Multi-Casts

The podcast app for on the go families.

- The podcast app for on the go families.

Description

Intended User

Features

User Interface Mocks
  - Screen 1
  - Screen 2

Key Considerations
  - How will your app handle data persistence?
  - Describe any edge or corner cases in the UX.
  - Describe any libraries you'll be using and share your reasoning for including them.
  - Describe how you will implement Google Play Services or other external services.

Next Steps: Required Tasks
  - Task 1: Project Setup
  - Task 2: Implement UI for Each Activity and Fragment
  - Task 3: Implement Extras
  - Task 4: Iterate

GitHub Username: galamdring

Multi-casts

Description

Multi-casts is a podcast app designed to manage podcasts and settings for multiple listeners. Group podcasts by listener or type and design your custom settings! It will be written solely in Java.

Intended User

This app is for anyone who needs control over groups of podcasts and the ability to control settings for those groups.

Features

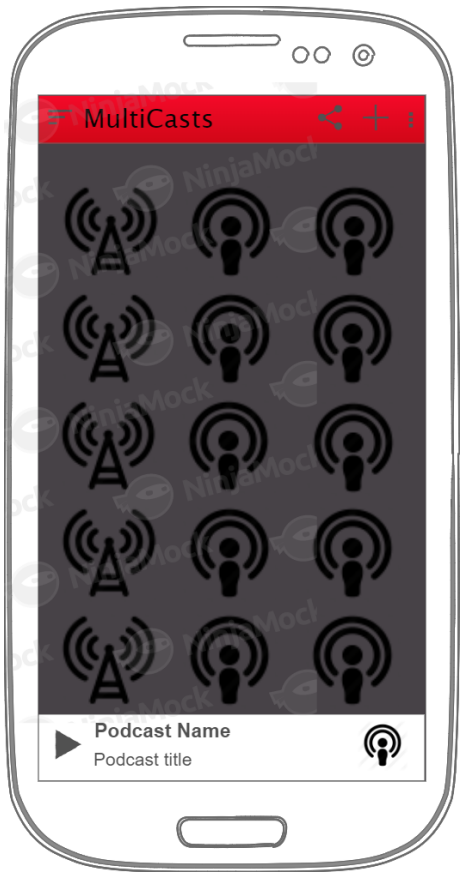
- Download or Stream podcast episodes
- Subscribe to podcasts
- Group podcasts and apply custom settings
- Play all from groups
- Playlists

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 Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

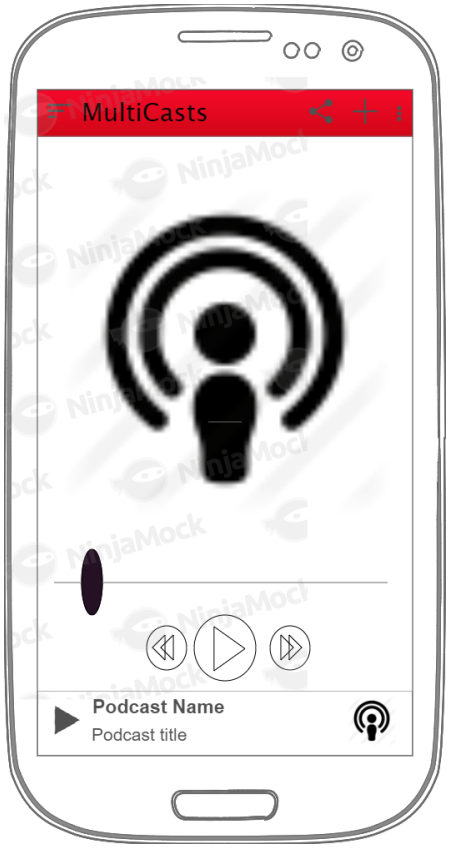
Completed with Ninja Mock

Screen 1

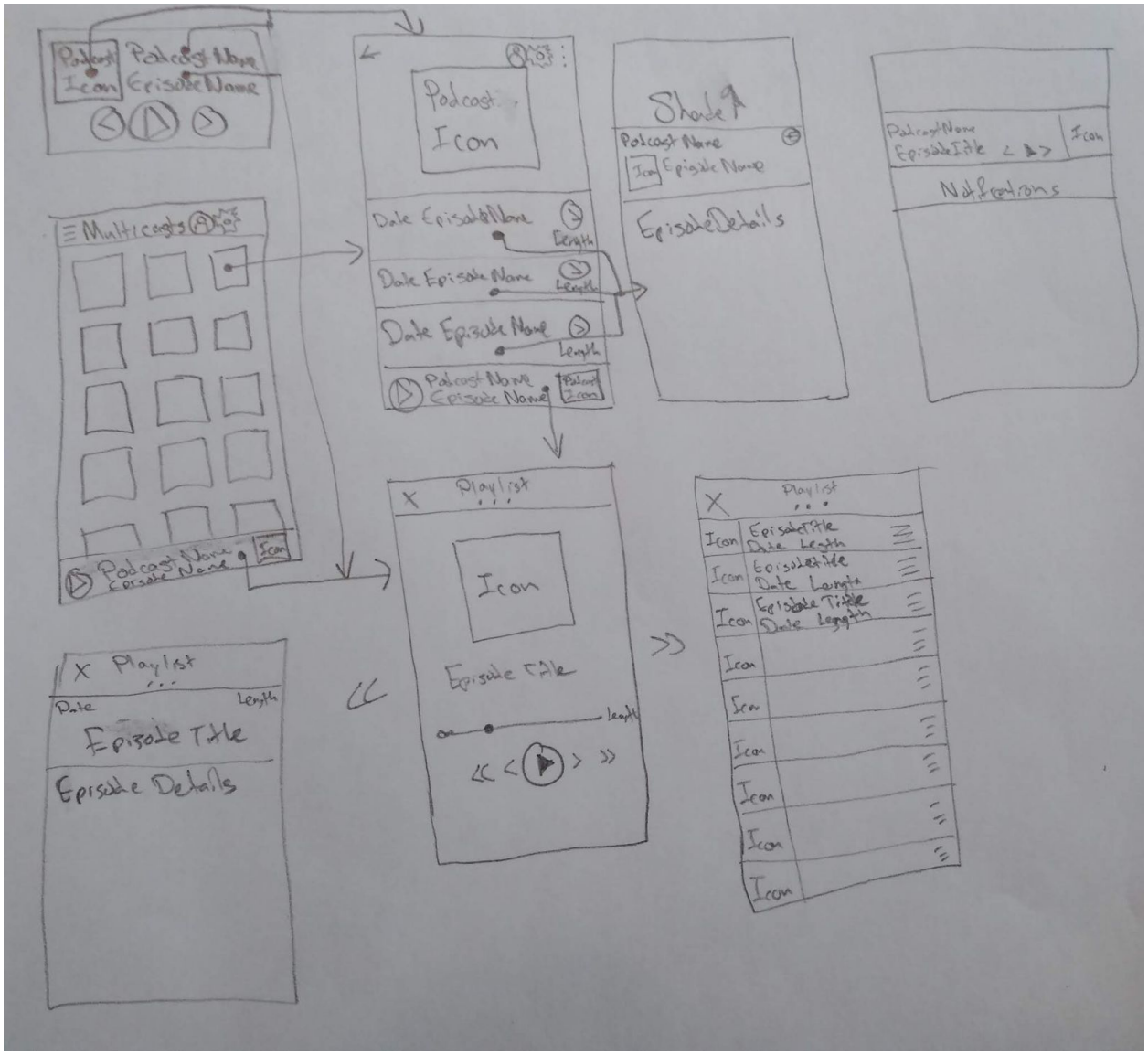


This is the main screen of the podcast app. Player controls are located at the bottom of the page, and all the subscribed podcasts are listed. Podcast icons should badge with number of new episodes.

Screen 2



This is the detail screen for the currently playing podcast. All menu items should be available in the action bar from this screen. Player controls located in main window. Pull up on base control should display playlist of podcasts.



App will have a widget and notification that can be used to control playback. Clicking anywhere other than the controls in the notification will reopen the main activity. Touching the episode name in the widget should open the playlist activity.

The playlist activity will hold three fragments, one giving details on the currently playing episode, one with the controls, and one with the playlist. The three line icon in the playlist will be the handle for reordering the episodes. Touching the X in the upper left will return the user to the main activity.

Key Considerations

How will your app handle data persistence?

Individual episodes will be downloaded to internal or external storage. RSS feed data for subscribed podcasts will be stored in a local database implemented via Room and refreshed with a background service, or manually from the UI. List of available podcasts will be retrieved from a Firebase Realtime Database Rest API, or RSS feed urls entered manually.

Describe any edge or corner cases in the UX.

Tapping on the player control at the base of the screen should take the user into the Now Playing screen. Pulling up on the control should show Up Next playlist.

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

Picasso for images, to handle background image loading. Version 2.71828
Exoplayer for media streaming. Versio 2.8.4
Firebase-ads:11.8.0 For admob banner

Describe how you will implement Google Play Services or other external services.

I will use firebase realtime database service to serve a list of podcasts. This list will contain the rss link and an icon link. I will also use the adMob service in the free version for displaying ads.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

- Create new project in Android Studio
- Configure Picasso
- Configure Firebase
- Implement background player using Exoplayer
- Implement settings groups
- Implement IntentService for retrieving podcast list
- Implement IntentService for downloading podcasts
- Implement AsyncTask for podcast search

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Podcast List
- Build UI for Now Playing
- Build UI for Add New Podcast

Task 3: Implement Extras

- Implement firebase integration for podcast list
- Create free and paid versions

Task 5: Widget

- Implement widget

Task 4: Iterate

- Test functionality
- Improve Material Design
- Improve functionality
- Ensure all strings are encoded in strings.xml
- Ensure RTL layout switching to support accessibility on RTL supported languages

---

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**"