[Code Stories](#_Toc486080373)

[Description](#_Toc486080374)

[Intended User](#_Toc486080375)

[Features](#_Toc486080376)

[User Interface Mocks](#_Toc486080377)

[Screen 1](#_Toc486080378)

[Screen 2](#_Toc486080379)

[Screen 3](#_Screen_3)

[Screen 4](#_Screen_4)

[Key Considerations](#_Toc486080380)

[Data Persistance](#_Key_Considerations)

[Activity Transitions](#_Toc486080382)

[Libraries used](#_Toc486080383)

[Next Steps: Required Tasks](#_Toc486080384)

[Task 1: Project Setup](#_Toc486080385)

[Task 2: Implement UI for Each Activity and Fragment](#_Toc486080386)

[Task 3: Add Google Sign In](#_Toc486080387)

[Task 4: Master Detail FLow](#_Toc486080388)

[Task 5: Hooking up with Firebase Realtime Database.](#_Toc486080389)

[Task 6: Add Feature Bookmark](#_Toc486080390)

[Task 7: Subscribing to topics](#_Toc486080391)

[Task 8: Home Screen Widget](#_Toc486080392)

[Task 9: Polish UI](#_Toc486080393)

**GitHub Username**: temunide

# Code Stories

# Description

Code Stories is a social networking app that lets users post and read small stories.

Code Stories helps distressed programmers to get over with it by reading stories of other like-minded programmers.

# Intended user

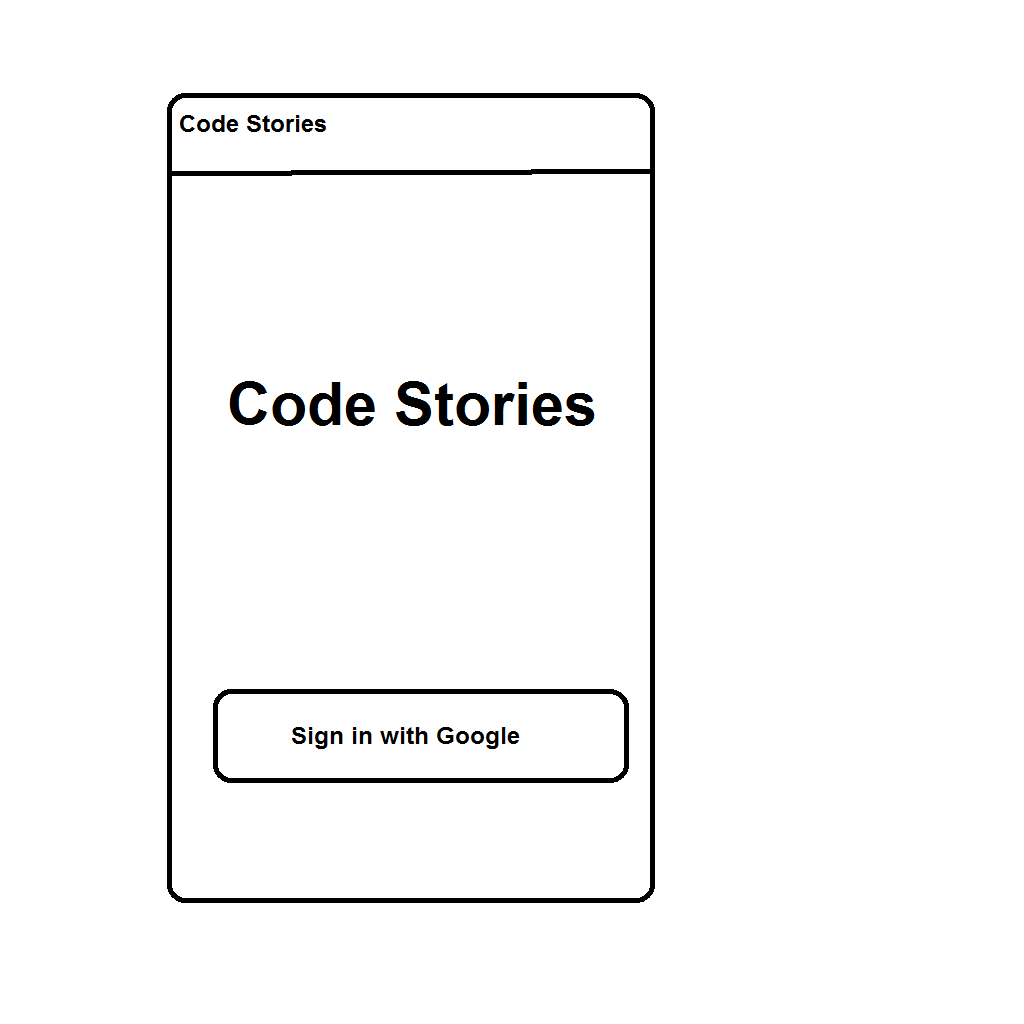
Programmers, Developers, all IT Guys.

# Features

* Shows Stories
* Posts Stories
* Bookmark Stories to read them Offline
* Allows user to subscribe to topics.
* Home Screen widget to display latest posts.
* Notifies users when a new has been posted by subscribed topic.

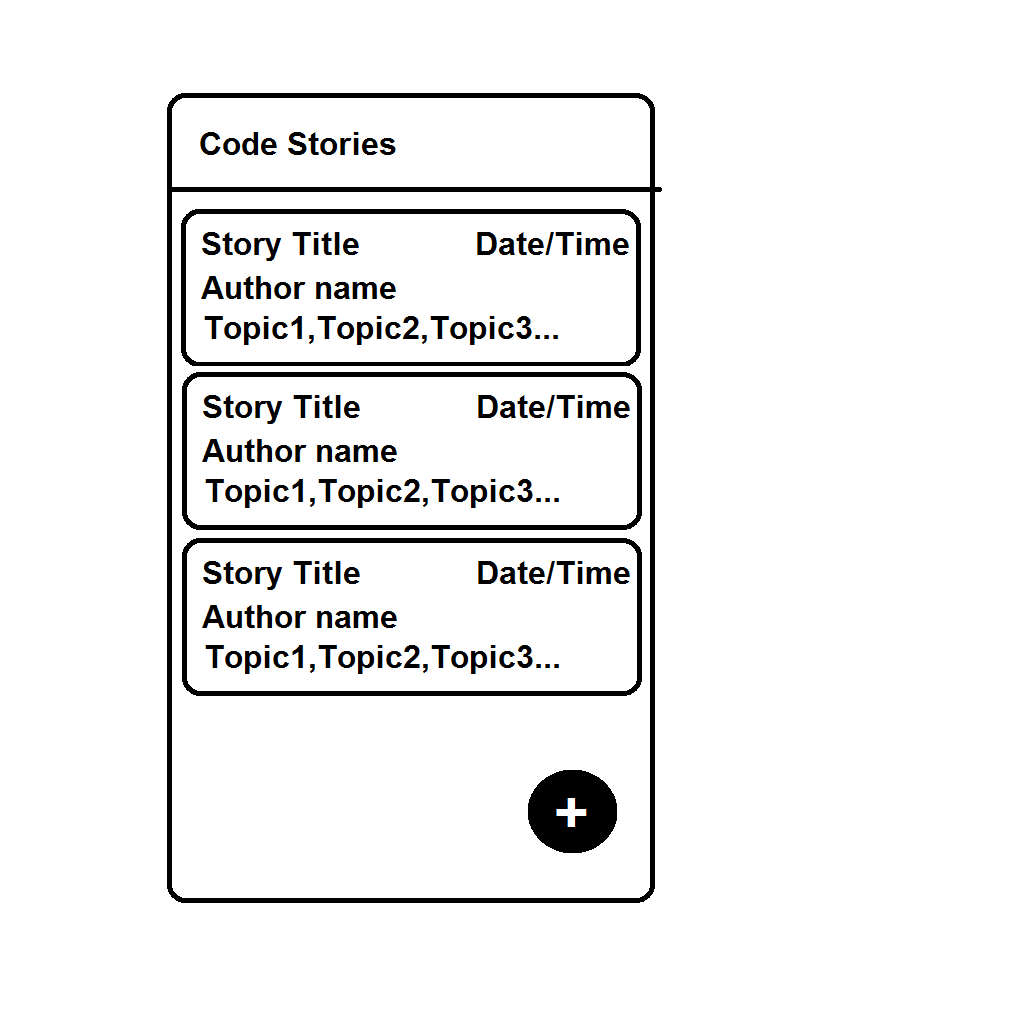
# User Interface Mocks

## Screen 1

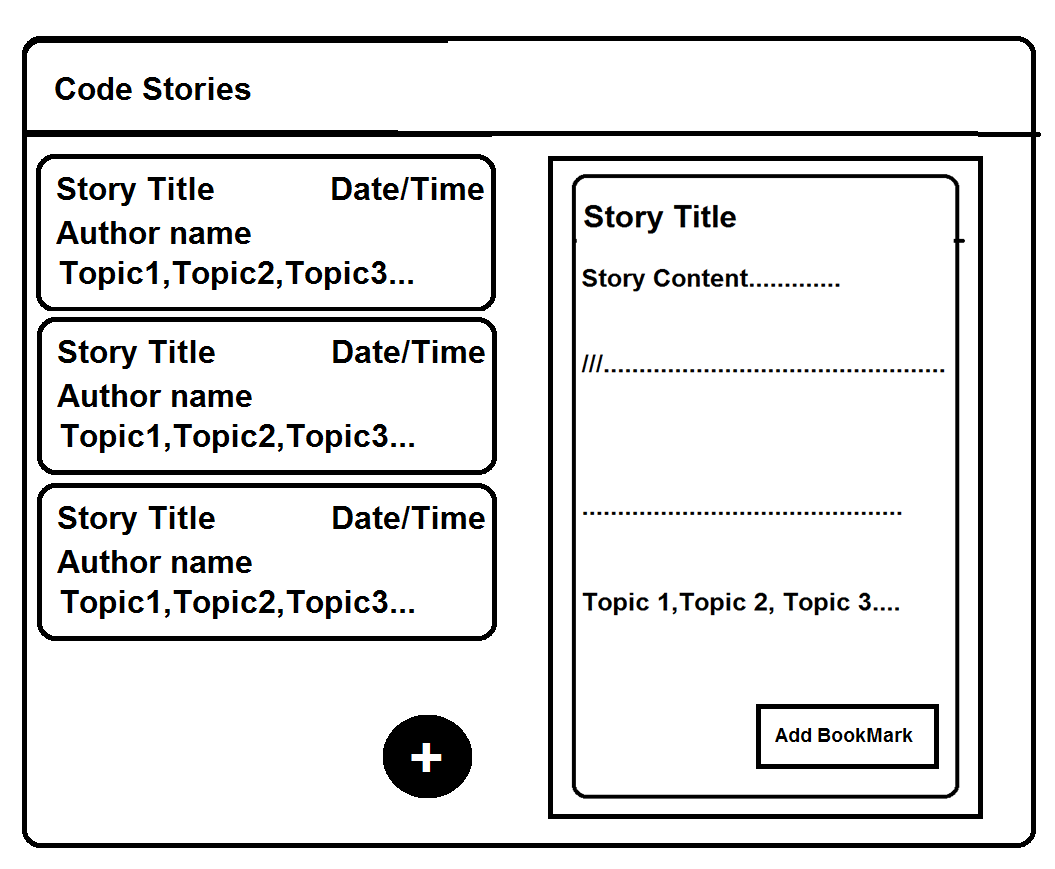


## Screen 2

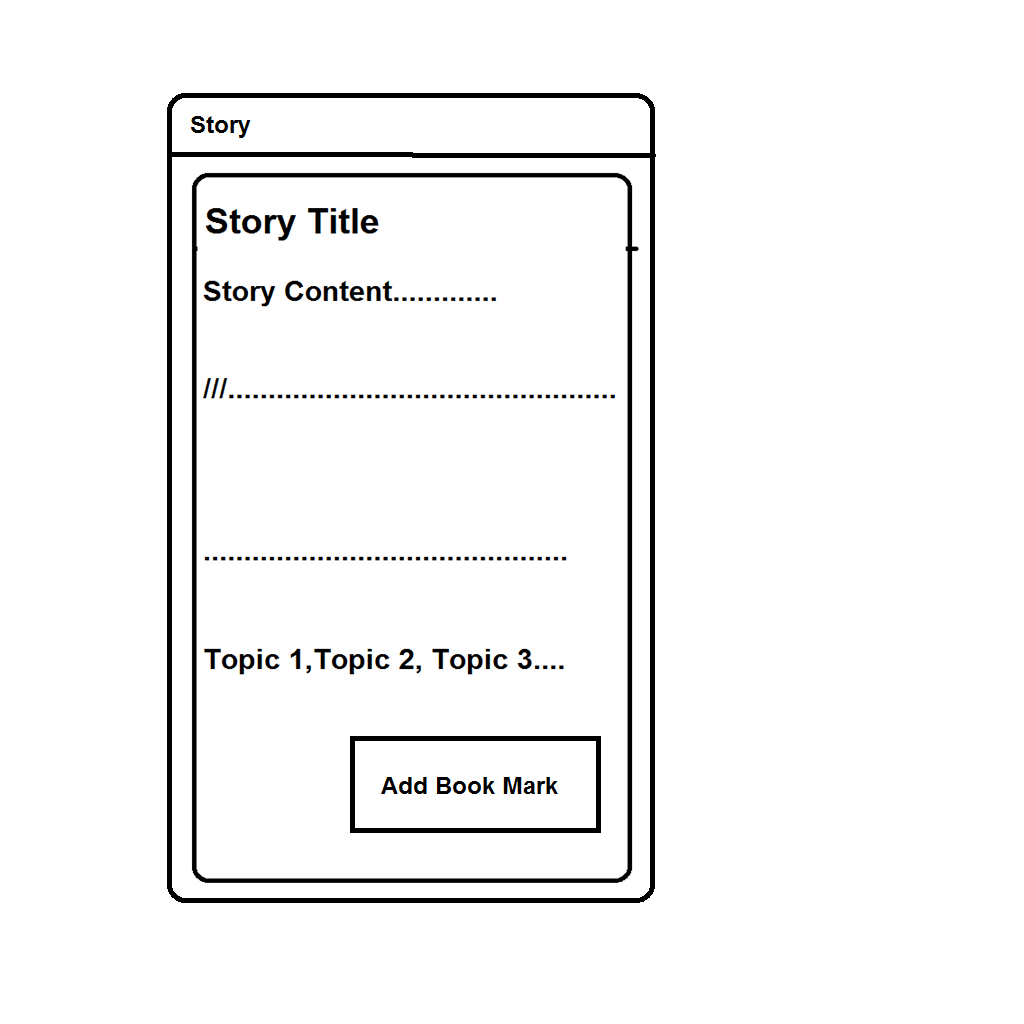
**Phone**



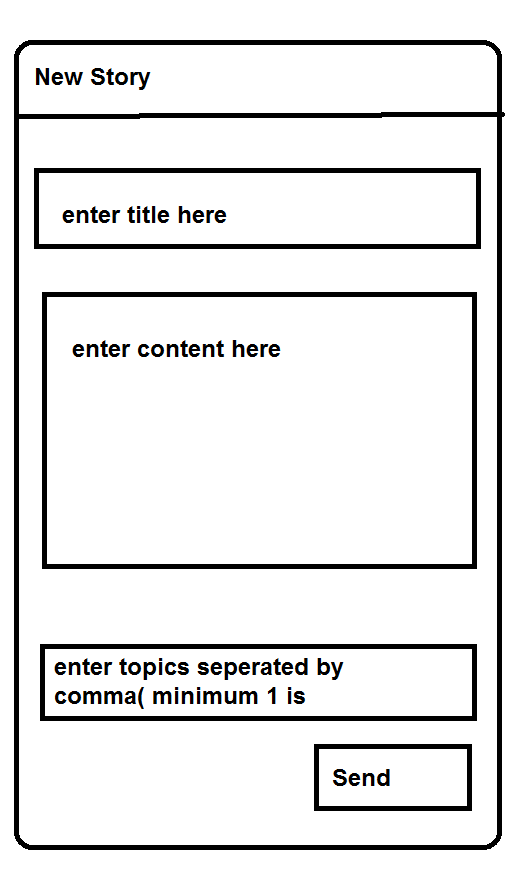
**Tablet**



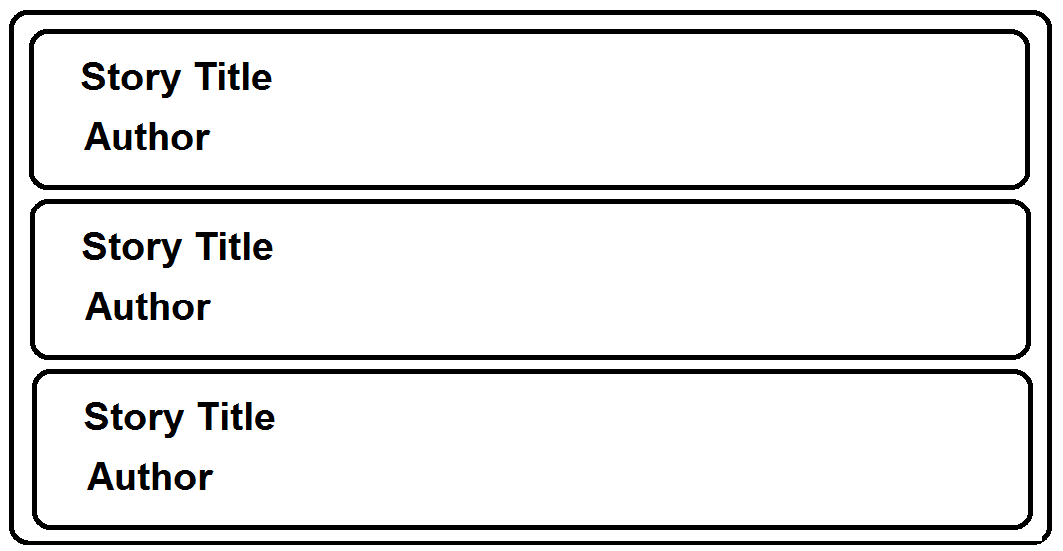
## Screen 3



## Screen 4



## Wdiget



# Key Considerations

### Data Persistence

App uses Firebase Realtime Database to sync stories among all users.

### Activity Transitions

App implements standard transitions between Activities

### Libraries used

* ButterKnife

ButterKnife simplifies the UI Binding procedures.

### Services Used

* Google Sign in

To Authenticate users of the app.

* Firebase Realtime Database

Used to store and sync the stories among all the users.

* Firebase Cloud Functions

To send cloud messages when a new story is posted.

* Firebase Cloud Messaging

To notify users when a story is added to their subscribed topic.

* Firebase Crash Reporting

To detect any misbehaviour of production builds after release

# Next Steps: Required Tasks

## Task 1: Project Setup

You may want to list the subtasks. For example:

* Create Android Project
* Integrate Version Control with the help of GitHub.

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

* Build UI for SignInActivity
* Build UI for StoriesListActivity
* Build UI for ReadStoryFragment.
* Build UI for ReadStoryActivity for NON - Tablet Devices
* Build UI for PostStoryActivity

## Task 3: Add Google Sign In

* Add Dependencies to build.gradle
* Configure Google Api’s console to enable Google Sign IN
* Add a SignInActivity as a Launcher Activity.

## Task 4: Master Detail FLow

* Create new Layout Resource for Tablets.
* Implement Master Detail Flow with Story Fragment.

## Task 5: Hooking up with Firebase Realtime Database.

* Add Dependencies to build.gradle
* Display stories from Firebase database in the List of Stories
* Add a new Story to database from PostActivity.

## Task 6: Add Feature Bookmark

* Enable offline capabilities of Firebase Database.
* Store Bookmarked stories offline.
* Add a Floating Action Button to allow users to add/remove bookmark.

## Task 7: Subscribing to topics

* Subscribe user to a topic when user clicks on a topic,
* Saved subscribed topics in Shared preferences
* Unsubscribe from topic if clicked on already subscribed topic.
* Receive a Cloud Message when a story is posted in that particular topic.
* Add Firebase Message Handler Service extending Firebase Messaging Service
* Display a Notification.

## Task 8: Home Screen Widget

* Add a Home Screen widget Provider
* Add Layout Resources and Provide info XML
* Add a service Sync data from Firebase database to home screen widget

## Task 9: Polish UI

* Polish UI to make it usable, and improve accessibility.
* Make necessary changes to build a beautiful UI.
* Follow Material design Specifications while Building UI

**Submission Instructions**

* After you’ve completed all the sections, download this document as a PDF [ File → Download as PDF ]
  + Make sure the PDF is named “**Capstone\_Stage1.pdf**”
* Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

* Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
* Add this document to your repo. Make sure it’s named “**Capstone\_Stage1.pdf**”