**DOCUMENTATION OF**

**SERVE INDIA**

**(ON ANDROID)**

by:

Poosarla Mansi

Potnuru Pavitra

Sakalabhaktula Gireesh

Sasanapuri Indusha

**INTRODUCTION**

The proposed system aims at providing help to an orphan, an uneducated child, old men in showing a way to their respective places which may be an orphanage, old age home. The user will capture the picture of the needy and pins the location and selects the category of needy. Once the category is selected the information is sent to orphanages, old age homes, and NGOs based on categories regarding the person. Sponsors can also sponsor by easily knowing about the organization and its actions. Also In order to get the required blood group in the blood bank nearby, the person must manually visit all the blood banks which can be resolved by this app. Food wastage can also be controlled by donating that food to orphanages. Once the help has arrived the greetings are sent to the user.

All the users, orphanages, old age homes NGOs sponsors must register with this app. Once the user clicks a picture, pins location, and selects the category the information is sent to the orphanage/old age homes. The details of orphanages and old age homes are sent to sponsors. The information of uneducated children is sent to NGOs.

The snap of remnants of food in any party is captured and the location is pinned, this information is sent to all the orphanages and old age homes nearby the person who is nearby can take the food. This app provides the contact numbers and addresses of the blood banks nearby so that the person can know whether blood is available nearby and get it.

**Purpose**

Poverty is a major societal problem for all nations, not only poverty and illiterate people. Consequently, in order to overcome this problem, new methods should be established.

It is very helpful to needy people like orphans, old people, uneducated people, etc…

**Document Conventions**

To prepare this SRS we had used the lettering style of Times New Roman and Font size for Main headings is 16, subheading 14 with bold. The matter which is mentioned in this is 12 with a lettering style Times New Roman. Then headings are of lettering style Times New Roman with font size 20.

**Intended Audience and Reading Suggestions**

This SRS can be read by all developers. The rest of the part of the SRS mentions the benefits of our project, how to use the project, how the project was developed, what are the major things taken into consideration.

**Project Scope**

We are proposing people don’t go to orphanages, old age homes or any other to help needy people. We are developing an Android application where people can capture the picture of the needy and set the location through the application.

**PROBLEM DEFINITION**

**Existing System**

There are no existing specific software applications which deal with all the problems

through one application

**Proposed System**

We are proposing people don't go to separate applications to different problems. people can capture the pictures of all possible problems and post in our android application

**OVERALL DESCRIPTION**

**Product Perspective:**

The project **SERVE INDIA** is an independent application. **SERVE INDIA** is in the domain of JAVA, XML, Firebase Database.

**SYSTEM FEATURES**

**Modules**

**Number of modules:**

There are 6 modules in this application namely Donate, Orphanage, Old age Home, Blood Bank, Sponsor, NGO.

**DONATE:**

This module helps the Donor to register and also include login details. This module helps him to upload the details of the items he wanted to donate, quantity, item name, and contact address.

**ORPHANAGE:**

This module helps the Orphanages to register and also include login details. This module helps Orphanages to request for help by uploading the details of what they wanted and receiving the things from donors and sponsors.

**OLD AGE HOME:**

This module helps the Old age home to register and also include login details. This module helps Old age homes to request for help by uploading the details of what they wanted and receiving the things from donors and sponsors.

**NGO:**

This module helps the NGOs to register and also include login details. This module helps NGOs to request for help by uploading the details of what they wanted and receiving the things from donors and sponsors.

**BLOOD BANK:**

This module helps the Blood Banks to register and also include login details. This module helps users to donate blood or to view the blood donors list.

**SPONSOR:**

This module helps the Sponsor to register and also include login details. This module helps him to upload the details of the money that he wanted to Sponsor to the NGOs, Old Age Homes, Orphanages.

**Specific Requirements**

**EXTERNAL INTERFACE REQUIREMENTS:**

**User Interface:**

The user interface implemented to simulate this system is user-friendly Java, XML.

**Hardware Interface:**

RAM : 8GB

Hard Disk : 1TB

Processor : i5

**Software Interface:**

Operating System : Windows 10

Database : FireBase

Java : jdk 1.7.0

Web Server : Mobile

**Development End:**

Technologies : Android Studio

## **FEASIBILITY STUDY:**

**Technical Feasibility:**

As we are developing this Application on the Android studio platform which is open source and free of cost. Once we started developing this application in the android studio platform then there is no need of purchasing any special software or application software for support.

**Operational Feasibility:**

To determine the operational feasibility of the system we should take into consideration the awareness level of the users. Users who are using this Application don’t require much knowledge of how to use it. Everything will be understood by the user once he sees the application.

**Economic Feasibility:**

To decide whether a project is economically feasible, or not we have to consider various factors as:

* Long term returns : Database will not get overridden.
* Maintenance costs : Low maintenance cost.

## **SYSTEM DESIGN**

SOFTWARE DESIGN : JAVA

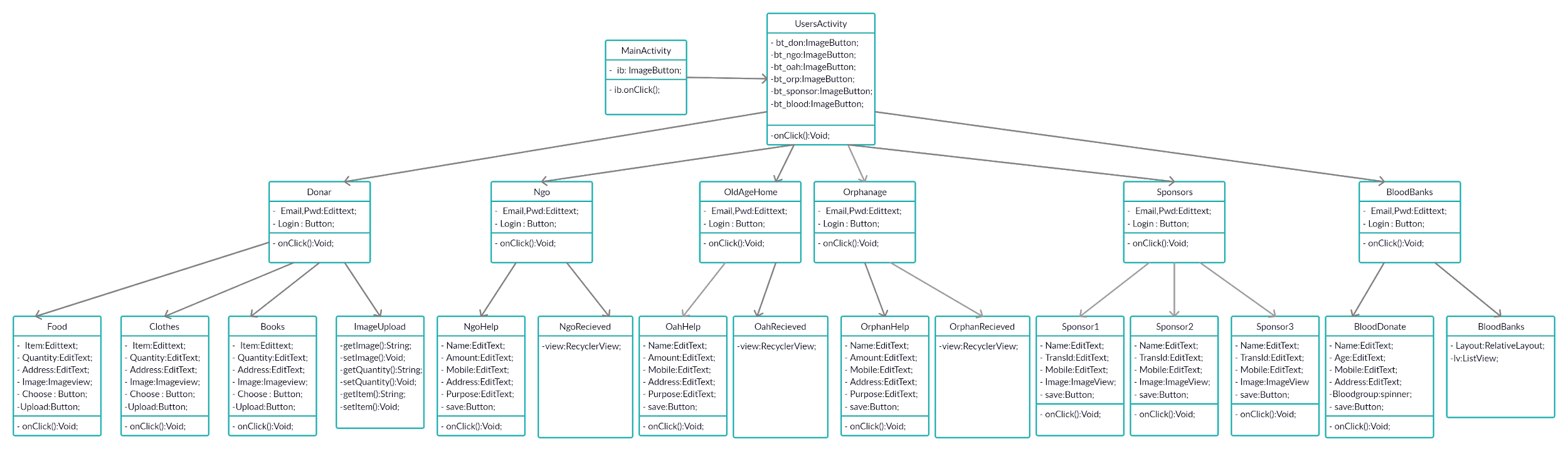
**INPUT/OUTPUT DESIGNS**

## Input design : XML

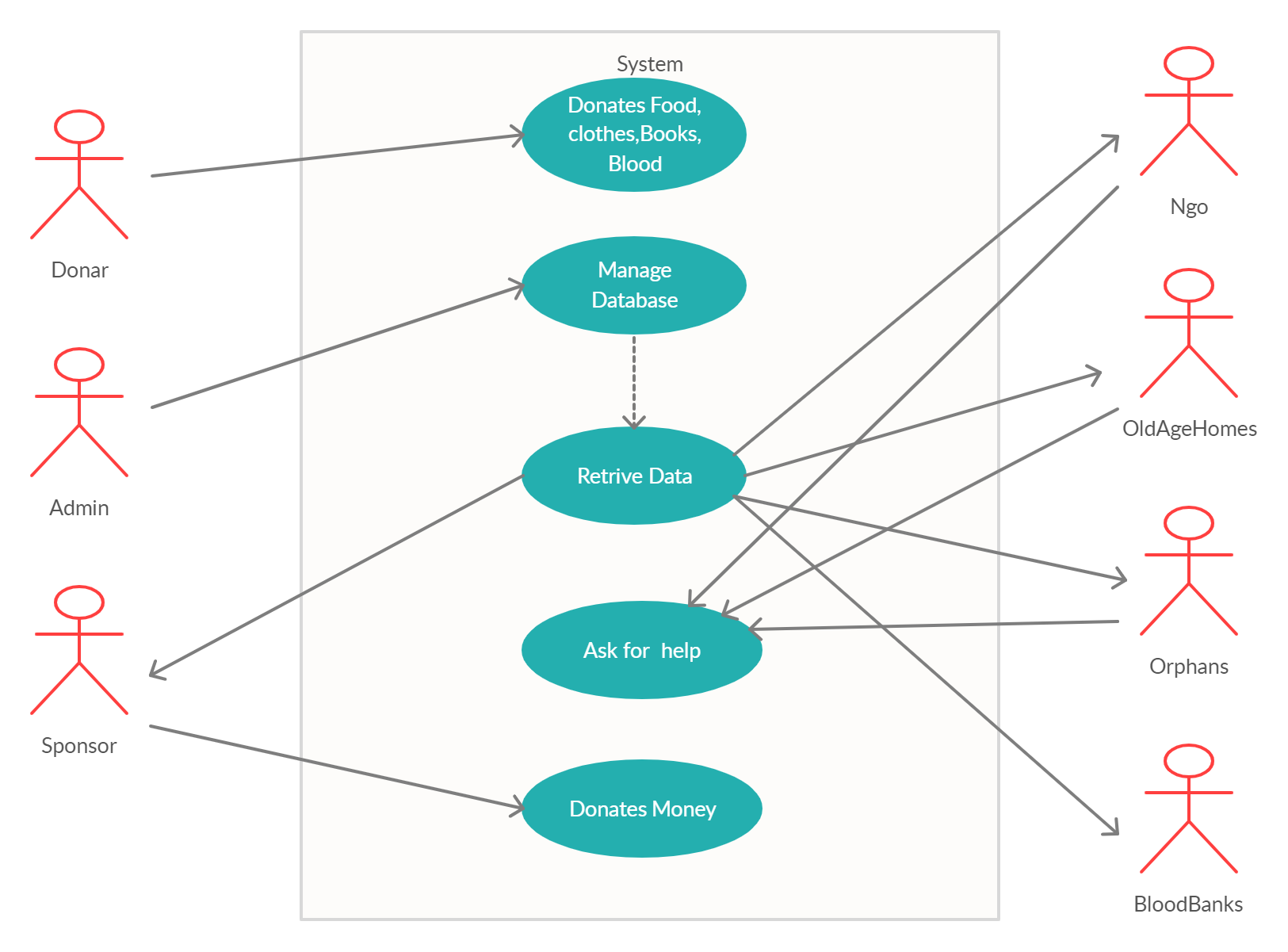
## Output design : XML

# **UML DIAGRAMS**

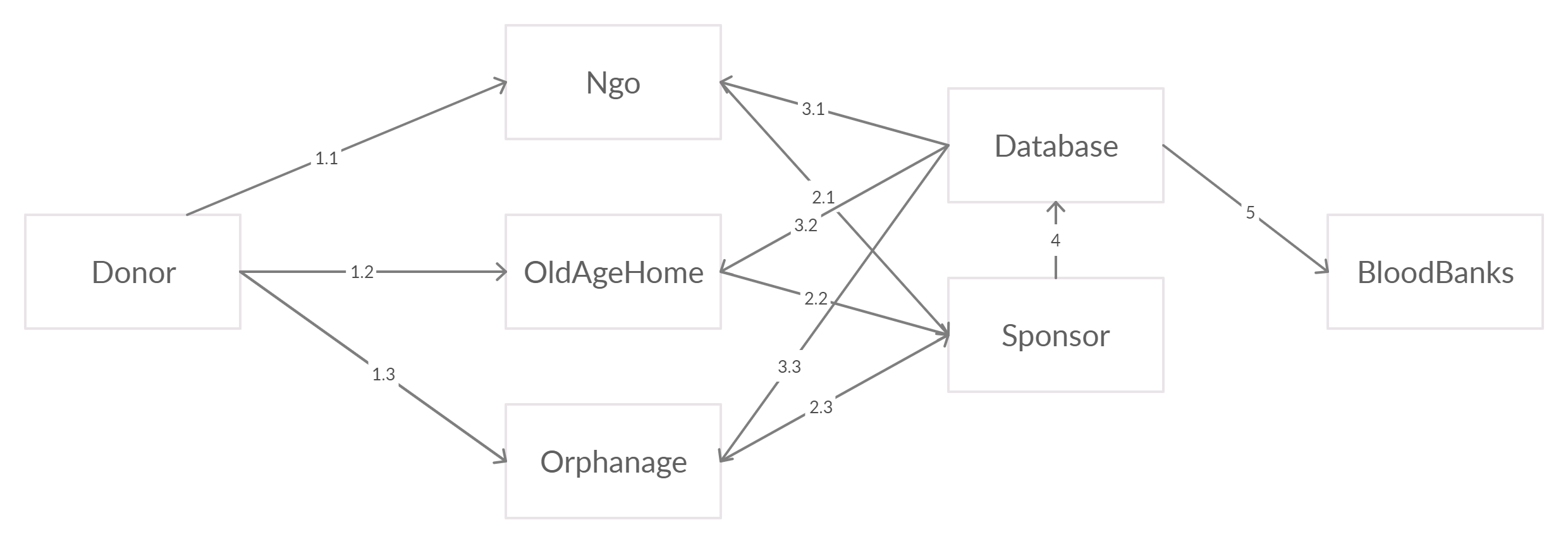
**i.ClassDiagram:-**

****

**ii.UseCaseDiagram:-**

****

**iii.Collaboration Diagram:-**

****

1. 1-Donates Food, Books, Clothes, Blood

1. 2-Donates Food, Books, Clothes, Blood

1. 3-Donates Food, Books, Clothes, Blood

2. 1-Ask for help

2. 2-Ask for help

2. 3-Ask for help

3. 1-Get Data

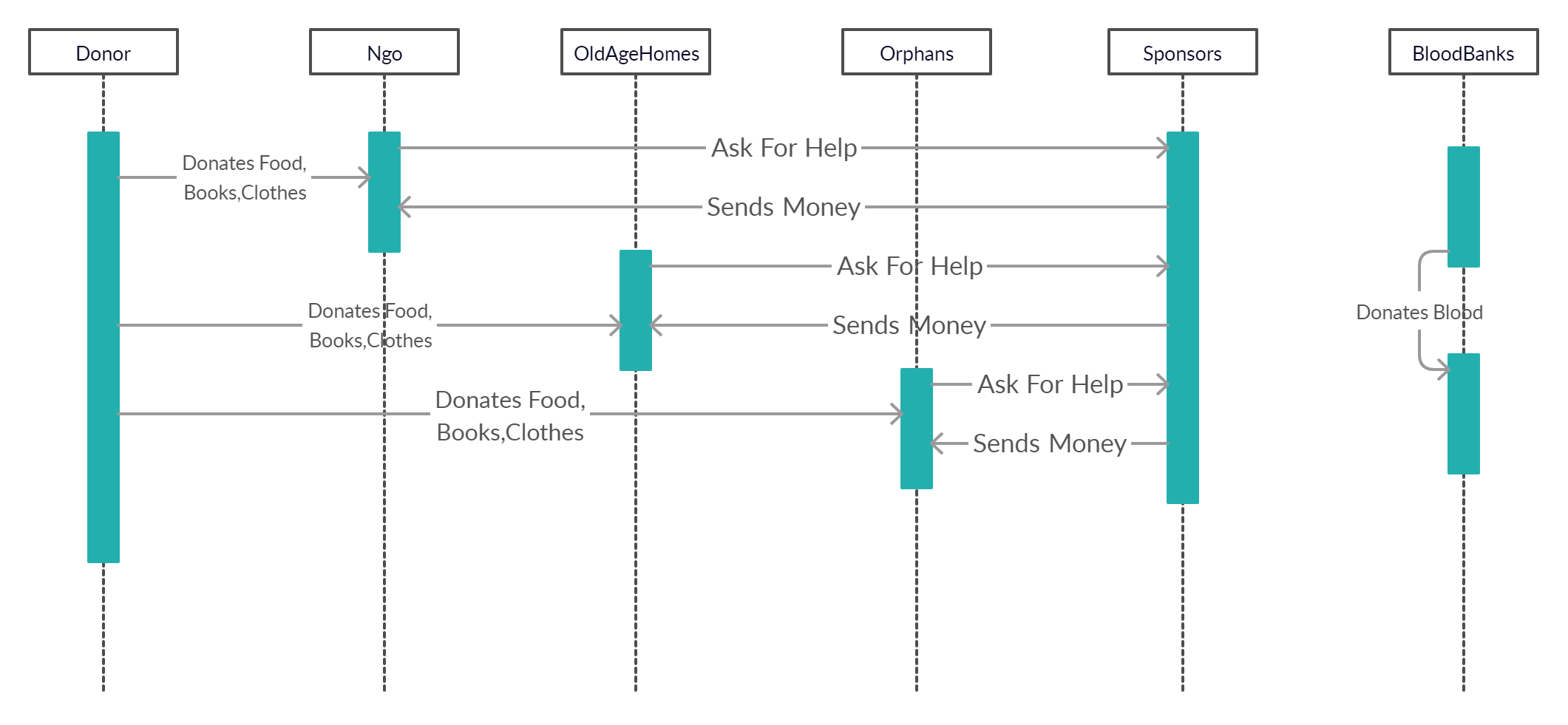
3. 2-Get Data

3. 3-Get Data

4. Sends Donation Details

5. Get Blood Donors Details

**iv. Sequence Diagram:-**

****

## **OVERVIEW OF SOFTWARE DEVELOPMENT TOOLS**

**Software**

The life-cycle paradigm demands a systematic, sequential approach to software development that begins at the system level and progresses through requirements analysis, design, coding, testing, and maintenance.

**a) Analysis Phase:**

The analysis phase consists of two sub-phases: planning and requirements definition. During planning the activities that are performed are - understand the customer’s problem, developing a recommended solution. Requirements definition is concerned with identifying the basic functions of a software component in a hardware/software/people system.

**b) Design Phase:**

Design is concerned with identifying software components, specifying relationships among components, maintaining a record of design decisions. Design consists of architectural design and detailed design.

i) Architectural Design involves identifying the software components, decoupling and decomposing them into processing modules and conceptual data structures, and specifying the interconnection between the components.

ii) Detailed Design is concerned with the details of how to package the processing modules and how to implement the processing algorithms, data structures and interconnection between them

**c) Implementation Phase:**

The implementation phase of software development involves the translation of design specifications source code and debugging, documentation, and unit testing of the source code.

**d) Testing Phase:**

It involves two kinds of testing:

i) In integration testing, the individual program units or programs are integrated and tested.

ii) Acceptance Testing involves planning and execution of various types of tests in order to demonstrations that the implemented software satisfies the stated requirements

# **Technology**

## **Android**

Android is a software stack for mobile devices that includes an operating system, middleware, and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.

## **Features**

## **Android application**

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

## **Libraries**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

**System C library** - a BSD-derived implementation of the standard C system library (libs), tuned for embedded Linux-based devices

**Media Libraries** - based on Packet Video’s Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG

**Surface Manager** - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications

**LibWebCore** - a modern web browser engine which powers both the Android browser and an embeddable web view

**SGL** - the underlying 2D graphics engine

**3D libraries** - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer

**Free Type** - bitmap and vector font rendering

**SQLite** - a powerful and lightweight relational database engine available to all applications

## **Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language.

Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool.

The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management.

An Android code editor that helps you write valid XML for your Android manifest and resource files. It will even export your project into a signed APK, which can be distributed to users.

To begin developing Android applications in the Eclipse IDE with ADT, you first need to download the Eclipse IDE and then download and install the ADT plugin. To do so, follow the steps given in [Installing the ADT Plugin](http://developer.android.com/sdk/eclipse-adt.html#installing).

## **Developing in Eclipse with ADT:**

The Android Development Tools (ADT) plug-in for Eclipse adds powerful extensions to the Eclipse integrated development environment. It allows you to create and debug Android applications easier and faster. If you use Eclipse, the ADT plug-in gives you an incredible boost in developing Android applications:

It gives you access to other Android development tools from inside the Eclipse IDE. For example, ADT lets you access the many capabilities of the DDMS tool: take screenshots, manage port-forwarding, set breakpoints, and view thread and process information directly from Eclipse.

It provides a New Project Wizard, which helps you quickly create and set up all of the basic files you'll need for a new Android application. It automates and simplifies the process of building your Android application.

## **Creating an Android project**

The ADT plugin provides a New Project Wizard that you can use to quickly create a new Android project (or a project from existing code). To create a new project:

1. Select **File** > **New** > **Project**.
2. Select **Android** > **Android Project**, and click **Next**.
3. Select the contents for the project:

Enter a *Project Name*. This will be the name of the folder where your project is created.

Under Contents, select **Create new project in workspace**. Select your project workspace location.

Under Target, select an Android target to be used as the project's Build Target. The Build Target specifies which Android platform you'd like your application built against.

Unless you know that you'll be using new APIs introduced in the latest SDK, you should select a target with the lowest platform version possible, such as Android 1.1.

**Note:** You can change your Build Target for your project at any time: Right-click the project in the Package Explorer, select **Properties**, select **Android,** and then check the desired Project Target.

Under Properties, fill in all necessary fields.

Enter an *Application name*. This is the human-readable title for your application — the name that will appear on the Android device.

Enter a *Package name*. This is the package namespace (following the same rules as for packages in the Java programming language) where all your source code will reside.

Select *Create Activity* (optional, of course, but common) and enter a name for your main Activity class.

Enter a *Min SDK Version*. This is an integer that indicates the minimum API Level required to properly run your application. Entering this here automatically sets the min Sdk Version attribute in the [<uses-sdk>](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html) of your Android Manifest file. If you're unsure of the appropriate API Level to use, copy the API Level listed for the Build Target you selected in the Target tab.

4. Click **Finish**.

Once you complete the New Project Wizard, ADT creates the following folders and files in your new project:

src/

Includes your stub Activity Java file. All other Java files for your application go here.

*<Android Version>*/ (e.g., Android 1.1/)

Includes the android.jar file that your application will build against. This is determined by the build target that you have chosen in the *New Project Wizard*.

gen/

This contains the Java files generated by ADT, such as your R.java file and interfaces created from AIDL files.

assets/

This is empty. You can use it to store raw asset files. See [Resources and Assets](http://developer.android.com/guide/topics/resources/index.html).

res/

A folder for your application resources, such as drawable files, layout files, string values, etc. See [Resources and Assets](http://developer.android.com/guide/topics/resources/index.html).

AndroidManifest.xml

The Android Manifest for your project. See [The AndroidManifest.xml File](http://developer.android.com/guide/topics/manifest/manifest-intro.html).

Default Properties

This file contains project settings, such as the build target. This file is integral to the project, as such, it should be maintained in a Source Revision Control system. It should never be edited manually — to edit project properties, right-click the project folder and select "Properties".

**Running your application:**

Before you can run your application on the Android Emulator, you **must** create an Android Virtual Device (AVD). An AVD is a configuration that specifies the Android platform to be used on the emulator. You can read more in the [Android Virtual Devices](http://developer.android.com/guide/developing/tools/avd.html) document, but if you just want to get started, follow the simple guide below to create an AVD.

If you will be running your applications only on actual device hardware, you do not need an AVD — see [Developing On a Device](http://developer.android.com/guide/developing/device.html) for information on running your application.

**Creating an AVD:**

With ADT 0.9.3 and above, the Android SDK and AVD Manager provides a simple graphical interface for creating and managing AVDs. (If you're using ADT version 0.9.1 or older, you must use the android tool to create your AVDs—read the AVD guide to [Creating an AVD](http://developer.android.com/guide/developing/tools/avd.html#creating).)

To create an AVD with the AVD Manager:

Select **Window > Android SDK and AVD Manager**, or click the Android SDK and AVD Manager icon (a black device) in the Eclipse toolbar.

In the Virtual Devices panel, you'll see a list of existing AVDs. Click **New** to create a new AVD.

Fill in the details for the AVD.

Give it a name, a platform target, an SD card image (optional), and a skin (HVGA is default).

Click **Create AVD**.

Your AVD is now ready and you can close the AVD Manager. In the next section, you'll see how the AVD is used when launching your application on an emulator.

For more information about AVDs, read the [Android Virtual Devices](http://developer.android.com/guide/developing/tools/avd.html) documentation.

**Running your application:**

**Note:** Before you can run your application, be sure that you have created an AVD with a target that satisfies your application's Build Target. If an AVD cannot be found that meets the requirements of your Build Target, you will see a console error telling you so and the launch will be aborted.

To run (or debug) your application, select **Run** > **Run** (or **Run** > **Debug**) from the Eclipse main menu. The ADT plugin will automatically create a default launch configuration for the project.

When you choose to run or debug your application, Eclipse will perform the following:

Compile the project (if there have been changes since the last build).

Create a default launch configuration (if one does not already exist for the project).

Install and start the application on an emulator or device (based on the Deployment Target defined by the run configuration).

By default, Android application run configurations use an "automatic target" mode for selecting a device target. For information on how automatic target mode selects a deployment target, see [Automatic and manual target modes](http://developer.android.com/guide/developing/eclipse-adt.html#AutoAndManualTargetModes) below.

If debugging, the application will start in the "Waiting For Debugger" mode. Once the debugger is attached, Eclipse will open the Debug perspective.

To set or change the launch configuration used for your project, use the launch configuration manager. See [Creating a Launch Configuration](http://developer.android.com/guide/developing/eclipse-adt.html#launchconfig) for information.

## **Creating a run configuration:**

The run configuration specifies the project to run, the Activity to start, the emulator options to use, and so on. When you first run a project as an *Android Application*, ADT will automatically create a run configuration. The default run configuration will launch the default project Activity and use automatic target mode for device selection (with no preferred AVD). If the default settings don't suit your project, you can customize the launch configuration or even create a new.

To create or modify a launch configuration, follow these steps as appropriate for your Eclipse version:

Open the run configuration manager.

In Eclipse 3.3 (Europa), select **Run** > **Open Run Dialog** (or **Open Debug Dialog**)

In Eclipse 3.4 (Ganymede), select **Run** > **Run Configurations** (or **Debug Configurations**)

Expand the **Android Application** item and create a new configuration or open an existing one.

To create a new configuration:

Select **Android Application** and click the *New launch configuration* icon above the list (or, right-click **Android Application** and click **New**).

Enter a Name for your configuration.

In the Android tab, browse and select the project you'd like to run with the configuration.

To open an existing configuration, select the configuration name from the list nested below the **Android Application**.

Adjust your desired launch configuration settings.

In the Target tab, consider whether you'd like to use Manual or Automatic mode when selecting an AVD to run your application. See the following section on [Automatic and manual target modes](http://developer.android.com/guide/developing/eclipse-adt.html#AutoAndManualModes)).

You can specify any emulator options to the Additional Emulator Command Line Options field. For example, you could add -scale 96dpi to scale the AVD's screen to an accurate size, based on the dpi of your computer monitor. For a full list of emulator options, see the [Android Emulator](http://developer.android.com/guide/developing/tools/emulator.html) document.

## **Designing:**

**Designing for performance:**

An Android application should be fast. Well, it's probably more accurate to say that it should be *efficient*. That is, it should execute as efficiently as possible in the mobile device environment, with its limited computing power and data storage, smaller screen, and constrained battery life.

As you develop your application, keep in mind that, while the application may perform well enough in your emulator, running on your dual-core development computer, it will not perform that well when running a mobile device — even the most powerful mobile device can't match the capabilities of a typical desktop system. For that reason, you should strive to write efficient code, to ensure the best possible performance on a variety of mobile devices.

Generally speaking, writing fast or efficient code means keeping memory allocations to a minimum, writing tight code, and avoiding certain language and programming idioms that can subtly cripple performance. In object-oriented terms, most of this work takes place at the *method* level, on the order of actual lines of code, loops, and so on.

## **TESTING**

## **SOFTWARE TESTING**

Software testing is a critical element of software quality and assurance and represents the ultimate review of specifications, design, and coding. Testing is an exposure of the system to trial input to see whether it produces correct output.

## **Testing phases**

Software testing includes the following:

1. Test activities are determined and test data selected
2. The test is conducted and test results are compared with the expected results.

## **Testing activities**

1. INSPECTING COMPONENTS: This finds faults in the individual component through the manual inspection of its source code.
2. UNIT TESTING: This finds faults by isolating an individual component using test stubs and drivers and by exercising the components using a test case.
3. INTEGRATION TESTING: This finds faults by integrating several components together. System testing, which focuses on the complete system, its functional and non-functional requirements, and its target environment.

## **Unit Testing**

Unit testing focuses on the building blocks of the software system, that is, objects and subsystems. They are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Unit testing makes it easier to pinpoint and correct faults given that few computers are involved in this test. Unit testing allows parallelism in the testing activities; that is each component can be tested independently of one another.

The specific candidates for unit testing are chosen from the object model and the system decomposition of the system. In principle, all the objects developed during the development process should be tested. Which is often not feasible because of time and budget?

Many unit testing techniques have been devised. Some of them are**:**

## **Equivalence Testing**

It is a black-box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes, and a test case is selected for each class. The assumption of equivalence testing is that the system usually behaves in similar ways for all members of a class. To test the behavior associated with an equivalence class, we only need to test one member of the class. Equivalence testing consists of two steps: identification of the equivalence classes and selection of the test inputs.

The following criteria are used for the equivalence testing:

1. COVERAGE: Every possible input belongs to one of the equivalent classes.
2. DISJOINTEDNESS: No input belongs to one of the equivalent classes.
3. REPRESENTATION: If the execution demonstrates an error when a particular member of an equivalence class is used as input, then the same error can be detected by using any other member of the class as input.

## **Boundary Testing**

Boundary testing is a special case of equivalence testing and focuses on the conditions at the boundary of the equivalence classes. Rather than selecting any element in the equivalence class, boundary testing requires that the element be selected from the “edges” of the equivalence class.

A disadvantage of equivalence class and boundary testing is that these techniques do not explore a combination of test input data.

## **Path testing**

Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind the path is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.

The path testing technique was developed for imperative languages. Object-oriented language introduces several difficulties when using path testing.

1. POLYMORPHISM: Polymorphism enables messages to be bound to different methods based on the class of the target. Although this enables developers to reuse code across a large number of classes, it also introduces more cases to test.
2. SHORTER METHODS: Methods in object-oriented language have the tendency to be shorter then procedures and functions in imperative languages. This decreases the likelihood of control flow faults, which can be uncovered using the path testing technique.

## **State-based testing**

Object-oriented languages introduce the opportunity for new types of faults in object-oriented systems.

State-based testing is a recent testing technique, which focuses on object-oriented systems. Most testing technique, which focuses on selecting a number of test inputs for a given state of the system, exercising a component or a system, and comparing the observed outputs with java. State-based testing focuses on comparing the resulting state of the system with the expected state. In the context of a class, state-based testing consists of deriving test cases from the UML statechart diagram for the class.

## **Integration testing**

It detects faults that have not been detected during unit testing, by focusing on a small group of components.

## **Test Case Design**

The design of tests for software and other engineering products can be as challenging as the initial design of the product. Test case methods provide the developer with a systematic approach to testing. Moreover, these methods provide a mechanism that can help to ensure the completeness of tests and provide the highest likelihood for uncovering errors in software.

Any Engineered product can be tested in either of the two ways:

1. Knowing the specified function that a product has been designed to perform, tests can be conducted. These tests demonstrate whether each function is fully operational and at the same time searches for errors in each function.
2. Knowing the internal workings of a product, tests can be conducted to ensure that internal operations are performed according to specifications and all internal components hence been adequately exercised.

Test case design methods are divided into two types:

1. White-box testing
2. Black-box testing

## **White-box testing**

White –box testing, sometimes called glass-box testing is a test, case design method that uses the control structure of the procedural design to derive test cases. Using white-box testing methods, the s/w engineer can derive test cases that guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structures to ensure their validity.

Basis path testing is a white-box testing technique. The basis path method enables the test case designer to derive a logical complexity measure of a procedural design and use this measure as a guide for defining a basis set is guaranteed to exercise every statement in the program at least one time during testing.

## **Black-box testing**

Black-box testing, also called behavioral testing, focuses on the functional requirements of the s/w. Black-box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements of a program. It is a complementary approach that is likely to uncover a different class of errors that white-box methods could not.

Black-box testing attempted to find errors in the following categories.

* Incorrect or missing functions.
* Interface er
* Errors in data structures or external database access.
* Behavior or performance errors.
* Initialization and termination errors.

Black-box testing purposely disregards control structure; attention is focused on the information domain. By applying black-box techniques, we derive a set of cases that satisfies the criteria test cases that reduce, by a count that is greater than one, the number of additional test cases that must be designed to achieve reasonable testing. Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specified test.

**CODING**

*logindonar.xml*

*<?***xml version="1.0" encoding="utf-8"***?>*

<**androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:tools="http://schemas.android.com/tools"**

**android:layout\_width="match\_parent"**

**android:background="@drawable/back"**

**android:layout\_height="match\_parent"**

**tools:context=".Login\_donar"**>

<**ImageView**

**android:id="@+id/imageView"**

**android:layout\_width="134dp"**

**android:layout\_height="131dp"**

**android:layout\_marginStart="126dp"**

**android:layout\_marginLeft="126dp"**

**android:layout\_marginTop="132dp"**

**android:layout\_marginEnd="126dp"**

**android:layout\_marginRight="126dp"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.28"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"**

**app:srcCompat="@drawable/help1"** />

<**EditText**

**android:id="@+id/et\_logindonor\_pwd"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="89dp"**

**android:layout\_marginLeft="89dp"**

**android:layout\_marginTop="24dp"**

**android:layout\_marginEnd="99dp"**

**android:layout\_marginRight="99dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:hint=" Enter password"**

**android:inputType="textPassword"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_logindonor\_email"** />

<**EditText**

**android:id="@+id/et\_logindonor\_email"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="89dp"**

**android:layout\_marginLeft="89dp"**

**android:layout\_marginTop="65dp"**

**android:layout\_marginEnd="108dp"**

**android:layout\_marginRight="108dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:hint=" Enter email"**

**android:inputType="textPersonName"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/imageView"** />

<**TextView**

**android:id="@+id/textView5"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="161dp"**

**android:layout\_marginLeft="161dp"**

**android:layout\_marginTop="280dp"**

**android:layout\_marginEnd="180dp"**

**android:layout\_marginRight="180dp"**

**android:fontFamily="@font/hel"**

**android:text="Log In"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="23dp"**

**android:textStyle="bold"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"** />

<**Button**

**android:id="@+id/bt\_logindonor\_login"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="56dp"**

**android:layout\_marginLeft="56dp"**

**android:layout\_marginTop="45dp"**

**android:background="@drawable/bg2"**

**android:fontFamily="@font/hel"**

**android:text="Login"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="15dp"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_logindonor\_pwd"** />

<**Button**

**android:id="@+id/bt\_logindonor\_register"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginTop="45dp"**

**android:layout\_marginEnd="56dp"**

**android:layout\_marginRight="56dp"**

**android:background="@drawable/bg2"**

**android:fontFamily="@font/hel"**

**android:text="signup"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="15dp"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_logindonor\_pwd"** />

</**androidx.constraintlayout.widget.ConstraintLayout**>

LoginDonar.java:-

**package** com.example.serveindia;

**import** androidx.annotation.NonNull;

**import** androidx.appcompat.app.AppCompatActivity;

**import** android.content.Intent;

**import** android.os.Bundle;

**import** android.view.View;

**import** android.widget.Button;

**import** android.widget.EditText;

**import** android.widget.Toast;

**import** com.google.android.gms.tasks.OnFailureListener;

**import** com.google.android.gms.tasks.OnSuccessListener;

**import** com.google.firebase.auth.AuthResult;

**import** com.google.firebase.auth.FirebaseAuth;

**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;

**public class** Login\_donar **extends** AppCompatActivity {

EditText **donar\_pwd**;

EditText **donar\_email**;

Button **login**;

Button **reg**;

FirebaseAuth **at**;

**private** DatabaseReference **db**;

@Override

**protected void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_login\_donar***);

**donar\_email** = findViewById(R.id.***et\_logindonor\_email***);

**donar\_pwd** = findViewById(R.id.***et\_logindonor\_pwd***);

**login** = findViewById(R.id.***bt\_logindonor\_login***);

**reg** = findViewById(R.id.***bt\_logindonor\_register***);

**at**=FirebaseAuth.*getInstance*();

**db**= FirebaseDatabase.*getInstance*().getReference().child(**"Users"**).child(**"Donors"**);

**reg**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View view) {

Intent it = **new** Intent(Login\_donar.**this**, Reg\_donar.**class**);

startActivity(it);

}

});

**login**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View view) {

*// Intent it = new Intent(Login\_donar.this, Foodactivity.class);*

*//startActivity(it);*

String email=**donar\_email**.getText().toString();

String pwd=**donar\_pwd**.getText().toString();

Toast.*makeText*(getApplicationContext(),**"cool"**,Toast.***LENGTH\_LONG***).show();

**if**(validate(email,**donar\_email**)validate(pwd,**donar\_email**))

{

Toast.*makeText*(getApplicationContext(),**"Loging In"**,Toast.***LENGTH\_SHORT***).show();

loginDon(email,pwd);

}

**donar\_email**.setText(**""**);

**donar\_pwd**.setText(**""**);

}

});

}

**private void** loginDon(String email,String password){

**at**.signInWithEmailAndPassword(email,password).addOnSuccessListener(**this**, **new** OnSuccessListener<AuthResult>() {

@Override

**public void** onSuccess(AuthResult authResult) {

**db**.addValueEventListener(**new** ValueEventListener() {

@Override

**public void** onDataChange(@NonNull DataSnapshot dataSnapshot) {

**for**(DataSnapshot data: dataSnapshot.getChildren())

{

**if**(data.getKey().equals(FirebaseAuth.*getInstance*().getUid()))

{

startActivity(**new** Intent(Login\_donar.**this**, Selectionactivity.**class**));

**break**;

}

**else**

{

Toast.*makeText*(getApplicationContext(),**"User doesn't exist"**,Toast.***LENGTH\_SHORT***).show();

}

}

}

@Override

**public void** onCancelled(@NonNull DatabaseError databaseError) {

}

});

}

}).addOnFailureListener(**new** OnFailureListener() {

@Override

**public void** onFailure(@NonNull Exception e) {

Toast.*makeText*(getApplicationContext(),e.getMessage(),Toast.***LENGTH\_SHORT***).show();

}

});

}

**private boolean** validate(String string ,EditText editText)

{

**if**(string.isEmpty())

{

editText.setError(**"plz enter the field"**);

**return false**;

}

**return true**;

}

}

Registerdonar.xml:-

*<?***xml version="1.0" encoding="utf-8"***?>*

<**androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:tools="http://schemas.android.com/tools"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:background="@drawable/back"**

**tools:context=".Reg\_donar"**>

<**ImageView**

**android:id="@+id/imageView"**

**android:layout\_width="134dp"**

**android:layout\_height="131dp"**

**android:layout\_marginStart="126dp"**

**android:layout\_marginLeft="126dp"**

**android:layout\_marginTop="16dp"**

**android:layout\_marginEnd="126dp"**

**android:layout\_marginRight="126dp"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"**

**app:srcCompat="@drawable/help1"** />

<**TextView**

**android:id="@+id/textView14"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="21dp"**

**android:layout\_marginLeft="21dp"**

**android:layout\_marginTop="176dp"**

**android:layout\_marginEnd="35dp"**

**android:layout\_marginRight="35dp"**

**android:text="Register Here"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="23dp"**

**android:textStyle="bold"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.488"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"** />

<**Button**

**android:id="@+id/bt\_registerdonar\_cancel"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginTop="41dp"**

**android:layout\_marginEnd="38dp"**

**android:layout\_marginRight="38dp"**

**android:background="@drawable/bg2"**

**android:fontFamily="@font/hel"**

**android:text="Cancel"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="15dp"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_registerdonar\_address"** />

<**Button**

**android:id="@+id/bt\_registerdonar\_save"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="38dp"**

**android:layout\_marginLeft="38dp"**

**android:layout\_marginTop="41dp"**

**android:background="@drawable/bg2"**

**android:fontFamily="@font/hel"**

**android:text="Save"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="15dp"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_registerdonar\_address"** />

<**EditText**

**android:id="@+id/et\_registerdonar\_username"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="19dp"**

**android:ems="10"**

**android:inputType="textPersonName"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter username"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/textView14"** />

<**EditText**

**android:id="@+id/et\_registerdonar\_email"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter emailid"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**android:inputType="textEmailAddress"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_registerdonar\_username"** />

<**EditText**

**android:id="@+id/et\_registerdonar\_pwd"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter password"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**android:inputType="textPassword"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_registerdonar\_email"** />

<**EditText**

**android:id="@+id/et\_registerdonar\_phone"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter phone number"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**android:inputType="phone"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_registerdonar\_pwd"** />

<**EditText**

**android:id="@+id/et\_registerdonar\_address"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter Address"**

**android:inputType="textMultiLine"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_registerdonar\_phone"** />

</**androidx.constraintlayout.widget.ConstraintLayout**>

RegisterDonar.java:-

**package** com.example.serveindia;

**import** androidx.annotation.NonNull;

**import** androidx.appcompat.app.AppCompatActivity;

**import** android.content.Intent;

**import** android.os.Bundle;

**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.database.DatabaseReference;

**import** com.google.firebase.database.FirebaseDatabase;

**public class** Reg\_donar **extends** AppCompatActivity {

EditText **regdonar\_name**;

EditText **regdonar\_pwd**;

EditText **regdonar\_email**;

EditText **regdonar\_phone**;

EditText **regdonar\_address**;

Button **regdonar\_save**;

Button **regdonar\_cancel**;

FirebaseAuth **mAuth**;

DatabaseReference **db**;

String **username**;

String **phone**;

String **email**;

String **pwd**;

@Override

**protected void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_reg\_donar***);

**regdonar\_name**=findViewById(R.id.***et\_registerdonar\_username***);

**regdonar\_pwd**=findViewById(R.id.***et\_registerdonar\_pwd***);

**regdonar\_email**=findViewById(R.id.***et\_registerdonar\_email***);

**regdonar\_phone**=findViewById(R.id.***et\_registerdonar\_phone***);

**regdonar\_address**=findViewById(R.id.***et\_registerdonar\_address***);

**regdonar\_save**=findViewById(R.id.***bt\_registerdonar\_save***);

**regdonar\_cancel**=findViewById(R.id.***bt\_registerdonar\_cancel***);

**mAuth**= FirebaseAuth.*getInstance*();

**db**= FirebaseDatabase.*getInstance*().getReference(**"Users"**).child(**"Donars"**);

**regdonar\_save**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View view) {

**username**=**regdonar\_name**.getText().toString();

**phone**=**regdonar\_phone**.getText().toString();

**email**=**regdonar\_email**.getText().toString();

**pwd**=**regdonar\_pwd**.getText().toString();

**if**(validate(**username**,**regdonar\_name**)&&validate(**phone**,**regdonar\_phone**)&&validate(**email**,**regdonar\_email**)&&validate(**pwd**,**regdonar\_pwd**))

{

Toast.*makeText*(getApplicationContext(),**"validated"**,Toast.***LENGTH\_LONG***).show();

register(**email**,**pwd**);

}

**regdonar\_pwd**.setText(**""**);

**regdonar\_email**.setText(**""**);

**regdonar\_phone**.setText(**""**);

**regdonar\_name**.setText(**""**);

**regdonar\_address**.setText(**""**);

}

});

}

**private void** register(**final** String email, **final** String pwd) {

**mAuth**.createUserWithEmailAndPassword(email, pwd)

.addOnCompleteListener(**this**, **new** OnCompleteListener<AuthResult>() {

@Override

**public void** onComplete(@NonNull Task<AuthResult> task) {

**if** (task.isSuccessful()) {

User user=**new** User(**username**,**phone**,email);

**db**.child(**mAuth**.getUid()).setValue(user);

Toast.*makeText*(Reg\_donar.**this**,user.getUsername(),Toast.***LENGTH\_LONG***).show();

Intent it=**new** Intent(Reg\_donar.**this**,Login\_donar.**class**);

startActivity(it);

}

}

}).addOnFailureListener(**this**, **new** OnFailureListener() {

@Override

**public void** onFailure(@NonNull Exception e) {

Toast.*makeText*(Reg\_donar.**this**,e.getMessage(),Toast.***LENGTH\_LONG***).show();

}

});

}

**private boolean** validate(String string ,EditText editText)

{

**if**(string.isEmpty())

{

editText.setError(**"plz enter the field"**);

**return false**;

}

**return true**;

}

}

foodact.xml:-

*<?***xml version="1.0" encoding="utf-8"***?>*

<**androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:tools="http://schemas.android.com/tools"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:background="@drawable/back"**

**tools:context=".Foodact"**>

<**ImageView**

**android:id="@+id/imageView"**

**android:layout\_width="168dp"**

**android:layout\_height="139dp"**

**android:layout\_marginStart="122dp"**

**android:layout\_marginLeft="122dp"**

**android:layout\_marginTop="24dp"**

**android:layout\_marginEnd="122dp"**

**android:layout\_marginRight="122dp"**

**android:layout\_marginBottom="49dp"**

**app:layout\_constraintBottom\_toTopOf="@+id/spinner"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="1.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/textView"**

**app:srcCompat="@drawable/ic\_launcher\_foreground"**

**tools:ignore="VectorDrawableCompat"** />

<**Spinner**

**android:id="@+id/spinner"**

**android:layout\_width="250dp"**

**android:layout\_height="54dp"**

**android:layout\_marginStart="80dp"**

**android:layout\_marginLeft="80dp"**

**android:layout\_marginTop="48dp"**

**android:layout\_marginEnd="80dp"**

**android:layout\_marginRight="80dp"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/imageView"** />

<**EditText**

**android:id="@+id/editText"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="97dp"**

**android:layout\_marginLeft="97dp"**

**android:layout\_marginTop="30dp"**

**android:layout\_marginEnd="97dp"**

**android:layout\_marginRight="97dp"**

**android:ems="10"**

**android:inputType="number"**

**android:fontFamily="@font/times"**

**android:textColorHint="#808080"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:hint="Enter Quantity of food in kgs"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/spinner"** />

<**Button**

**android:id="@+id/choose"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="58dp"**

**android:layout\_marginLeft="58dp"**

**android:layout\_marginTop="60dp"**

**android:background="@drawable/bg2"**

**android:text="Choose"**

**android:fontFamily="@font/hel"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/editText2"** />

<**Button**

**android:id="@+id/upload"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginTop="60dp"**

**android:layout\_marginEnd="58dp"**

**android:layout\_marginRight="58dp"**

**android:background="@drawable/bg2"**

**android:fontFamily="@font/hel"**

**android:text="Upload"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/editText2"** />

<**EditText**

**android:id="@+id/editText2"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="99dp"**

**android:layout\_marginLeft="99dp"**

**android:layout\_marginTop="5dp"**

**android:layout\_marginEnd="99dp"**

**android:layout\_marginRight="99dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:hint="Enter address"**

**android:inputType="textPostalAddress"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/editText"** />

<**TextView**

**android:id="@+id/textView"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="110dp"**

**android:layout\_marginLeft="110dp"**

**android:layout\_marginTop="56dp"**

**android:layout\_marginEnd="122dp"**

**android:layout\_marginRight="122dp"**

**android:text="Donate Your Food"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="23dp"**

**android:textStyle="bold"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"** />

<**Button**

**android:id="@+id/button8"**

**android:layout\_width="40dp"**

**android:layout\_height="24dp"**

**android:layout\_marginStart="330dp"**

**android:layout\_marginLeft="330dp"**

**android:background="@drawable/exit2"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="1.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"** />

<**EditText**

**android:id="@+id/editText7"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="99dp"**

**android:layout\_marginLeft="99dp"**

**android:layout\_marginTop="5dp"**

**android:layout\_marginEnd="99dp"**

**android:layout\_marginRight="99dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:hint="Enter item"**

**android:inputType="textPostalAddress"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/editText2"** />

</**androidx.constraintlayout.widget.ConstraintLayout**>

FoodAct.java:-

**package** com.example.serveindia;

**import** androidx.annotation.NonNull;

**import** androidx.appcompat.app.AppCompatActivity;

**import** androidx.core.app.ActivityCompat;

**import** androidx.core.content.ContextCompat;

**import** android.Manifest;

**import** android.app.ProgressDialog;

**import** android.content.ContentResolver;

**import** android.content.Intent;

**import** android.content.pm.PackageManager;

**import** android.graphics.Bitmap;

**import** android.net.Uri;

**import** android.os.Bundle;

**import** android.provider.MediaStore;

**import** android.view.Menu;

**import** android.view.MenuInflater;

**import** android.view.MenuItem;

**import** android.view.View;

**import** android.webkit.MimeTypeMap;

**import** android.widget.AdapterView;

**import** android.widget.ArrayAdapter;

**import** android.widget.Button;

**import** android.widget.EditText;

**import** android.widget.ImageView;

**import** android.widget.Spinner;

**import** android.widget.TextView;

**import** android.widget.Toast;

**import** com.google.android.gms.tasks.OnFailureListener;

**import** com.google.android.gms.tasks.OnSuccessListener;

**import** com.google.firebase.auth.FirebaseAuth;

**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.google.firebase.database.annotations.Nullable;

**import** com.google.firebase.storage.FirebaseStorage;

**import** com.google.firebase.storage.OnProgressListener;

**import** com.google.firebase.storage.StorageReference;

**import** com.google.firebase.storage.UploadTask;

**import** java.io.IOException;

**import** java.util.ArrayList;

**import** java.util.Date;

**public class** Foodact **extends** AppCompatActivity {

**private** Spinner **sp**;

**private** TextView **tv**;

**private** ImageView **iv**;

**private** EditText **et**,**et2**,**et3**;

**private** Button **bt\_choose**,**bt\_upload**,**exit**;

**private** ArrayList<String> **al**;

**private** ArrayAdapter<String> **ad**;

**private** Uri **uri**;

String **item**;

**private int Image\_Request\_Code** = 7;

**private** ProgressDialog **progressDialog** ;

**private** StorageReference **sr**;

**private** DatabaseReference **dr**;

@Override

**protected void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_foodact***);

**tv**=findViewById(R.id.***textView***);

**exit**=findViewById(R.id.***button8***);

**iv**=findViewById(R.id.***imageView***);

**et**=findViewById(R.id.***editText***);

**et3**=findViewById(R.id.***editText7***);

**et2**=findViewById(R.id.***editText2***);

**bt\_choose**=findViewById(R.id.***choose***);

**bt\_upload**=findViewById(R.id.***upload***);

**sp**=findViewById(R.id.***spinner***);

**progressDialog** = **new** ProgressDialog(Foodact.**this**);

**al**= **new** ArrayList<String>();

**al**.add(**"NGO "**);

**al**.add(**"OldAgeHome"**);

**al**.add(**"Orphanage "**);

**ad**=**new** ArrayAdapter<String>(getApplicationContext(),android.R.layout.***simple\_spinner\_dropdown\_item***,**al**);

**sp**.setAdapter(**ad**);

**sp**.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {

@Override

**public void** onItemSelected(AdapterView<?> parent, View view, **int** position, **long** id) {

**if**(position==0){

**sr**= FirebaseStorage.*getInstance*().getReference().child(**"Food"**).child(**"Food\_Ngo"**);

**dr**= FirebaseDatabase.*getInstance*().getReference(**"Food"**).child(**"Food\_Ngo"**);

}

**if**(position==1){

**sr**= FirebaseStorage.*getInstance*().getReference().child(**"Food"**).child(**"Food\_OldAgeHomes"**);

**dr**= FirebaseDatabase.*getInstance*().getReference(**"Food"**).child(**"Food\_OldAgeHomes"**);

}

**if**(position==2){

**sr**= FirebaseStorage.*getInstance*().getReference().child(**"Food"**).child(**"Food\_Orphanagess"**);

**dr**= FirebaseDatabase.*getInstance*().getReference(**"Food"**).child(**"Food\_Orphanagess"**);

}

}

@Override

**public void** onNothingSelected(AdapterView<?> parent) {

}

});

**bt\_choose**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View view) {

Intent intent = **new** Intent();

intent.setType(**"Clothes/\*"**);

intent.setAction(Intent.***ACTION\_GET\_CONTENT***);

startActivityForResult(Intent.*createChooser*(intent, **"Please Select Image"**), **Image\_Request\_Code**);

}

});

**bt\_upload**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View view) {

UploadImageFileToFirebaseStorage();

thank();

}

});

**exit**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View view) {

Intent it=**new** Intent(Foodact.**this**,Thankyouact.**class**);

startActivity(it);

}

});

}

@Override

**protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {

**super**.onActivityResult(requestCode, resultCode, data);

**if** (requestCode == **Image\_Request\_Code** && resultCode == ***RESULT\_OK*** && data != **null** && data.getData() != **null**) {

**uri** = data.getData();

**try** {

Bitmap bitmap = MediaStore.Images.Media.*getBitmap*(getContentResolver(),**uri**);

**iv**.setImageBitmap(bitmap);

**bt\_choose**.setText(**"Image Selected"**);

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**public** String GetFileExtension(Uri uri) {

ContentResolver contentResolver = getContentResolver();

MimeTypeMap mimeTypeMap = MimeTypeMap.*getSingleton*();

**return** mimeTypeMap.getExtensionFromMimeType(contentResolver.getType(uri)) ;

}

**public void** UploadImageFileToFirebaseStorage() {

**if** (**uri**!= **null**) {

**progressDialog**.setTitle(**"Image is Uploading..."**);

**progressDialog**.show();

**final** StorageReference storageReference2nd = **sr**.child(System.*currentTimeMillis*() + **"."** + GetFileExtension(**uri**));

storageReference2nd.putFile(**uri**)

.addOnSuccessListener(**new** OnSuccessListener<UploadTask.TaskSnapshot>() {

@Override

**public void** onSuccess(UploadTask.TaskSnapshot taskSnapshot) {

storageReference2nd.getDownloadUrl().addOnSuccessListener(**new** OnSuccessListener<Uri>() {

@Override

**public void** onSuccess(Uri uri) {

String quantity = **et**.getText().toString().trim();

String item=**et3**.getText().toString().trim();

String address = **et2**.getText().toString().trim();

String image=uri.toString();

**progressDialog**.dismiss();

Toast.*makeText*(getApplicationContext(), **"Image Uploaded Successfully "**, Toast.***LENGTH\_SHORT***).show();

@SuppressWarnings(**"VisibleForTests"**)

ImageUploadInfo imageUploadInfo = **new** ImageUploadInfo( image,quantity,item,address);

String ImageUploadId = **dr**.push().getKey();

**dr**.child(ImageUploadId).setValue(imageUploadInfo);

}

});

}

})

.addOnFailureListener(**new** OnFailureListener() {

@Override

**public void** onFailure(@NonNull Exception exception) {

**progressDialog**.dismiss();

Toast.*makeText*(Foodact.**this**, exception.getMessage(), Toast.***LENGTH\_LONG***).show();

}

})

.addOnProgressListener(**new** OnProgressListener<UploadTask.TaskSnapshot>() {

@Override

**public void** onProgress(UploadTask.TaskSnapshot taskSnapshot) {

**progressDialog**.setTitle(**"Image is Uploading..."**);

}

});

}

**else** {

Toast.*makeText*(Foodact.**this**, **"Please Select Image or Add Image Name"**, Toast.***LENGTH\_LONG***).show();

}

}

**public void** thank(){

Intent it=**new** Intent(Foodact.**this**,Thankyouact.**class**);

startActivity(it);

}

}

sponsorhelpngo.xml:-

*<?***xml version="1.0" encoding="utf-8"***?>*

<**androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:tools="http://schemas.android.com/tools"**

**android:layout\_width="match\_parent"**

**android:background="@drawable/back"**

**android:layout\_height="match\_parent"**

**tools:context=".Sponserhelpold"**>

<**ImageView**

**android:id="@+id/imageView"**

**android:layout\_width="134dp"**

**android:layout\_height="131dp"**

**android:layout\_marginStart="126dp"**

**android:layout\_marginLeft="126dp"**

**android:layout\_marginTop="16dp"**

**android:layout\_marginEnd="126dp"**

**android:layout\_marginRight="126dp"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.0"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"**

**app:srcCompat="@drawable/help1"** />

<**TextView**

**android:id="@+id/textView14"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="21dp"**

**android:layout\_marginLeft="21dp"**

**android:layout\_marginTop="176dp"**

**android:layout\_marginEnd="35dp"**

**android:layout\_marginRight="35dp"**

**android:text="Request For Help"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="23dp"**

**android:textStyle="bold"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintHorizontal\_bias="0.488"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toTopOf="parent"** />

<**Button**

**android:id="@+id/bt\_reqo\_cancel"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginTop="41dp"**

**android:layout\_marginEnd="38dp"**

**android:layout\_marginRight="38dp"**

**android:background="@drawable/bg2"**

**android:fontFamily="@font/hel"**

**android:text="Cancel"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="15dp"**

**app:layout\_constraintEnd\_toEndOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_reqo\_address"** />

<**Button**

**android:id="@+id/bt\_reqo\_save"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="38dp"**

**android:layout\_marginLeft="38dp"**

**android:layout\_marginTop="41dp"**

**android:background="@drawable/bg2"**

**android:fontFamily="@font/hel"**

**android:text="Save"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textSize="15dp"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_reqo\_address"** />

<**EditText**

**android:id="@+id/et\_reqo\_username"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="19dp"**

**android:ems="10"**

**android:inputType="textPersonName"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter username"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/textView14"** />

<**EditText**

**android:id="@+id/et\_reqo\_amount"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter amount"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**android:inputType="phone"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_reqo\_username"** />

<**EditText**

**android:id="@+id/et\_reqo\_purpose"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter purpose"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**android:inputType="textMultiLine"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_reqo\_amount"** />

<**EditText**

**android:id="@+id/et\_reqo\_phone"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter phone number"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**android:inputType="phone"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_reqo\_purpose"** />

<**EditText**

**android:id="@+id/et\_reqo\_address"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginStart="85dp"**

**android:layout\_marginLeft="85dp"**

**android:layout\_marginTop="24dp"**

**android:ems="10"**

**android:fontFamily="@font/times"**

**android:gravity="start|top"**

**android:hint="Enter Address"**

**android:inputType="textMultiLine"**

**android:textColor="@color/common\_google\_signin\_btn\_text\_dark\_default"**

**android:textColorHint="#808080"**

**app:layout\_constraintStart\_toStartOf="parent"**

**app:layout\_constraintTop\_toBottomOf="@+id/et\_reqo\_phone"** />

</**androidx.constraintlayout.widget.ConstraintLayout**>

sponsorhelpngo.java:-

**package** com.example.serveindia;

**import** androidx.appcompat.app.AppCompatActivity;

**import** android.content.Intent;

**import** android.os.Bundle;

**import** android.view.View;

**import** android.widget.Button;

**import** android.widget.EditText;

**import** android.widget.Toast;

**import** com.google.firebase.auth.FirebaseAuth;

**import** com.google.firebase.database.DatabaseReference;

**import** com.google.firebase.database.FirebaseDatabase;

**public class** Sponserhelpold **extends** AppCompatActivity {

EditText **et1**,**et2**,**name**,**mobile**,**address**;

Button **save**,**cancel**;

**private** FirebaseAuth **mAuth**;

**private** DatabaseReference **db\_sp**;

@Override

**protected void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_sponserhelpold***);

**et1**=findViewById(R.id.***et\_reqo\_amount***);

**et2**=findViewById(R.id.***et\_reqo\_purpose***);

**name**=findViewById(R.id.***et\_reqo\_username***);

**mobile**=findViewById(R.id.***et\_reqo\_phone***);

**address**=findViewById(R.id.***et\_reqo\_address***);

**save**=findViewById(R.id.***bt\_reqo\_save***);

**cancel**=findViewById(R.id.***bt\_reqo\_cancel***);

**mAuth**=FirebaseAuth.*getInstance*();

**db\_sp**= FirebaseDatabase.*getInstance*().getReference().child(**"Request to help"**).child(**"Oldagehomes"**);

**save**.setOnClickListener(**new** View.OnClickListener() {

@Override

**public void** onClick(View view) {

String amount=**et1**.getText().toString().trim();

String purpose=**et2**.getText().toString().trim();

String username=**name**.getText().toString().trim();

String contact=**mobile**.getText().toString().trim();

String add=**address**.getText().toString().trim();

String id=**db\_sp**.push().getKey();

Help ob=**new** Help(id,username,amount,purpose,contact,add);

**db\_sp**.child(id).setValue(ob);

Toast.*makeText*(getApplicationContext(),**"Request Sent"**,Toast.***LENGTH\_SHORT***).show();

}

});

}

}

viewact.xml:-

*<?***xml version="1.0" encoding="utf-8"***?>*

<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:tools="http://schemas.android.com/tools"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:orientation="vertical"**

**android:background="@drawable/back"**

**tools:context=".Viewact"**>

<**androidx.recyclerview.widget.RecyclerView**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:id="@+id/recyclerview"**/>

</**LinearLayout**>

ViewAct.java:-

**package** com.example.serveindia;

**import** androidx.annotation.NonNull;

**import** androidx.appcompat.app.AppCompatActivity;

**import** androidx.recyclerview.widget.LinearLayoutManager;

**import** androidx.recyclerview.widget.RecyclerView;

**import** android.graphics.Bitmap;

**import** android.graphics.BitmapFactory;

**import** android.os.Bundle;

**import** android.view.MotionEvent;

**import** android.view.View;

**import** android.widget.ArrayAdapter;

**import** android.widget.Button;

**import** android.widget.EditText;

**import** android.widget.ListView;

**import** android.widget.Toast;

**import** com.firebase.ui.database.FirebaseRecyclerAdapter;

**import** com.google.android.gms.tasks.OnSuccessListener;

**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** java.lang.reflect.Member;

**import** java.util.ArrayList;

**public class** Viewact **extends** AppCompatActivity {

RecyclerView **mrecyclerview**;

FirebaseDatabase **mfdbb**;

DatabaseReference **mdr**;

@Override

**protected void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_viewact***);

**mrecyclerview** = findViewById(R.id.***recyclerview***);

**mrecyclerview**.setHasFixedSize(**true**);

**mrecyclerview**.setLayoutManager(**new** LinearLayoutManager(**this**));

**mfbdb** = FirebaseDatabase.*getInstance*();

**mdr** = **mfbdb**.getReference().child(**"Food"**).child(**"Food\_Orphanagess"**);

}

@Override

**protected void** onStart() {

**super**.onStart();

FirebaseRecyclerAdapter<ImageUploadInfo,Viewholder>firebaseRecyclerAdapter=**new**

FirebaseRecyclerAdapter<ImageUploadInfo, Viewholder>(

ImageUploadInfo.**class**,

R.layout.***image***,

Viewholder.**class**,

**mdr**

){

@Override

**protected void** populateViewHolder(Viewholder viewholder, ImageUploadInfo imageUploadInfo, **int** i) {

viewholder.setdetails(getApplicationContext(),imageUploadInfo.getImageURL(),imageUploadInfo.getAddress(),imageUploadInfo.getQuantity(),imageUploadInfo.getItem());

*//Toast.makeText(getApplicationContext(),"asss",Toast.LENGTH\_LONG).show();*

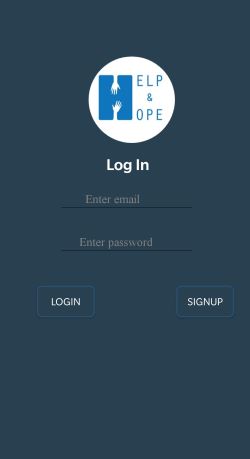
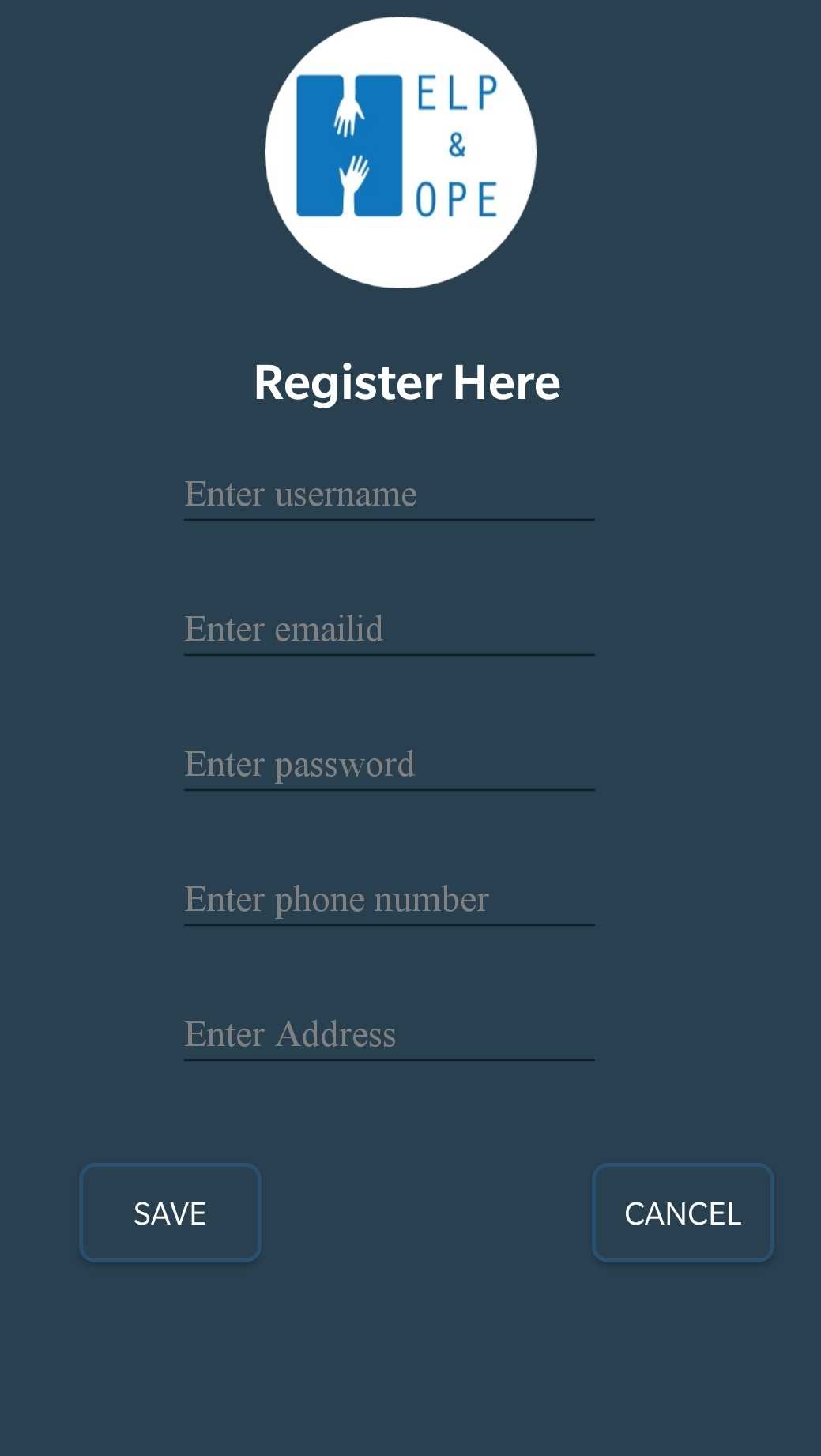
}

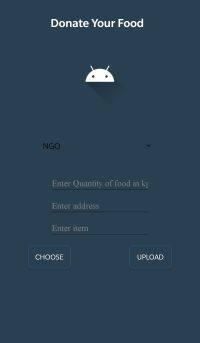
};

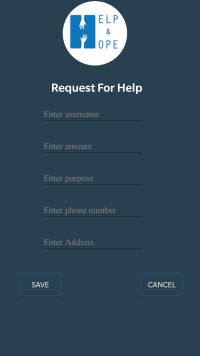
**mrecyclerview**.setAdapter(firebaseRecyclerAdapter);

}

}





**References:**

[**www.developers.android.com**](http://www.developers.android.com)

[**www.eclipse.org**](http://www.eclipse.org)