GitHub Username: mvnshrikanth

# The Blood Donor

## Description

The blood donor gives the ability to the user who want to donate blood and are in desperate need of blood. It finds nearby blood donor and gives you a channel to talk to them, finds you nearby hospitals for you. Sends you reminder notifications for regular donations. And most importantly gives you the power to save people.

## Intended User

Will be used by everyone who donates and is in desperate need of blood.

## Features

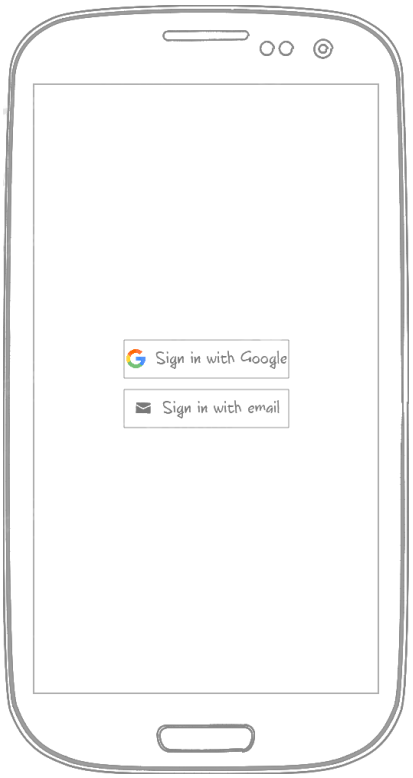
List of the main features of the app:

- Convenient and easy way to find donors.
- Notifies you if the users blood type is needed in the area.
- Helps find nearby hospitals.
- Gives a channel to talk to the donor.

## User Interface Mocks

### Screen 1

This will be the welcome screen for first time logging into the app. This is provided by the ***firebase authentication UI***, if the user ID was not found it will prompt the user to create a new account.



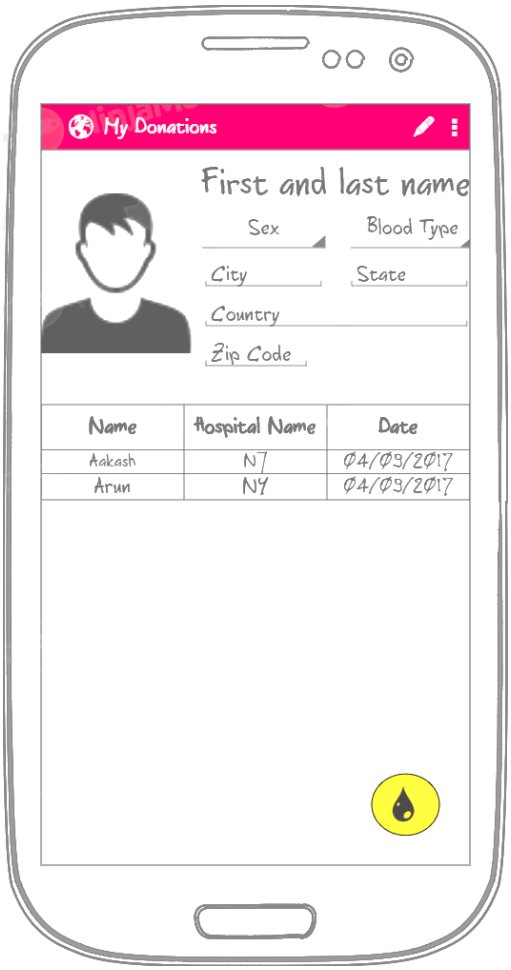
## Screen 2

This screen will be prompted if the user is signing in for the first time on the device to ***confirm his personal details***.

## Screen 3

This screen will ***show all the past donation*** of the user. Also, will allow to do the below:

1) Sign out of the app using the settings menu.
2) Edit his profile
3) Add a new donation or find a donor with floating action button



## Screen 4

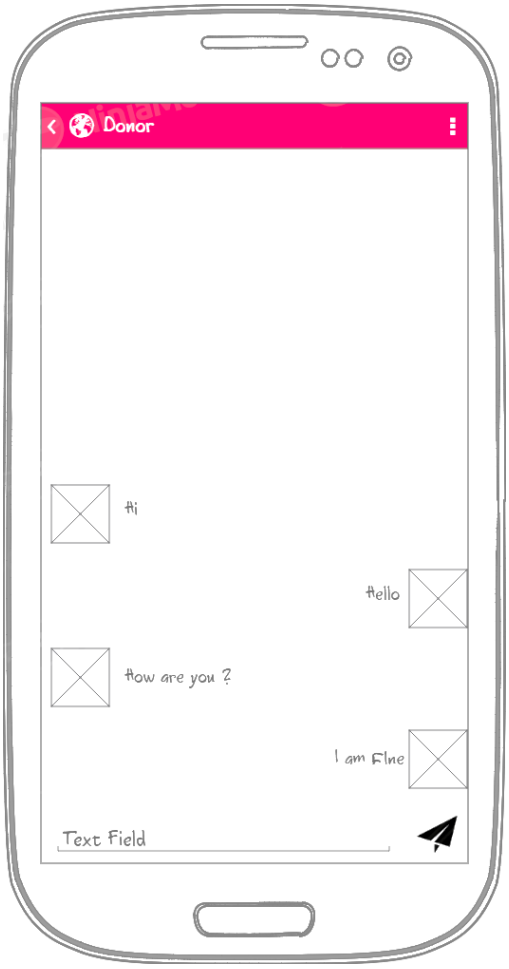
The bellow screen is when a user clicks on floating button he can either ***choose to donate or find a donor***.

## Screen 5

This screen the user can ***find donors or can find people who need donation*** based on the selection from previous screen. Taping on the person name will ***start a chat with the user***.

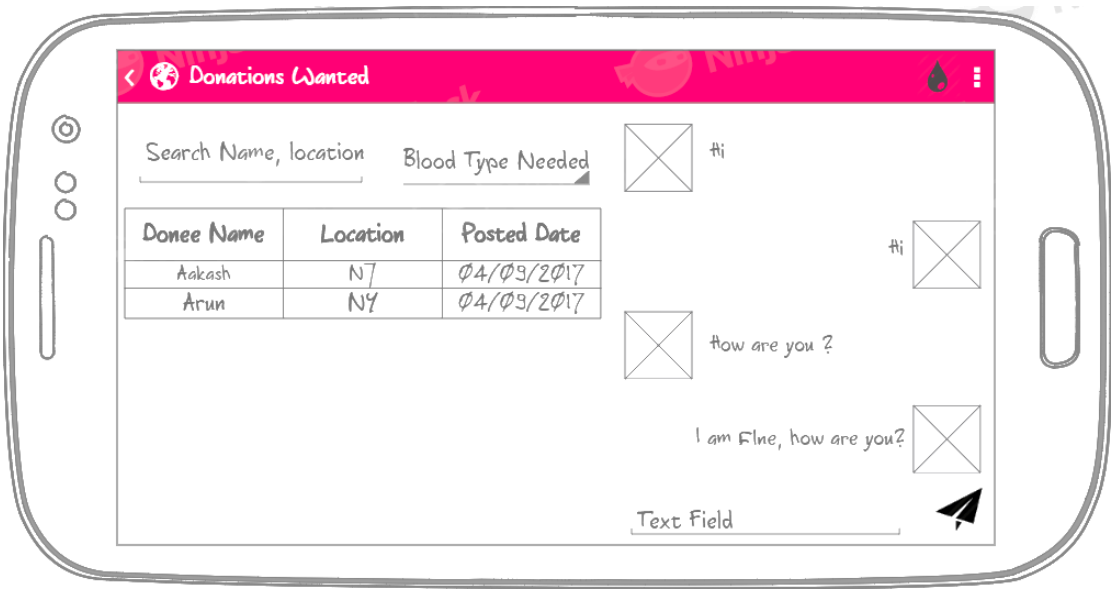
## Screen 6

This is the ***chat window*** for the user to talk to a donor or a donee.



## Screen 7

This is the ***landscape or tablet mode screen*** when the user chooses to chat with another user.




## Screen 8

The below mock is for a ***widget*** which will show list of the users who require blood currently.

## Key Considerations

### How will your app handle data persistence?

The app will be using Firebase Realtime database, to store the user information and firebase will also help provide the cached information to the application when the app is offline and will update the apps as soon as its online.

### Describe any edge or corner cases in the UX.

1. **Orientation change while API request is being made**: App might crash in such circumstances.
2. **User rapidly moves in b/w app screens**: App might lag or slow down.
3. **User closes the application while a API request was taking place**: App will crash in such circumstances.
4. **Loading a large image in the app**: App might crash in low end devices due to out of memory issue.

### Describe any libraries you'll be using and share your reasoning for including them.

The app will make use of the below listed libraries:

1) Butterknife

```
compile 'com.jakewharton:butterknife:8.8.1'
```

App will use the butterknife library so we can reduce the amount of boilerplate code needs to be written to find the views and eliminate the resource lookups by using the annotations.

2) Glide

```
compile 'com.github.bumptech.glide:glide:4.2.0'
```

App will be using the glide library to fetch the user profile pictures and show the images on the chat windows, reducing the boilerplate code for networking.

3) Design Support Library

```
compile 'com.android.support:design:26.1.0'
```

App will use the material design using the design library.

### Describe how you will implement Google Play Services or other external services.

The app will be having google, firebase services.

## Next Steps: Required Tasks

### Task 1: Project Setup

- Add rules to your root-level `build.gradle`

```
buildscript {
    // ...
    dependencies {
        // ...
        classpath 'com.google.gms:google-services:3.1.1' // google-services plugin
    }
}

allprojects {
    // ...
    repositories {
        // ...
        maven {
            url "https://maven.google.com" // Google's Maven repository
        }
    }
}
```

- Configure libraries by adding to the app/build.gradle dependencies.

```
dependencies {
    ...
    compile 'com.jakewharton:butterknife:8.5.1'
    annotationProcessor 'com.jakewharton:butterknife-compiler:8.5.1'

    //Material Design
    compile 'com.android.support:design:26.1.0'

    // Displaying images
    compile 'com.github.bumptech.glide:glide:4.2.0'

    // Firebase
    compile 'com.google.firebase:firebase-database:9.6.0'
    compile 'com.google.firebase:firebase-auth:9.6.0'
    compile 'com.google.firebase:firebase-storage:9.6.0'
    compile 'com.google.firebase:firebase-messaging:9.6.0'

    // FirebaseUI
    compile 'com.firebaseui:firebase-ui-auth:0.6.0'
}

apply plugin: 'com.google.gms.google-services'
```

## Task 2: Implement UI for Each Activity and Fragment

- Build UI for MyDonationActivity – Contains the list of dates when the user donated blood with dates and location or the hospital names.
- Build UI for DonarDoneeActivity – Contains the ListFragment, ChatFragment in case of tablet or landscape mode.
- Build UI for ProfileActivity -- Screen for user to edit his profile details, like picture, blood type and location.
- Build UI for ListFragment -- Contains the list of all the donors and their type and location.
- Build UI for ChatFragment – Contains a chat window for the user to communicate to each other.

## Task 3: Create wrapper and adapter classes

Create the wrapper classes and adapter to pull the information from firebase and show the donor list and my donation data.

- DonorMessage
- MessageAdapter
- MyDonation
- DonarDoneeAdapter

## Task 5: Create the widget and its services

1) Create the widget layout for the app.
2) Create the widget service to pull the data from firebase job dispatcher for the latest blood donations needed.
3) Create the widget provider xml to define the widget metadata.

## Task 6: Write UI tests

Create UI tests to automatically test the application UI.

## Task 7: Prepare app for release

Fix out some of the minor bugs remaining and prepare the play store listing. Steps to follow before publishing to play store:

1. Create app icon.
2. Create splash screen for app.