GitHub Username: sbetageri

NanoTales

Description

Express a story, in as few words as possible.

Intended User

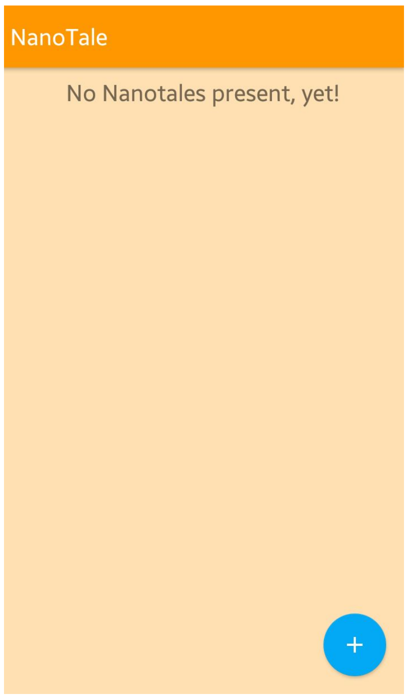
Main users are budding writers.

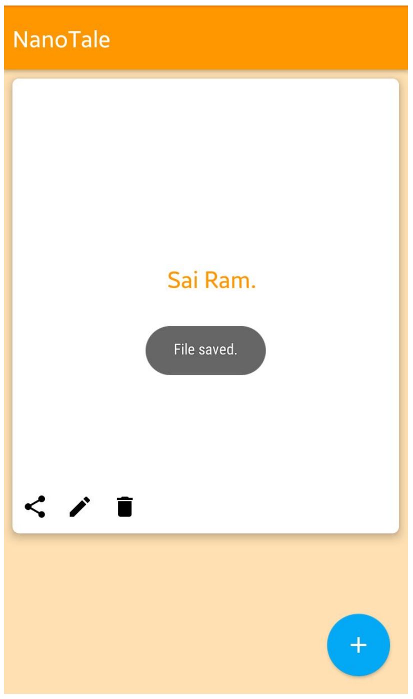
Features

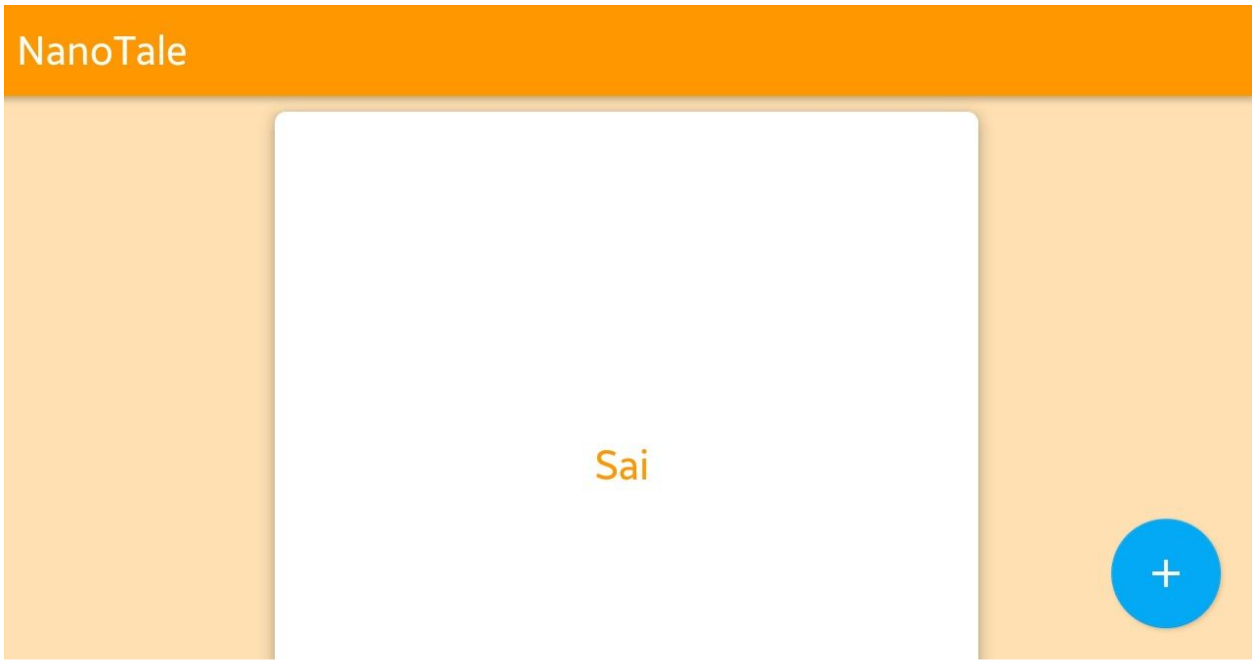
- Write extremely short stories(160 characters), called Nanotales, and save them as images.
- View saved Nanotales.
- Share Nanotales across various social media platforms.

User Interface Mocks

Screen 1 : Main Activity.

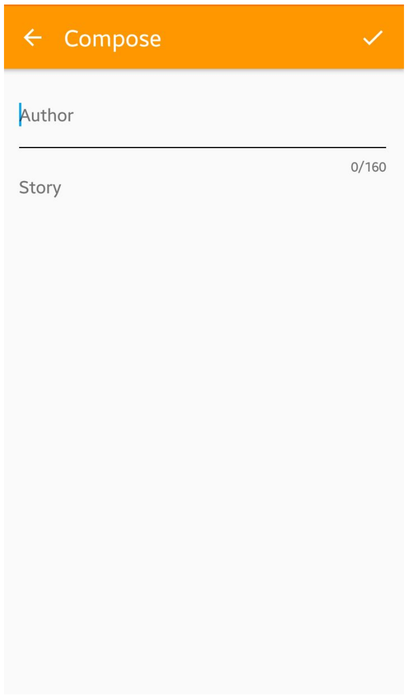






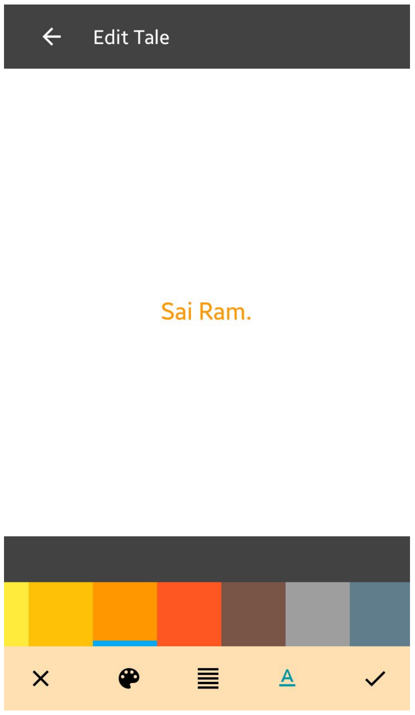
Displays all the nanotales that have been created by the user.

Screen 2 : Write Nanotale.



Allow the user to write the Nanotale.

Screen 3: Modify Nanotale.



Modify the attributes of the nanotale. Attributes are background, font color and text justification.

Key Considerations

How will your app handle data persistence?

Will be building a content provider.
Also, will be storing images on the disk.

Corner cases in the UX.

To return to the Home activity, from the activity which allows the user to write the nanotale, the user presses the back button.

To return to the Home activity from the activity which modifies the nanotale, the user presses the discard button. The icon with the ‘x’ on it.

Libraries used and reasoning for including them.

- Stetho to monitor the database.
- Butterknife to bind views.
- Picasso to display saved images.

Implementing Google Play Services.

Play services used are :
- AdMob : Displays an interstitial ad between the home activity and modifying the story
- Analytics : Keeps track of the various

Required Tasks

Task 1: Project Setup

Set the minimum SDK version to 21.

Add the following libraries to the respective Gradle file:
1. Butterknife : View Binding.
2. Picasso : Image Loading.
3. Stetho : Debuging via Chrome.

Add the following dependencies :
1. AppCompat-v7 24.2.1
2. Recyclerview-v7 24.2.1
3. Design library 24.2.1
4. Cardview-v7 24.2.1

Task 2 : Implement UI for Each Activity and Fragment.
1. Build UI for main task, with RecyclerView for saved images.
2. Build UI for card which displays the saved images and other options.
3. Build UI to write the Nanotale.
4. Build UI to modify the Nanotale’s background and font colors, along with the orientation of the text. Also add the discard and save dialogs.

Task 3 : Build UI and functionality to write Nanotale.

Allow user to enter a story(160 character limit) and author.

Subtasks :

1. Build Layout.
2. Add Listeners to monitor character limit.
3. Pass information to next activity, which allows modification of the tale.

Task 4 : Implement ContentProvider.

ContentProvider is used to store the attributes of the Nanotale. The attributes are :

- Background color.
- Font Color.
- Justification of the text.
- Name of the author.
- Text of the story.
- Date of creation, of the Nanotale.
- Path to the saved image.

Subtasks :

1. Implement Contract class for ContentProvider.
2. Implement Helper class for ContentProvider.
3. Implement ContentProvider :
    a. Implement Uri matcher.
    b. Implement Query, Delete, Update and Insert methods.

Task 5 : Build UI and functionality to modify Nanotale attributes.

Change the background color, font color and justification(left, center or right) of the tale.

Subtasks :

1. Build layout..
2. Add Color values for the background and font.
3. Build recyclerview to cycle through various colors.
4. Build list to display justification options.
5. Add functionality to edit the tale, by going back to the Write Nanotale activity.

6. Build dialogs and functinoality that confirm actions to save and discard nanotale.

Task 6 : Implement Firebase Service

Subtasks :
1. Create Firebase Analytics account.
2. Create AdMob account.
3. Integrate Firebase SDK.

Task 7 : Build Main Activity UI

Displays all the saved Nanotales, and allows user to either delete, share or modify the same nanotale.

Subtasks :
1. Build Layout.
2. Build Card layout. This card displays the individual nanotales and also displays the various actions possible.
3. Implement Adapter for the Recyclerview.
4. Implement Loader to obtain cursor to saved nanotales.