EarCasts

Udacity Nanodegree - Capstone Stage 1
Android Application

By

Layale Matta

Github username: yalematta

[This document is created as part of Android Nanodegree Capstone Stage 1]

Description

We’ve optimised your Podcast player for daily commute.
Listen to your favorite podcasts assembled into one collection.

Intended User

For people who enjoy listening to Podcast on their long drives / trips / rides.

Features

This application will be developed using Java Programming Language.

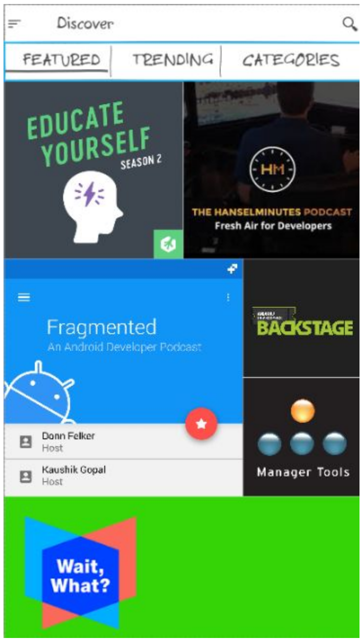
App will have the following features:

- Subscribe to any podcast in iTunes.
- Play/Pause podcasts using Voice Commands.
- Browse podcasts by charts, networks and categories.
- Share and spread the word with podcast and episode sharing.

User Interface Mocks

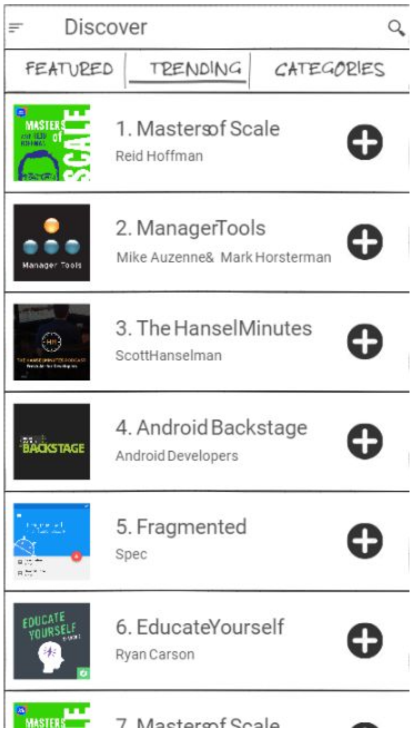
Home Screen

In this Home Screen, the featured iTunes podcasts are showing. The user can also search for podcasts in the Toolbar. By clicking on the podcast picture, the user can access the Podcast Details Screen.



Trending Podcasts Screen

In the second tab, the trending iTunes podcasts are showing. By clicking on the (+) button, the user will subscribe to the podcast. And by clicking on the podcast row, the user will access the Podcast Details Screen.



1. Mastersof Scale
Reid Hoffman

2. ManagerTools
Mike Auzenne& Mark Horsterman

3. The HanselMinutes
ScottHanselman

4. Android Backstage
Android Developers

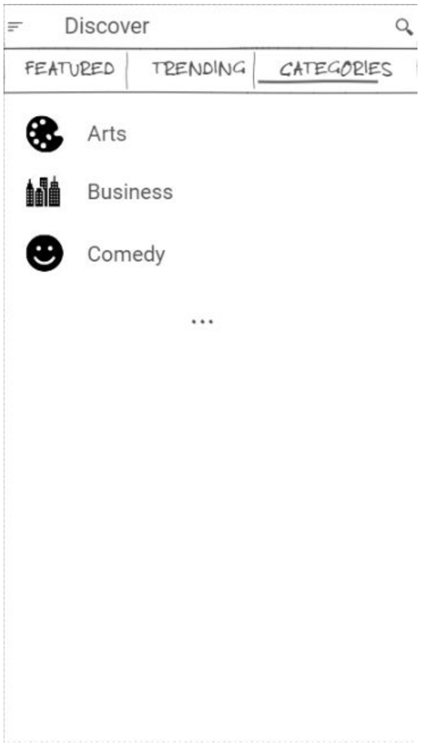
5. Fragmented
Spec

6. EducateYourself
Ryan Carson

7. Mastersof Scale

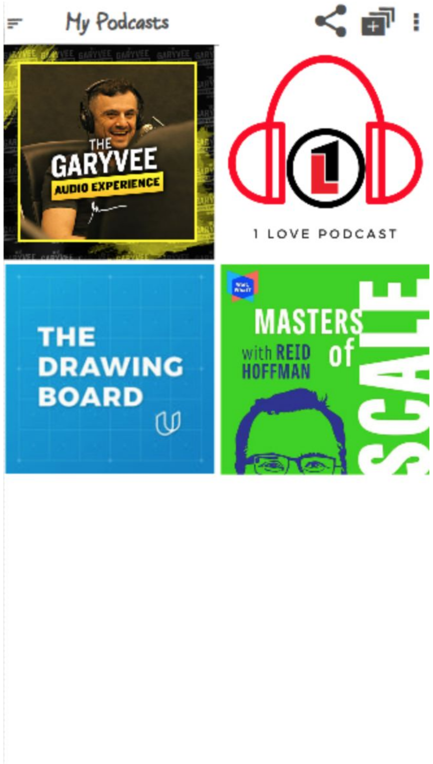
Podcasts by Categories

In this third tab, iTunes podcasts are grouped by category. When the user clicks on a category, a new screen will open. This screen will have a similar design of Screen 2, showing Podcasts of the same category.



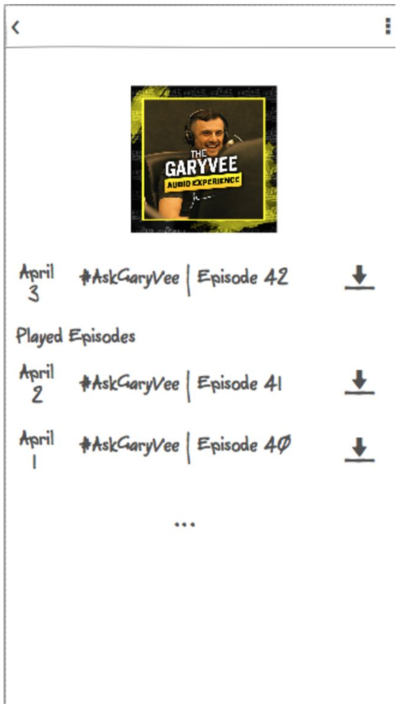
My Podcasts Screen

This is the My Podcasts Screen. The user can add more podcasts or share his own.



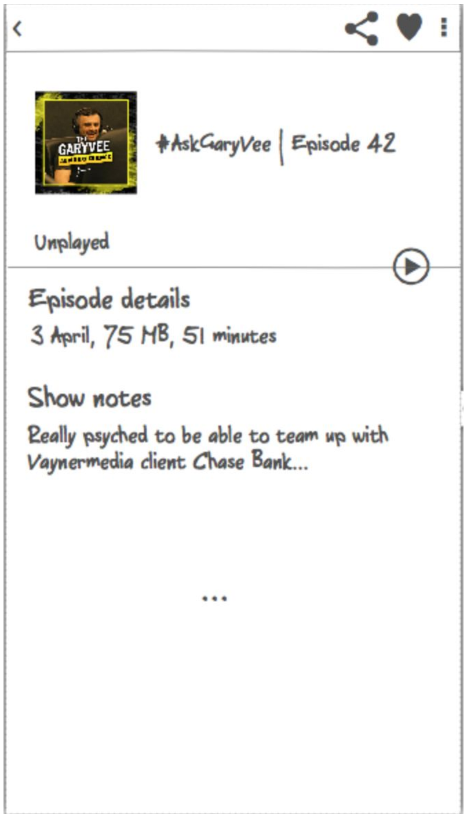
Podcast Details Screen

This is the Podcast Details Screen. Once an episode is downloaded, the user can play it.



Podcast Episode Details Screen

Podcast Episode Details Screen. It shows the notes, the date this episode was published, and its length. The user can play, favorite and share this specific episode.

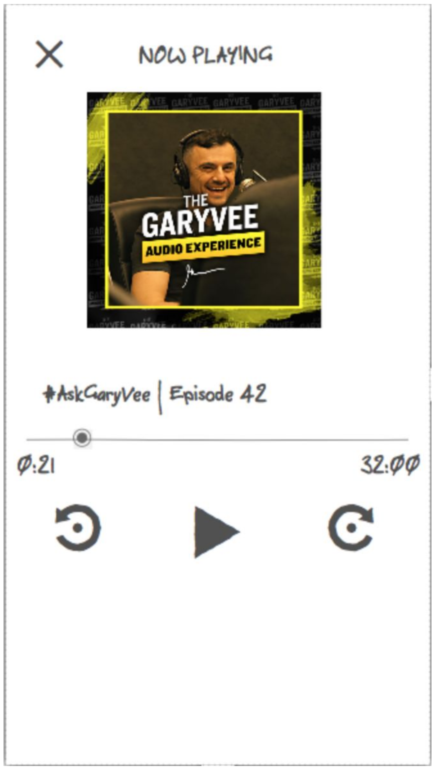


3 April, 75 MB, 51 minutes

Really psyched to be able to team up with Vaynermedia client Chase Bank...

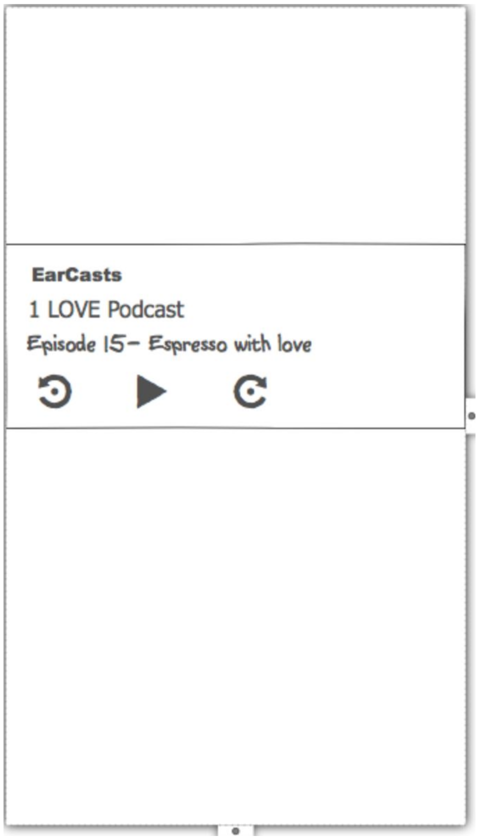
Podcast Player Screen

Podcast Player Screen enables the user to play the audio, make it faster or slower.



App Widget - mini player

The widget enables the user to play the podcast from the Home screen.



Key Considerations

Data persistence strategy

Data persistence will be done using the following ways:

1. Shared Preferences
2. Database
3. Firebase realtime database
    a. Favorite episodes will be stored in Shared Preferences
    b. Podcasts subscriptions will be saved in local database
    c. Podcasts will be retrieved from remote database

Corner cases

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

Third party libraries

1. Gradle: v4.1
2. Android Studio: v3.0
3. ExoPlayer for playing Podcasts: v2.8.0
4. GSON for serializing and deserializing POJOs: v2.8.4
5. Volley for making networking easier and faster: v1.1.0
6. Design Support Library for material goodness: v26.1.0
7. Glide to handle the loading and caching of Images: v4.7.1
8. Firebase for crash analytics and push notifications: v15.0.2
9. Butterknife as a view injector for avoiding findViewById: v8.8.1
10. Dagger2 as it helps in dependency injection and in making testing easier
11. Room for providing an abstraction layer over SQLite for fluent database access: v1.1.1

Google Play Services

1. Analytics
2. Firebase Storage
3. Firebase authentication
4. Firebase Realtime Database

Firebase (BAAS) will allow rapid development of app.
Firebase bring app in live with zero server side coding.

Next Steps: Required Tasks

Task 1: Project Setup

- Configure libraries
- Select minimum API version required
- Add all the required dependencies to my project
- Add all strings in the strings.xml file & enable RTL layout switching on all layouts

Task 2: Design database Schema and implement Content Providers

- Design the database and individual tables schema for the application.
- Implement content providers to access and manage the database.
- An IntentService will be used to fetch data from database and show it on widget.

Task 3: Implement UI for Each Activity and Fragment

- Build UI for **MainActivity**
    - Setting up the TabLayout with a ViewPager
    - Build the UI for 3 Fragments: **Featured**, **Trending** and **Categories**
        - Setting up 3 RecyclerViews
        - Their Adapters + Layouts
- Build UI for **Podcast By Category Fragment**
    - Setting up its RecyclerView
    - Its Adapter + Layout
- Build UI for **My Podcasts Fragment**
    - Setting up its RecyclerView
    - Its Adapter + Layout
- Build UI for **Podcast Details Fragment**
    - Setting up its RecyclerView
    - Its Adapter + Layout
- Build UI for **Podcast Episode Details Fragment**
    - Setting up its RecyclerView
    - Its Adapter + Layout
- Build UI for **Podcast Player Activity**
    - Setting Up Exoplayer View
    - Customize Exoplayer UI
- Build UI for **Favorites Fragment**
    - Setting up its RecyclerView
    - Its Adapter + Layout

Task 4: Implement Functionality for each Activity and Fragment

- Implement the functionality of **MainActivity**
  - Search for Podcasts functionality
  - Link the Models to the Views in Fragments: **Featured**, **Trending** & **Categories**
- Implement the functionality of **My Podcasts Fragment**
  - Share Podcast Functionality
  - Add / Subscribe to Podcast Functionality
  - Link the Models to the Views
- Implement the functionality of **Podcast Details Fragment**
  - Download Podcast Episode Functionality
  - Link the Models to the Views
- Implement the functionality of **Podcast Episode Details Fragment**
  - Share Podcast Episode Functionality
  - Favorite Podcast Episode Functionality
  - Play Podcast Episode Functionality
  - Link the Models to the Views
- Implement the functionality of **Podcast Player Activity**
  - Create + Add the MediaSession
  - Create + Add the MediaController
  - Set up Exoplayer Event Listeners
  - Add MediaStyle Notification
- Implement the functionality of **Podcast Player Activity**
  - Unfavorite Podcast Episode Functionality
  - Link the Models to the Views

Task 5: Setting up Animations

- Setting up meaningful animations
- Shared Element Transitions while launching another Activity

Task 6: Testing, review and rework

- Test for corner cases
- Test with real world users
- Write unit tests (if required)
- Review code and rework as necessary