**GitHub Username**: riochuong@gmail.com

# GET YOUR CASTS

## Description

This app helps you to manage all your favorites audio/video podcast channels so you can listen to it anytime anywhere.

## Intended User

This app is for all podcast lovers who like to listen to their favorite channel on their little free time. With all the features provided, podcast lovers can easily viewing/listening their downloaded episodes or stream the content live.

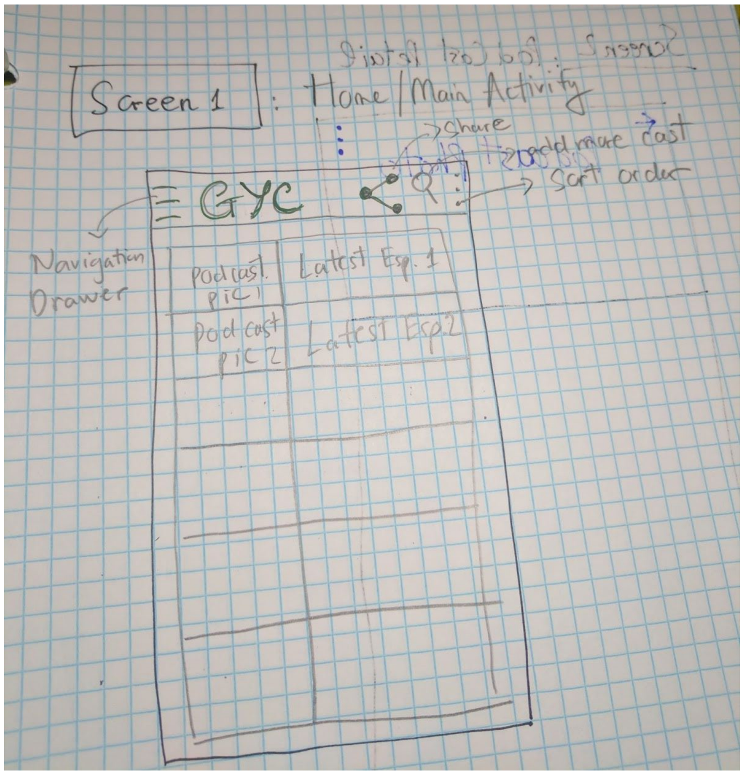
## Features

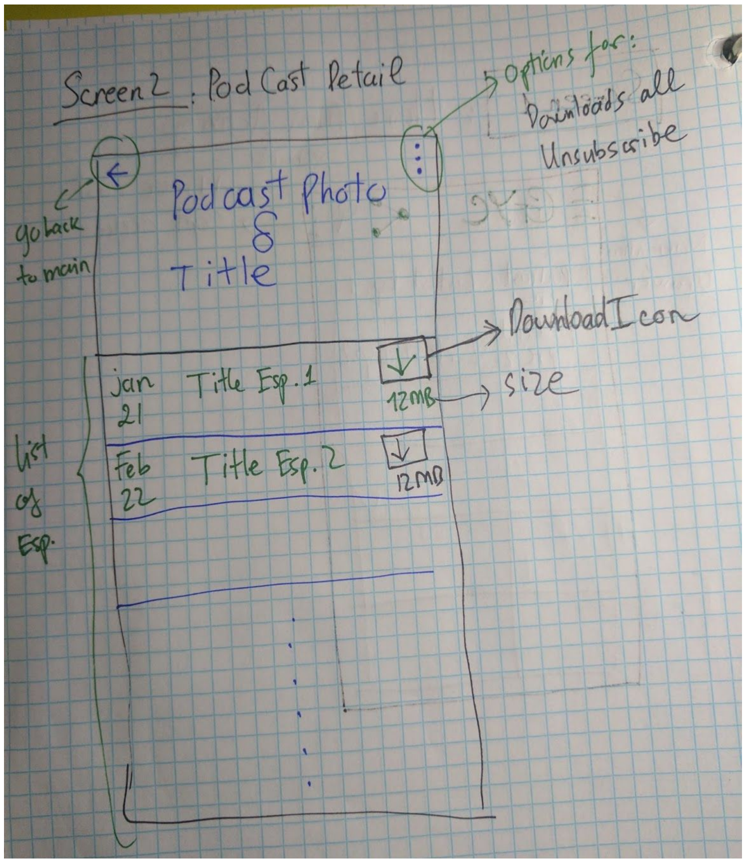
Main features:

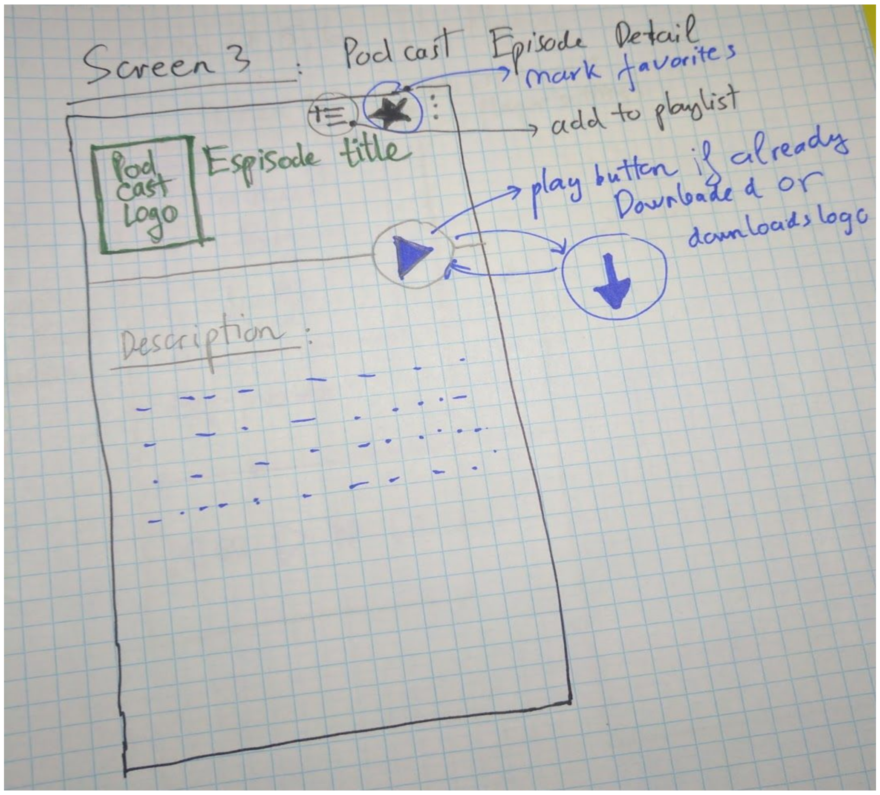
- Searching for any podcasts.
- Subscribe and Unsubscribe to any podcasts
- Supporting both audio and video podcasts.
- New Episode Notifications for subscribed podcasts.
- Offline playback for downloaded episodes.
- Filtering and Organizing your favorite and new release podcasts.

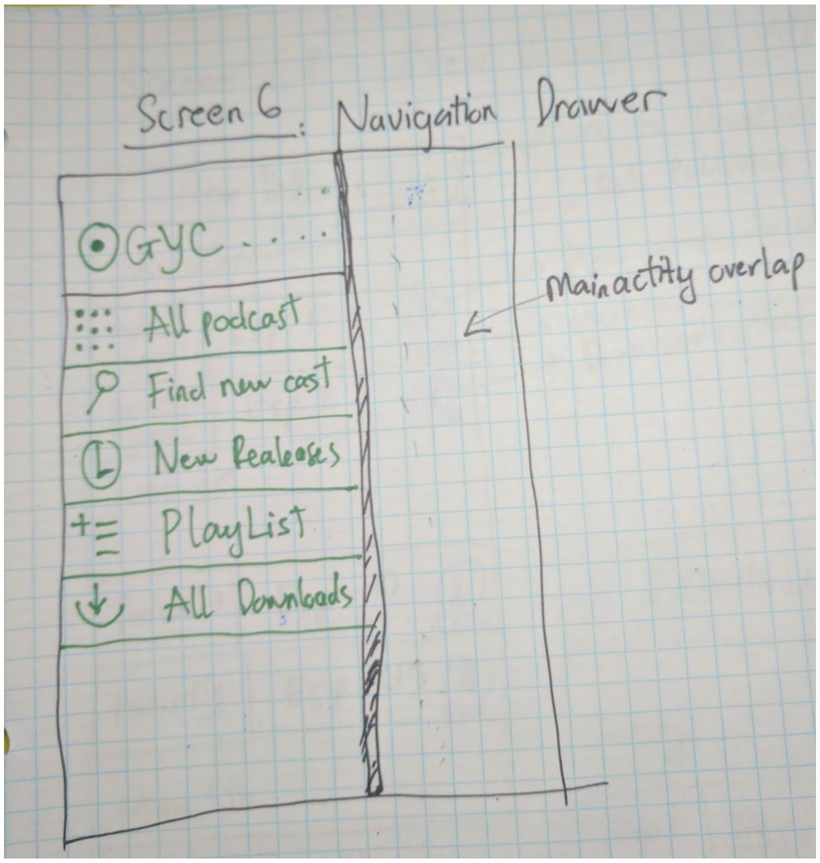
## User Interface Mocks

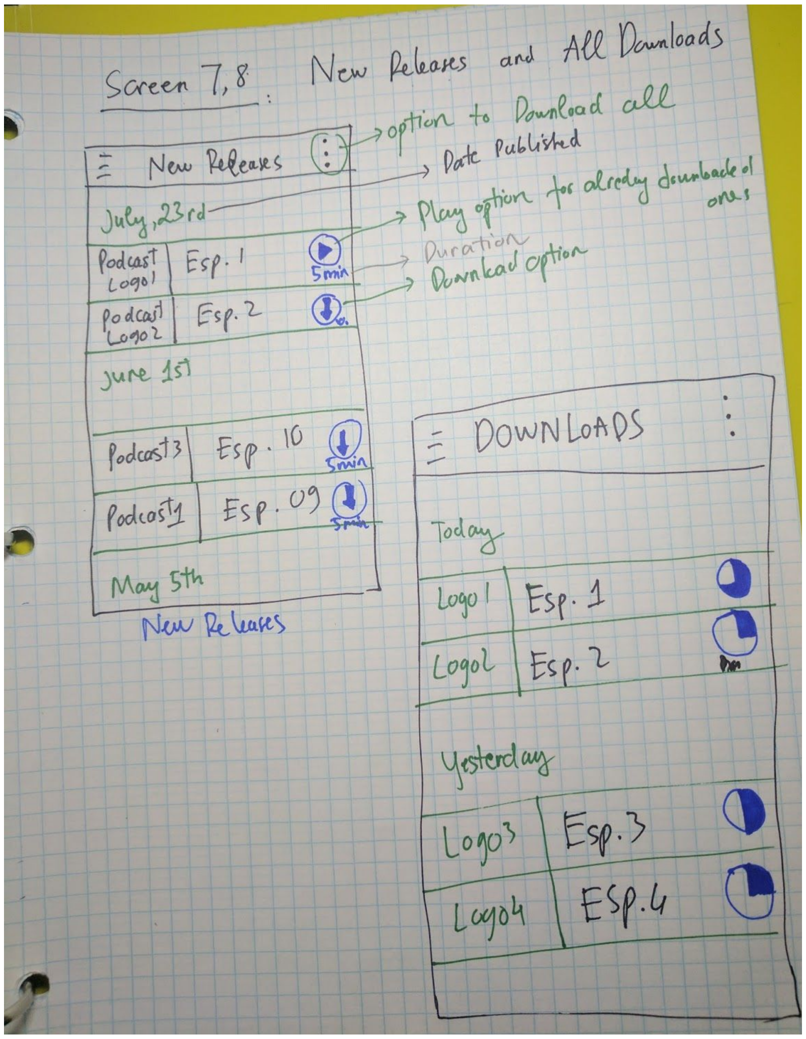
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.




Screen 2 : Pod Cast Detail
options for:
Downloads all
Unsubscribe
go back to main
Podcast Photo & Title
Download Icon
size
Jan 21 Title Esp. 1 12MB
Feb 22 Title Esp. 2 12MB
list of Esp.


Screen 3 : Pod cast Episode Detail
mark favorites
add to playlist
Pod cast logo
Espisode title
play button if already Downloaded or downloads logo
Description :


Screen 6: Navigation Drawer
GYC . . . .
All podcast
Find new cast
New Realeases
PlayList
All Downloads
Mainactity overlap



## Key Considerations

### How will your app handle data persistence?

To handle data persistence, we will use Realm DB backend along with Content Provider. We will keep one DB table for each podcast and one table for all podcast episodes which can be easily filter with type of podcast. Also, date downloads and released are also part of the episode db.

### Describe any corner cases in the UX.

There can be some corner cases which related to start and stop the media player while switching between different screens. Therefore, I decide to make the media player a

background service which can continue playing the soundtrack even if we are not at the media player screen.
Also, partial of the media player activity will be always on top of the bottom 10% screen area of the application. This helps users to easily navigate to the playlist as well as media player.

Also, updating the progress circle icon for each of the episode also a little bit tricky. We need an asynchronous observer pattern framework like RxJava to ease the updating task. Moreover, the database also needs to keep track of the progress of the episode to help display this information again when the application restarts.

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

_Glide - loading image seamlessly for different size.
_ExoPlayer - media player support playing video/audio podcast.
_RxJava/RxAndroid 2 - reactive framework helps to update UI items asynchronously after network requests or database query operations
_Dagger 2 - dependency injection for probably network service
_Kotlin plugin support.

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

_I will not use any google play services api for now.

## 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

_Start the new project from Android Studio 3.0 canary 6 with Kotlin support (I want to start using some kotlins in the app)
_Add library to gradle files: Glide, RxAndroid, RxJava, ExoPlayer, Retrofit
_Check in project to Github to start increment development process.

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

List the subtasks.
_Build UI for Screen1 as a fragment of MainActivity

_Build UI for Screen2 as a fragment of PodCastDetailActivity
_Build UI for Screen3 as a fragment for EpisodeDetailActivity
_Build UI for Navigation Drawer
_Build UI for Screen4 and 5 as fragments for PlaylistActivity
_Build UI for Screen7 as a fragment for NewReleaseActivity
_Build UI for Screen8 as a fragment for AllDownloadsActivity
_Build notification UI to notify of new episodes and download progress also.

Note: we will use MVVM architecture + new Android Architecture Components to complete these tasks.

## Task 3: **Build and Test Realm Database And Content Provider**

_Build Realm Database and Content Provider to store information about podcast and episodes.
+Design Podcast DB table and Episodes details DB table.
+Implement a DB helper for Realm DB to query the data.
+Design Content Provider for querying the Realm DB.
_Write some tests to test the DB feature like insert, remove, query, update.

## Task 4: **Build Network Related Service and Features**

_Test out different podcast api services such as:
https://www.audiosear.ch/
https://gpoddernet.readthedocs.io/en/latest/

_Design Network Service with Retrofit apis and RxJava for fetching the podcast raw data.
_Build the parser for the Raw data so we can try with different DB operators.
_Build image service with Glide to load different podcast images.
_Build module for checking new available episodes and send notification to user..
_Build download service to download.
_Design so we can use Dagger for Injection service.
_Write some tests for these features.

## Task 5: **Build the Media Player Service with ExoPlayer**

_Build Offline Play Audio Podcast Service.
_Build Offline Play Video Podcast Service.
_Implement features to keep track of each episodes progress and update the DB.
_Test these features with different size of media file and format.

_Try to design so we can use it through injection service.

**Task 6**: **Test Everything together and with different android devices screen sizes.**