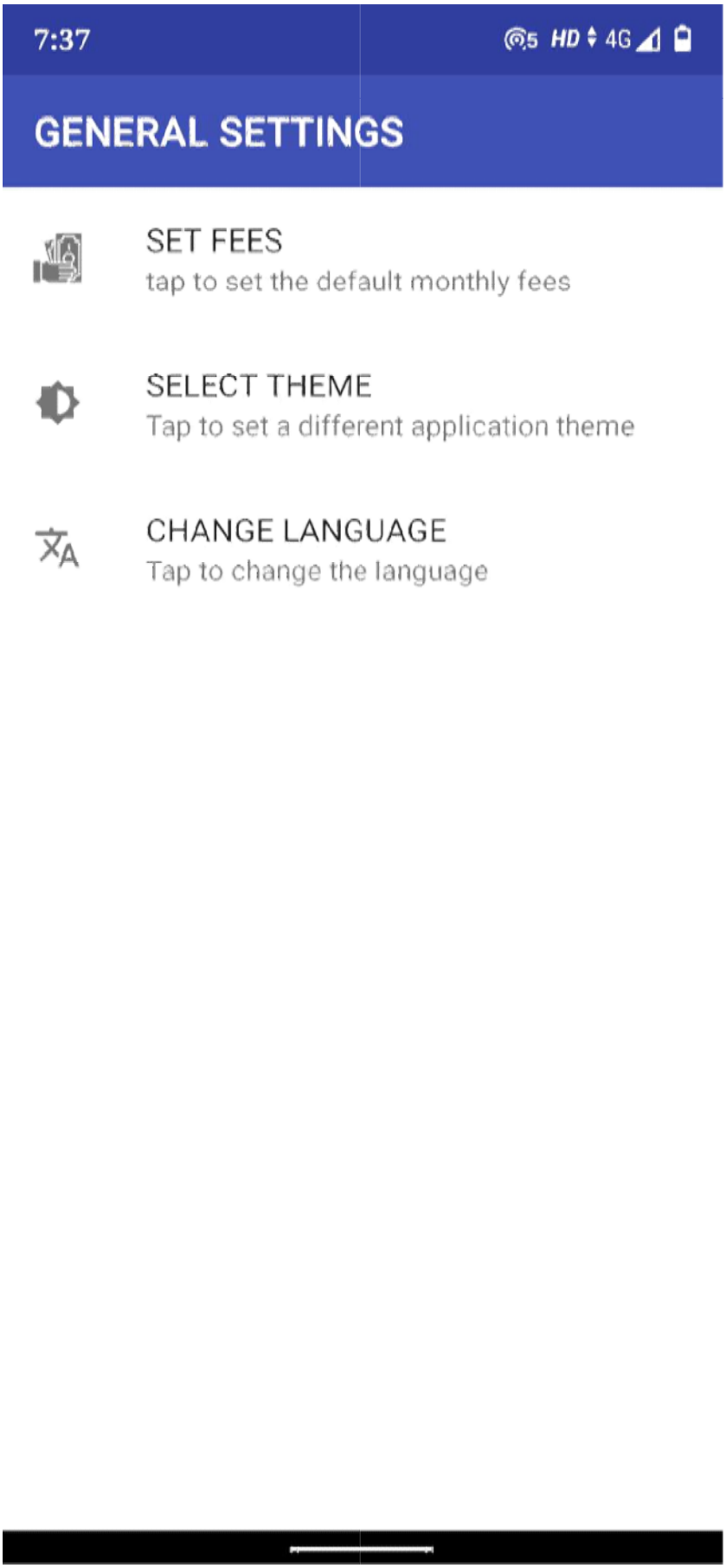
```
            private ImageView mArrow;
            private CircularImageView mThumbnailImage;
            Holder(LayoutInflater inflater,ViewGroup parent){
                  super (inflater.inflate(R.layout.member_name,parent,false));
                  mName = itemView.findViewById(R.id.member_name_textview);
                  mDaysLeft = itemView.findViewById(R.id.days);
                  mArrow = itemView .findViewById(R.id.arrow);
                  mThumbnailImage = itemView.findViewById(R.id.thumbnail_pic);

             }    itemView.setOnClickListener(this);
             void bind(Member member,long daysLeft)

             {      mMember = member;
                    if(daysLeft > 0)
mArrow.setBackground(getActivity().getDrawable(R.drawable.ic_arrow_upward_black_24dp)
);
                   else


mArrow.setBackground(getActivity().getDrawable(R.drawable.ic_arrow_downward_black_24d

p));                mName.setText(member.getMemberName());
                    mDaysLeft.setText(String.valueOf(daysLeft));

Picasso.with(getActivity()).load(member.getMemberImageFileUri()).fit().into(mThumbnai

lImage);       }
          @Override
          public void onClick(View v) {

          }       mCallBacks.onMemberNamesPressed(mMember);
     }
     private class FetchMemberTask extends AsyncTask<Void,Void,Member[]>
     {
        @Override
          protected Member[] doInBackground(Void... voids) {

          }  return MembersManager.getInstance(getActivity()).getMembers(null);


          @Override
          protected void onPostExecute(Member[] allMembers) {
                super.onPostExecute(allMembers);

                mMembersNameRecyclerView.setAdapter(mAdapter = new Adapter(allMembers));
          }
      }
}


package com.bignerdranch.practice.Dialogs;

import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.media.MediaMetadata;
```

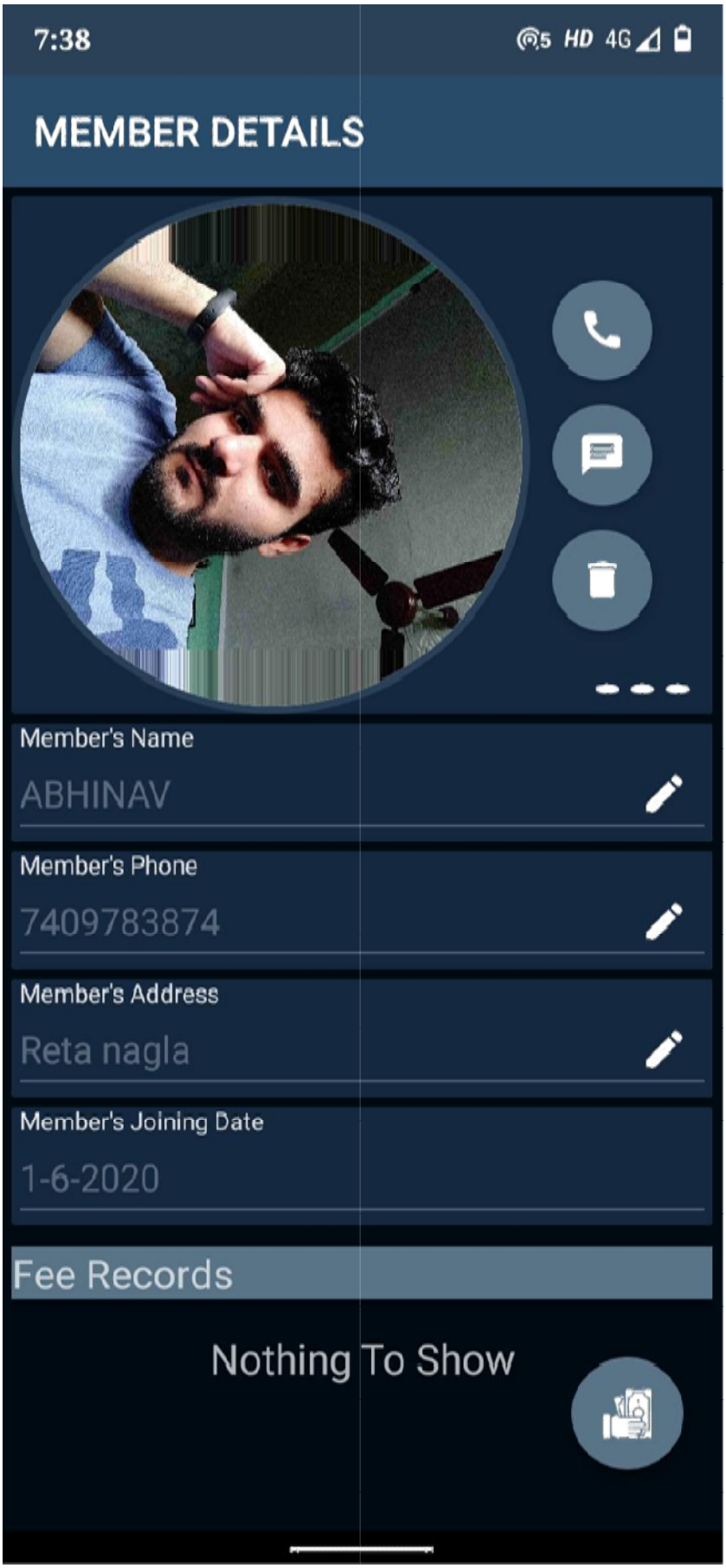
```
import android.os.Bundle;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.DialogFragment;

import com.bignerdranch.practice.Model.Member;
import com.bignerdranch.practice.R;
import com.bignerdranch.practice.database.MembersManager; public

class DeleteConformationDialog extends DialogFragment {


    private Member mMember;
    private int MEMBER_DELETED_RES = 223;
    public static String MEMBER_DELETED = "DELETED" ;
    private DeleteConformationDialog(Member member){

    }   mMember = member;
    public static DeleteConformationDialog newInstance(Member member) {
        DeleteConformationDialog fragment = new DeleteConformationDialog(member);

    }   return fragment;
    @NonNull
    @Override
    public Dialog onCreateDialog(@Nullable Bundle savedInstanceState) {
        super.onCreateDialog(savedInstanceState);
         return new
AlertDialog.Builder(getActivity()).setMessage(getString(R.string.sure_delete,mMember.
getMemberName()))
                                .setIcon(R.drawable.baseline_delete_black_18)
                        .setTitle(R.string.confrim_delete).setPositiveButton(R.string.yes,
new DialogInterface.OnClickListener() {
                                @Override

                                public void onClick(DialogInterface dialog, int which) {

MembersManager.getInstance(getActivity()).deleteMember(mMember);
                                    Intent intent = new Intent();

                                    intent.putExtra(MEMBER_DELETED,true);

   getTargetFragment().onActivityResult(getTargetRequestCode(),MEMBER_DELETED_RES,intent

);                                      DeleteConformationDialog.this.dismiss();

                                }           getActivity().finish();

                                    }).setNegativeButton(R.string.no,null).create();
    }
}
```

SCREEN SHOTS

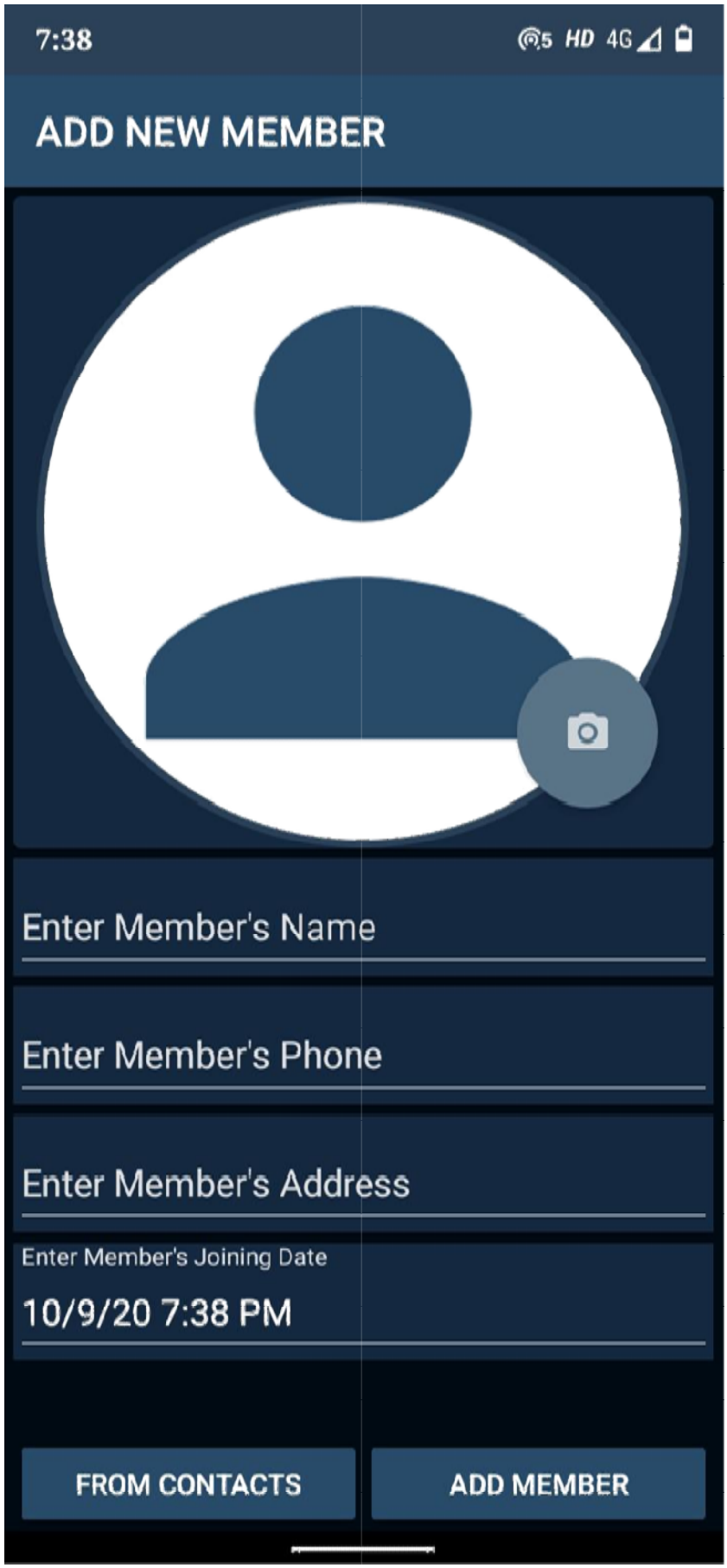

7:37
HD 4G
GYM DATA MANAGER
ALL MEMBERS
FEE RECOR
A K CHAUDHRY
-81
ABHINAV
-130
AKSHAY
-110
GOLDY
-52
GOLDY
-79
SUNNY BHYA
-77

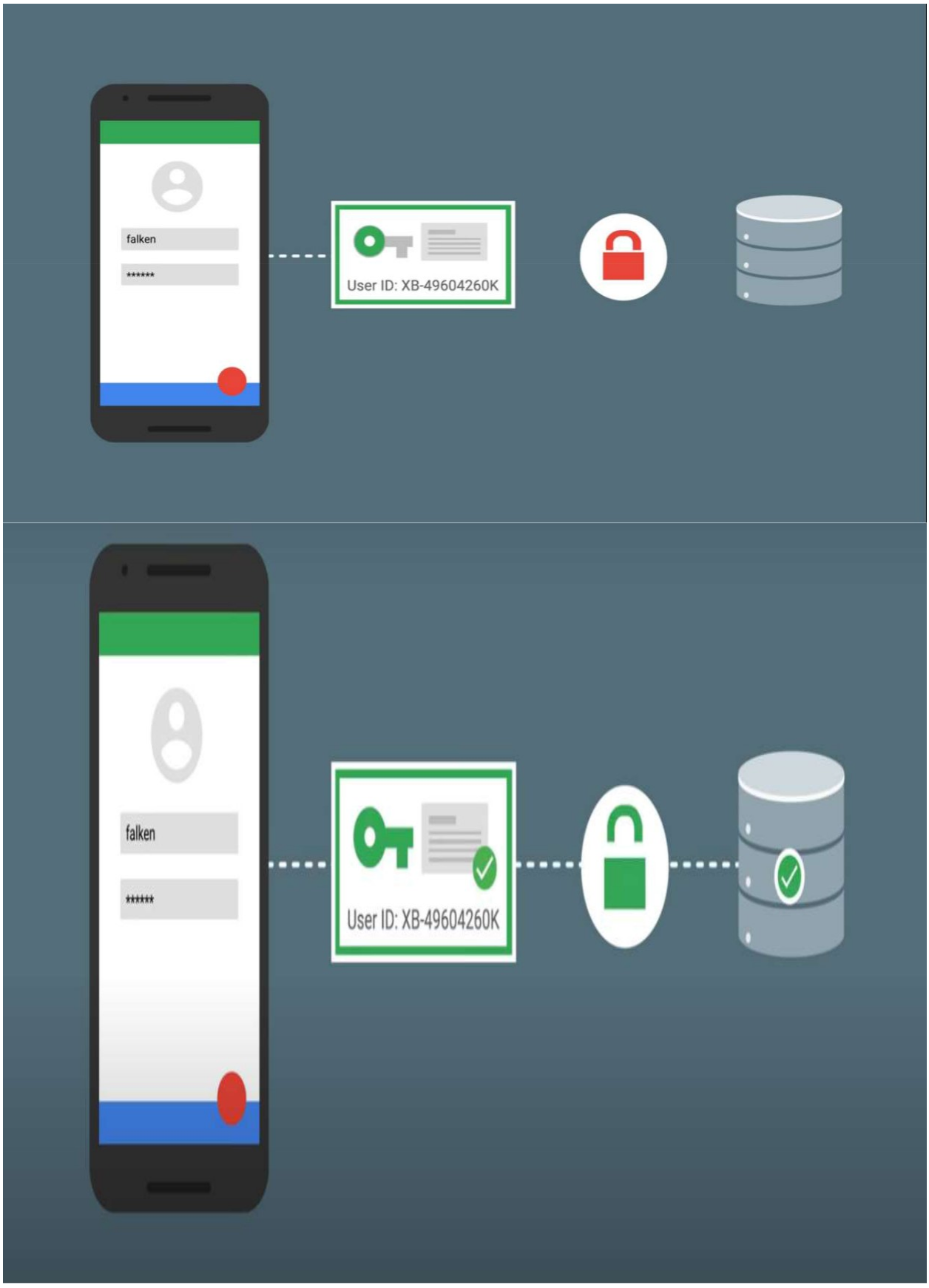

7:37
HD 4G
GYM DATA MANAGER
EE RECORDS
ANALYTICS
total fees submitted 2500
total members 3
members joined 3
members left 0
2,500
2,000
1,500
1,000
500
0
1,500
500
0
0
0
0
0
0
0
Description Label
jan
feb
mar
apr
may
jun
jul
2020

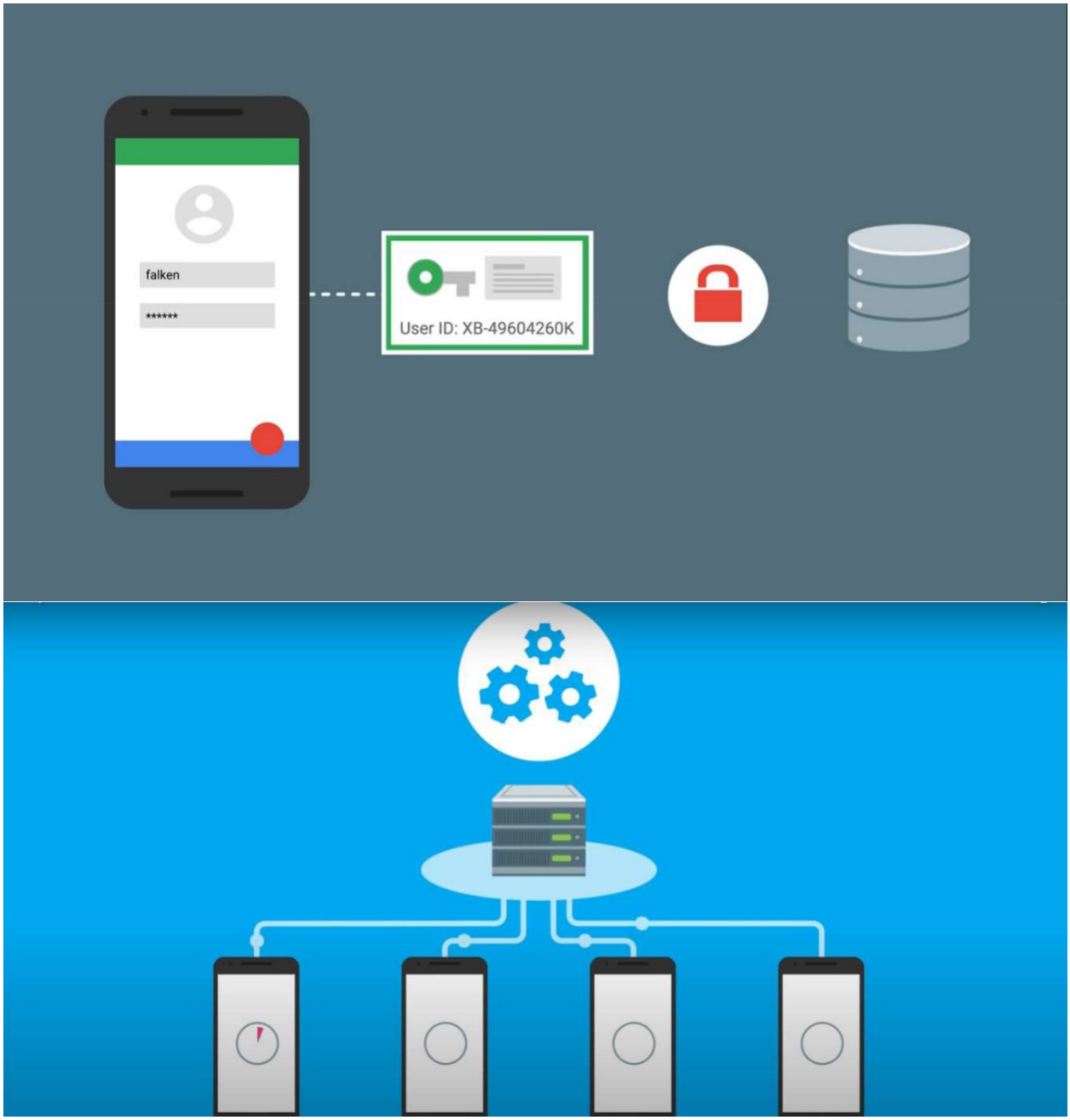

7:37
HD 4G
GENERAL SETTINGS
SET FEES
tap to set the default monthly fees
SELECT THEME
Tap to set a different application theme
CHANGE LANGUAGE
Tap to change the language


7:38
HD 4G
MEMBER DETAILS
Member's Name
ABHINAV
Member's Phone
7409783874
Member's Address
Reta nagla
Member's Joining Date
1-6-2020
Fee Records
Nothing To Show


7:38
HD 4G
GYM DATA MANAGER
MEMBERS
FEE RECORDS
ANALYTICS
AKSHAY
7/19/20 8:05 AM
500
A K CHAUDHRY
7/18/20 5:35 PM
500
GOLDY
31-May-2020
500
A K CHAUDHRY
21/05/20 4:34 PM
500
GOLDY
21/05/20 4:35 PM
1000
GOLDY
20/05/20 2:51 PM
500


7:38
HD 4G
ADD NEW MEMBER
Enter Member's Name
Enter Member's Phone
Enter Member's Address
Enter Member's Joining Date
10/9/20 7:38 PM
FROM CONTACTS
ADD MEMBER


falken
******
User ID: XB-49604260K
falken
******
User ID: XB-49604260K


falken
******
User ID: XB-49604260K

Ch 4 . CONCLUSION

The “Software Solution for Small scale Furniture Industry” is successfully designed and developed to fulfilling the necessary requirements, as identified in the requirements analysis phase, such as the system is very much user friendly, form level validation and field level validation are performing very efficiently.

The new computerized system was found to be much faster and reliable and user friendly then the existing system, the system has been designed and developed step by step and tested successfully. It eliminates the human error that are likely to creep in the kind of working in which a bulk quantity of data and calculations as to be processed.

The system results in quick retrieval of information that is very vital for the progress any organization. Cost is minimized in case of stationary. Burden of manual work is reduced as whenever transaction takes place, there is a no need to record it in many places manually.

Ch 5. FUTURE ENHANCEMENT

The software has been developed in such a way that it can accept modifications and further changes. The software is very user friendly and future any changes can be done easily.

Software restructuring is carried out. Software restructuring modifies source code in an effort to make it amenable to future changes. In general, restructuring does not modify the overall program architecture. It tends to focus on the design details of individual modules and on local data structure defined within modules.

Every system should allow scope for further development or enhancement. The system can be adapted for any further development. The system is so flexible to allow any modification need for the further functioning of programs.

Since the objectives may be brought broad in future, the system can be easily modified accordingly, as the system has been modularized. The future expansion can be done in a concise manner in order to improve the efficiently of the system

BIBILOGRAPHY

Books that were helpful in the development of this project are as follows

1. **Head First Android**
   a. **Auther** — **Dawn Griffiths & David Griffiths**
   b. **Publication** — **O'reilly**

2. **Big Nerd Ranch**
   a. **Auther** — **Kristin Marsicano**
   b. **Publication**

3. **Busy Coders Guide**
   a. **Auther** — **Mark L. Murphy**
   b. **Publication** — **Commonsware**

4. **The Definitive Guide to Firebase**
   a. **Auther** — **Laurence Moroney**
   b. **Publication** — **Prentice hall**

Websites that were helpful in development in this project are as follows

1.StackOverflow.com
2. Medium.com