How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: MohamedSayed9392

Mozica Player

Description

There are many music players but as I'm a music lover I faced a problem that sometimes I need to listen to the most listened tracks on my library so Mozica Player solved this problem.
You can easily sort your music by many categories like (listened count - length of track - name of the track file - track added date).
Also you can favourite any track you need and access your favourites easily.
Mozica Player feel the music ;)

Intended User

This app mainly targets Music lovers

Features

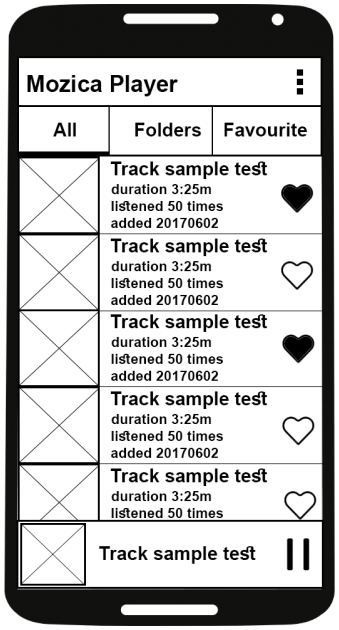
Main features of Mozica Player:

- Sort music by listened counts
- Sort music by date added
- Sort music by file name
- Sort music by track length
- Easy add tracks to favourite

User Interface Mocks

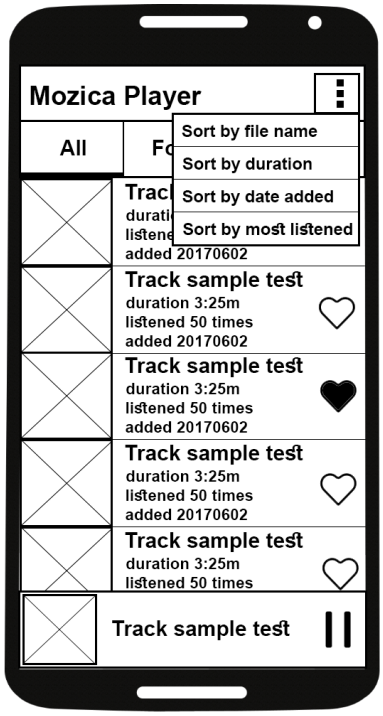
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



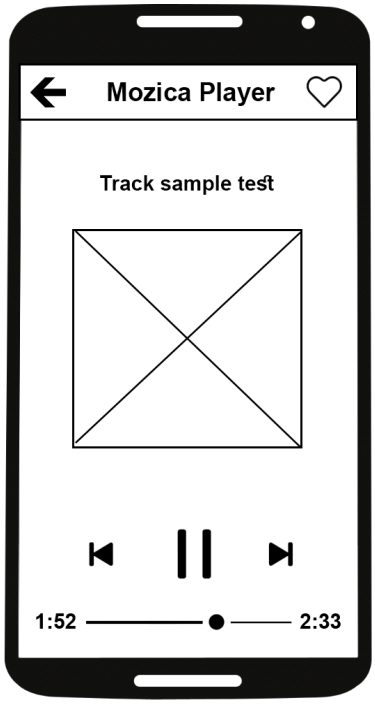
Music Library with tabs to make navigation easier between all tracks - music folders - favourite and bottom bar show the current playing track.

Screen 2



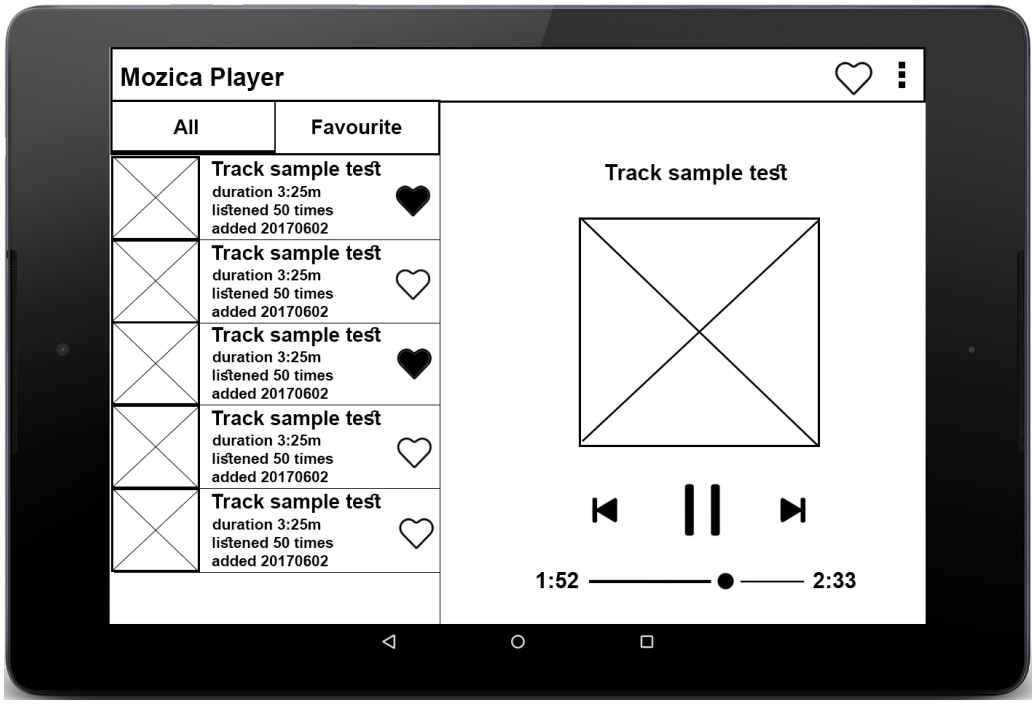
The 3 dots menu give the user options to sort music tracks by many categories.

Screen 3

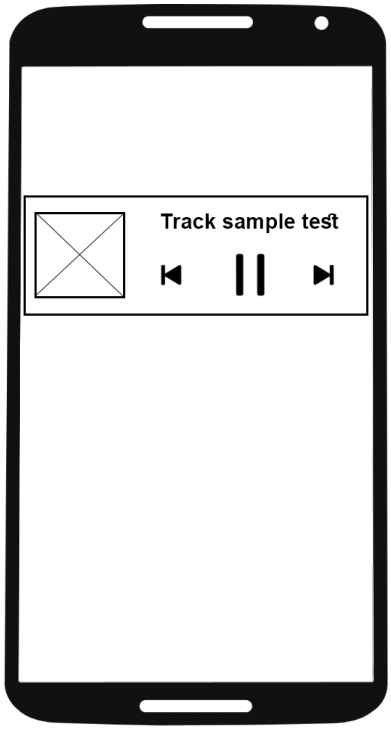


The player view with title,album art and seekbar to change time and favourite check box.

Tablet Screen



App Widget



Add as many screens as you need to portray your app's UI flow.

Key Considerations

How will your app handle data persistence?

1- Tracks list from Android MediaStore.
2- Make SQlite for listen count and favourites and get data from it using content provider.

Describe any edge or corner cases in the UX.

If user exit app and there is a track being played user can return to now playing using notification

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

1- Glide to handle the loading and caching of images.
2- ExoPlayer to play tracks.

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

1- Google analytics to make full analytics of screens and bugs.
2- FirebaseCloudMessaging for send push notifications to users

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

Write out the steps you will take to setup and/or configure this project. See previous implementation guides for an example.

1- Configure Firebase and Google analytics new Project
2- Prepare Code for getting All Tracks in the device

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for Track item layout on the list
- Build UI for now playing screen

Task 3: SQlite

- Prepare SQlite DB for store counts of listened tracks and favourite tracks
- Prepare Content provider for getting data from SQlite DB
- Prepare a loader to load music tracks into recycler view

Task 4: Functionality working

- Work with exoplayer to get Mozica player functionality work good

Task 5: Tablet UI

- Build UI for tablet

Task 6: App Widget

- Build Widget on home screen to play or pause currently playing track

Task 7: Fetch Track cover art

- Fetching track cover art from coverartarchive.org and make requests using AsyncTask.

Add as many tasks as you need to complete your app.

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**"