GitHub Username: ciscorucinski

Parable

Working - Title

Description

Parable is an app that allow for the congregation of YouTube videos that users have performed some form of activity on such as liking videos. YouTube, in its current form, does not yield any good mechanism to amass past videos that you have liked and might still enjoy to view.

Previously, YouTube did provide a mechanism to view all your previously liked videos that were categorized under Music; however, that mechanism has be disabled for some time now. It was a feature that I greatly miss, and have brought up to YouTube. Since this feature is not available anymore through YouTube, and it is possible to retrieve this information from the YouTube API, this app will allow the recreation of such a playlist. In addition, it will allow for the creation and watching entertainment of playlists from all categories; not just music.

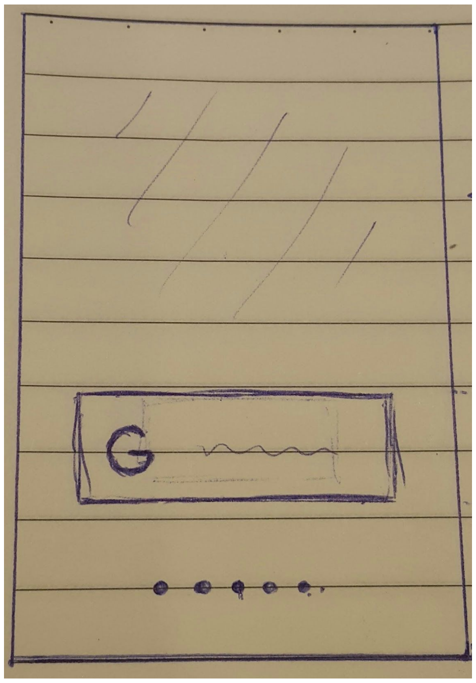
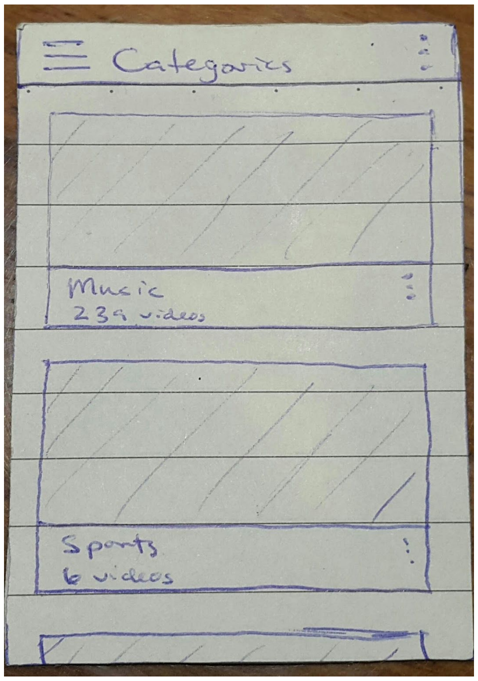
Intended User

The main user of Parable are users of YouTube that have performed actions on their service such as liking videos that they have watched.

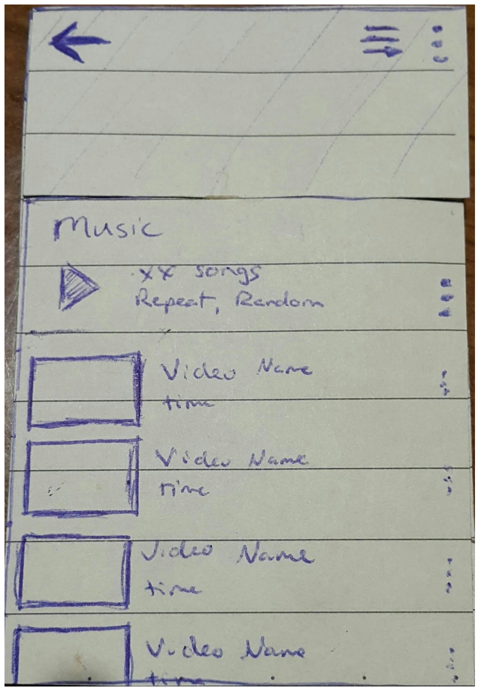
Features

- Log-in through Google Services to allow the gathering YouTube activity
- View a collection of videos based on user activity
- Categorization of videos based on YouTube categories
- Watch YouTube videos through the app
- Customization of playlists to modify the playing experience in the app
- Create permanent playlists on your YouTube channel based liked activity

User Interface Mocks

Screen 1	**Screen 2**
 Login Screen	 Main Screen
This is the first screen that a user is presented with when first using the app. It wil have a button to log into the user's Google Account, which should be linked to their YouTube account. The authentication token will be saved for future visits; therefore, the user should not be presented with this screer unless they log out. A polished version of this screen will have illustrations demonstrating the capabilities of the app to the user.	This is the first screen that users will be presented with on a regular basis. With the exception of logging into the user's Google account, this will be the screen that a user will start on. It displays a list of all the different YouTube categories of videos that the user has liked on YouTube. The individual categories are displayed as Cards. Also, there will be a navigation drawer with a few options. A polished version of this app will have several more options displayed to the user that will gather data from different endpoints of the YouTube Data API.

Screen 3a



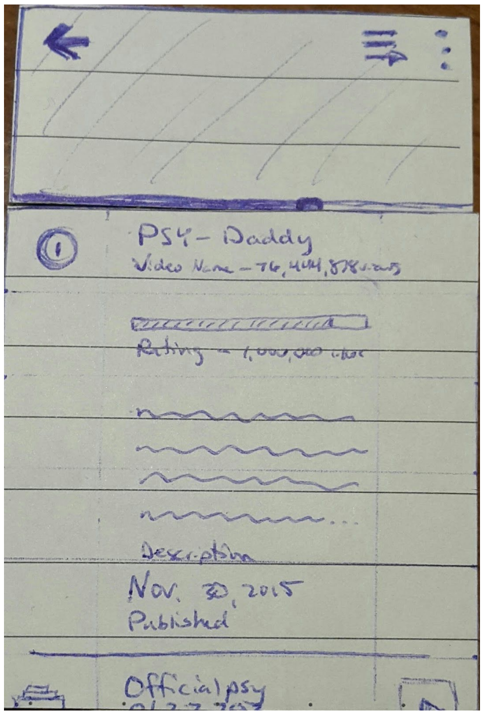
Video Player / Playlist Activity

This is the first screen that is displayed to the user after they click on a category from the previous screen. The top part of the screen will initially be just a toolbar with a solid color background; however, when the user selects a video from the bottom part, then the toolbar will become a hybrid YouTube Video Player. The video player will be chromeless (meaning won't have its own controls for interaction). To play or pause the video, the user will have to click on the appropriate control just under the video player. Also, the video player will transition into a final state where the toolbar controls will disappear after a short while to all no visual blocking of the video.

The bottom of the screen will just be a list of videos that the user liked in YouTube that are within the category that was selected. The top of the bottom part will be a header that gives some information about the selected category, and provides controls to play and pause the video. Hidden in the header are overflow actions to change the playing style of the videos such as Repeat and Random.

Lastly, on the Toolbar / YouTube Video view, there are two special actions. Within the Actic Overflow menu, there will be an option for users to create a new playlist on YouTube basec on the list in the bottom section. The second action is near the Toolbar's overflow menu ico that allows a user to switch between the playlist screen and the next screen (3b) presented below.

Screen 3b



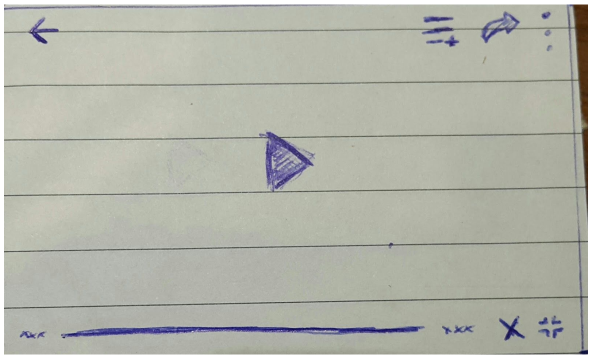
Video Player / Video Information Screen

The is a secondary screen that a user can bring up by clicking on the icon next to the Action Overflow menu icon on the Toolbar. This will change out the bottom of the previous screen (the playlist) and replace it with the selected video's details that accompany every video on Youtube.

The playlist can be brought back up by clicking on the same icon in the Toolbar.

A polished version of the app will have the ability of the user to subscribe to and view the YouTube page of the creator of the selected video.

Screen 3c



Video Player Screen - Horizontal Orientation

When the user rotates either screen 3a or 3b while a video is selected, the user will be presented with a chromed YouTube Video Player control. They will be able to use the native controls that the standard YouTube app allows them to do with videos.

If a user has not selected a video yet and they get to this screen, then the first video in the playlist will be watchable. The user only needs to press play.

Key Considerations

How will your app handle data persistence?

Data persistence will be handled with an SQLite Database and backed with a Content Provider to access the data. All data will be retrieved from the YouTube Data API initially; however, the user can generate some data to be persisted through the use of user-modified playlists that were auto-generated with help from the YouTube Data API.

Describe any corner cases in the UX.

1. If a user does NOT have a Google Account
    a. A Google account is the minimal level of interaction a user needs to perform to be able to log into YouTube. A user will not be able to sign in to my app if they don't have a Google Account. I will need to provide a small mechanism near the login to remind the user of this.

2. If a user does NOT have any YouTube Activity
    a. YouTube activity, in the apps current iteration, will need at least some form of user interaction. The current iteration will need to have LIKED videos by the user. If no Liked videos are found, then I will have to display an empty view to the user that will let them know what they need to do to use my app (i.e. Like videos on YouTube under their account).

3. If a specific category from YouTube does NOT have any user interaction
    a. The UI will either show a disabled look to that category, or not display anything related to that category. This will be looked into further while development to decide the better UX experience. In fact, a better alternative could be found and implemented.

4. The phone does NOT have Internet connectivity
    a. In the current iteration, a Snackbar will be displayed to the user to indicate that this app will need Internet connectivity to be used. The Snackbar should allow a user to connect online with ease. They should still be able to navigate around the app, but internet-centered actions will be unable to be completed.

Describe libraries to be used.

- Support (Design) Libraries : Simplify the coding and Materialization of my app.
- YouTube Player API: Playing of YouTube videos natively in my app.
- YouTube Data API: Retrieval of my user’s YouTube data & create Playlists for them.
- ReactiveNetwork: Easily determine if a user has Internet connectivity and react.
- Retrofit 2 : Simply networking logic and connects directly to YouTube endpoints.
- Timber : Advanced Logging capabilities to make debugging easier.

Next Steps: Required Tasks

Task 1: Project Setup

- Create project
- Configure libraries

Task 2: Implement UI for Each Activity and Fragment

- Build UI’s for…
  - Login Activity
  - Main Activity
  - Playlist / Video Activity
- Create the XML resources for all the UI’s

Task 3: Develop Build Variants

- Create Build Variant for Debug and Release.
- Implement special Logging capabilities in the debug build

Task 4: Create Data Access Layer

- Create an SQLite Database to store users YouTube data
- Develop the Content Provider to access data

Task 5: User Authentication

- Review the Google API documentation to figure out which Authentication API they provide is best suited for this application. Currently it appears that Google Sign-In is the best option
- Retrieve user data from Google that will allow me to access their YouTube data

Task 6: Retrieve User and Video Data

- Gather user’s liked video content from YouTube along with the relevant Video data
- Store that data within the database

Task 7: Implement the YouTube Player API

- Allow the user to play the video within the app
- Implement video to allow for different viewing experiences based on orientation, and focus

Task 8: Implement logic to manipulate videos in the App

- Allow the user to easily add, remove, replay, randomize, (etc), videos

Task 9: Create Playlists for users

- Using the YouTube Data API, create a playlist on the user’s YouTube account based on the videos they have arranged