

## How to Use this Template

1. Make a copy [ File → Make a copy... ]
2. Rename this file: “**Capstone\_Stage1**”
3. Replace the text in green

## Submission Instructions

1. After you’ve completed all the sections, download this document as a PDF [ File → Download as PDF ]
2. Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
3. Add this document to your repo. Make sure it’s named “**Capstone\_Stage1.pdf**”

---

[Description](#)

[Intended User](#)

[Features](#)

[User Interface Mocks](#)

[Screen 1](#)

[Screen 2](#)

[Key Considerations](#)

[How will your app handle data persistence?](#)

[Describe any corner cases in the UX.](#)

[Describe any libraries you’ll be using and share your reasoning for including them.](#)

[Describe how you will implement Google Play Services.](#)

[Next Steps: Required Tasks](#)

[Task 1: Project Setup](#)

[Task 2: Implement UI for Each Activity and Fragment](#)

[Task 3: Your Next Task](#)

[Task 4: Your Next Task](#)

[Task 5: Your Next Task](#)

**GitHub Username:** [ajitesh-tiwari](#)

# Hablar

## Description

Hablar is a domain oriented messenger.

## Problem:

Each organization has large number of people, who need to interact with each other and stay in touch with the updates within the organization. For example - College has many students who

need to stay in touch with each other. Each student also has an unique college email which can be used in this case.

### **Proposed Solution:**

Design an app which lets users to send updates to other colleagues having an email address with same domain name. We can create a common feed page on which all colleagues can post and watch for updates.

Only requirement for such an application is that the people watching the feed page of a particular organization must have an email belonging to the organization.

## **Intended User**

People working at any organization having an official mail address.

## **Features**

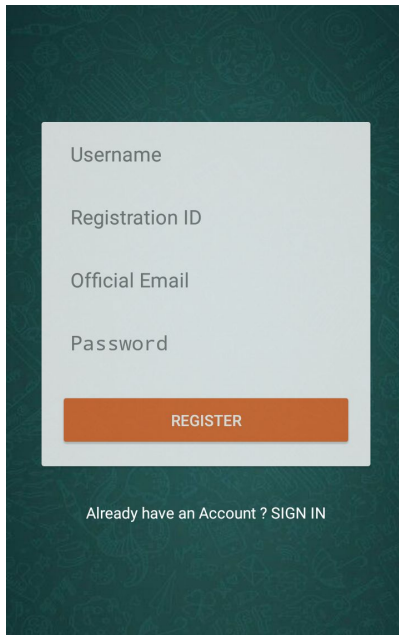
List the main features of your app. For example:

- Post Updates (Text, Images, etc.).
- Stay in touch with other colleagues.
- Get notification of updates in your organization.

## **User Interface Mocks**

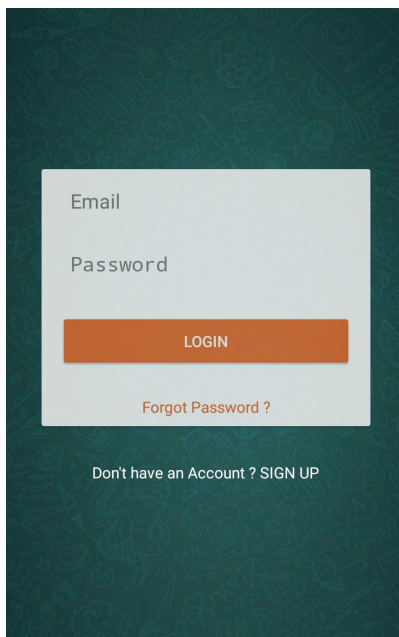
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

## Screen 1

A screenshot of the 'Sign Up' page. It features a dark green background with a subtle pattern. In the center is a light gray rectangular box containing four text input fields labeled 'Username', 'Registration ID', 'Official Email', and 'Password'. Below these fields is an orange button with the text 'REGISTER'. At the bottom of the gray box, there is a link that says 'Already have an Account ? SIGN IN'.

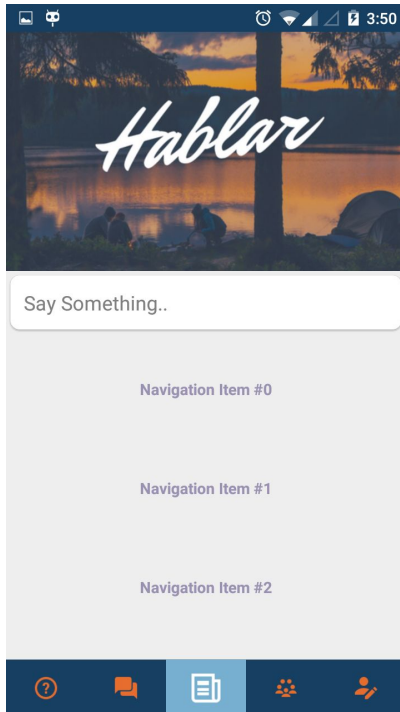
**Sign Up Page** - used to create account for the hablar application. Requires username, official ID number, official email address, and a suitable password.

## Screen 2

A screenshot of the 'Sign In' page. It features a dark green background with a subtle pattern. In the center is a light gray rectangular box containing two text input fields labeled 'Email' and 'Password'. Below these fields is an orange button with the text 'LOGIN'. Below the 'LOGIN' button, there is a link that says 'Forgot Password ?'. At the bottom of the gray box, there is a link that says 'Don't have an Account ? SIGN UP'.

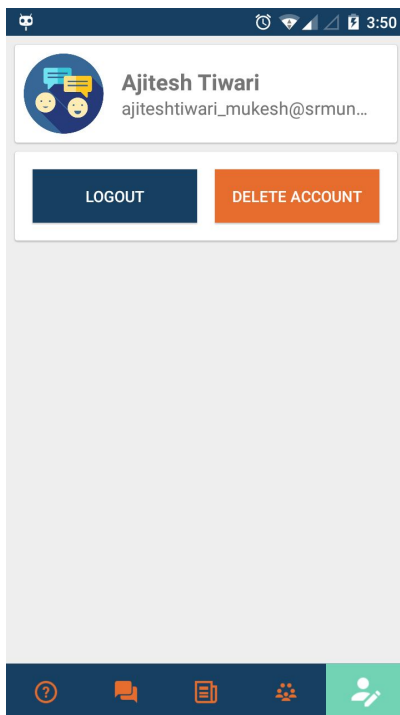
**Sign In Page** - used to sign in into the app. Requires the email address and password given during sign up.

### Screen 3



**Feed Page** - used to post updates on the feed page and also watch for updates within your organization.

### Screen 4



**Profile Page** - used to showcase your profile and also sign out or delete account if required.

Add as many screens as you need to portray your app's UI flow.

## Key Considerations

**How will your app handle data persistence?**

Content Provider can be used for data persistence although I am using firebase real time database which has in-built data persistence capability (<https://firebase.google.com/docs/database/android/offline-capabilities>).

**Describe any corner cases in the UX.**

When a user restarts the application after last login my app directly takes him to the feed page rather than the login screen.

Although if the user logs out in his previous login, the app restarts to the login page.

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

Glide- for loading images - because it is easy to use.

NavigationBar- for Navigation UI - because it suits my problem statement.

**Describe how you will implement Google Play Services.**

I am using Firebase for complete development of my application.

Firebase Auth - For creating and authenticating user accounts.

Firebase Database - For storing and retrieving real time data.

Firebase Analytics - For analyzing user behaviour.

## Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

## Task 1: Project Setup

- Create a new project.
- Include firebase in the project
- Enable firebase Auth, and Database.

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

- Build UI for Login Activity
- Build UI for Login and Register Fragments
- Build UI for Main Activity
- Build UI for fragment Profile Page
- Build UI for fragment Feed Page
- Build UI for Details Activity.

## Task 3: Create Login Activity

After including the firebase Auth in your project it's time to work on the login flow.

Following cases are possible in login flow

- Register
- Login
- Forget Password
- Verify email address
- Resend Verification email

### Login flow -

Register -> Send Verification Email -> Users verifies the email -> Login granted -> Successfully logs in.

## Task 4: Create Main Activity

Main Activity contains many fragments responsible for one function, like

- Feed Fragment
- Contacts Fragment
- Profile Fragment
- About Fragment

We need to implement NavigationTabBar library for this activity.

## Task 5: Sync everything with Database

Firebase Database is used as a store for all information for the app. Tables required are -

- Organizations
- Users
- Feed

Create firebase Database tables and perform transactions accordingly.

### Flow -

- When user registers check if the organization exists in the database.
- If it exists store his details in the user table
- Also store his feed into the feed table allotted for that organization.

Add as many tasks as you need to complete your app.

---

### Submission Instructions

1. After you've completed all the sections, download this document as a PDF [ File → Download as PDF ]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone\_Stage1.pdf**"