Udacity Android Developer Nanodegree

Project 7 - Capstone Design

GitHub Username: bennetimo

Chord Master

Description

Take your guitar playing to the next level!

Struggling with pauses in songs when you're stopping to look at your fingers, or having difficulty placing them down in time? Or putting them down quickly but in the wrong place and sounding...off?

Practicing the chords you find tricky repeatedly and regularly will make your playing smoother, sound better, and more enjoyable.

Chord Master helps you practice and track your progression with your chord changes. Fire up the app, practice a chord change, and input your score. The app will track your progress over time, help you work on your problem changes, and help you master that song you've been yearning to play.

What are you waiting for, get strumming with Chord Master!

Intended User

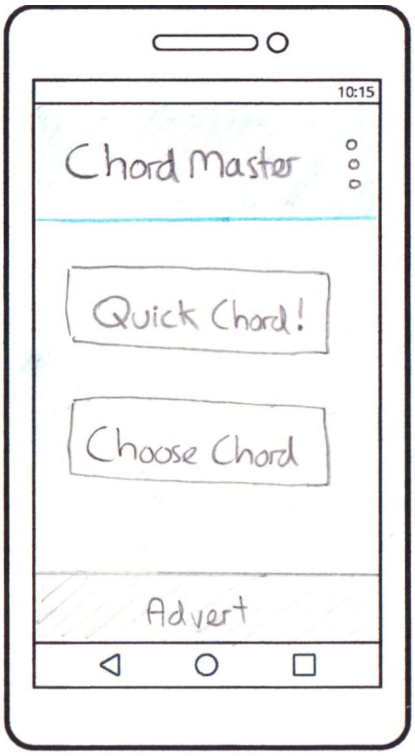
Beginner to intermediate guitar players

Features

- Shows countdown timer while the user is making as many chord changes as they can
- Saves data on the number of changes the user was able to make in one minute
- Choose specific chord or let the app choose one for you
- Tracks progress over time
- Widget for showing chord list and progress on the home screen

User Interface Mocks

Screen 1 - Welcome Screen



Initial screen the user sees when they load the app. Allows them to start their practicing session with a random chord change (‘Quick Chord’ button), or to choose a specific chord they want to practice (‘Choose Chord’ button).

Contains an advert at the bottom of the screen. Also, there is a hidden menu which displays an option to launch the settings screen (see below).

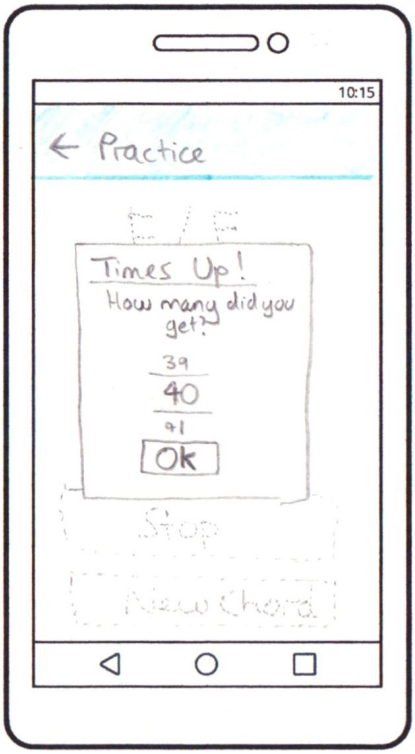
Screen 2 - Practice Screen

This screen is shown after choosing a chord, either via the ‘Quick Chord’ button or selecting it from the chord list. The main element on the screen is a timer which counts down 60 seconds, after a short lead in time.

While the timer is counting down, the user will use their guitar to strum the chord change shown. When the timer expires, a dialogue or similar will display so they can enter their score (how many changes they managed).

There are also buttons to stop the practice and go back to the main screen, or to change the chord for another.

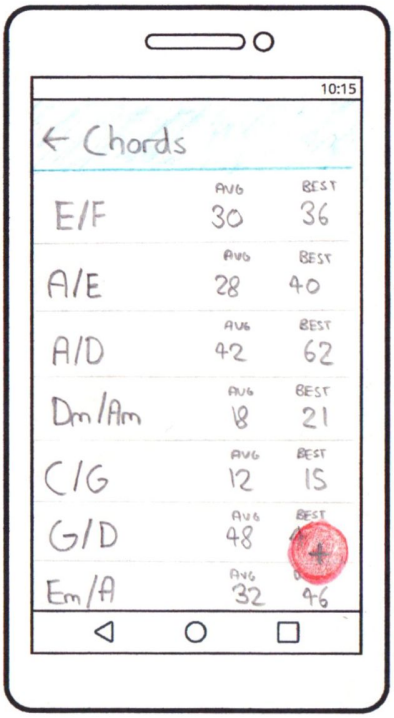
Screen 3 - Time's Up Dialogue



Displayed when the timer has finished and features a number picker or similar for the user to input how many chord changes they managed before the time ran out.

Clicking OK will persist the data for that change, and then show the user the Practice Screen again, but with a different chord selected to practice.

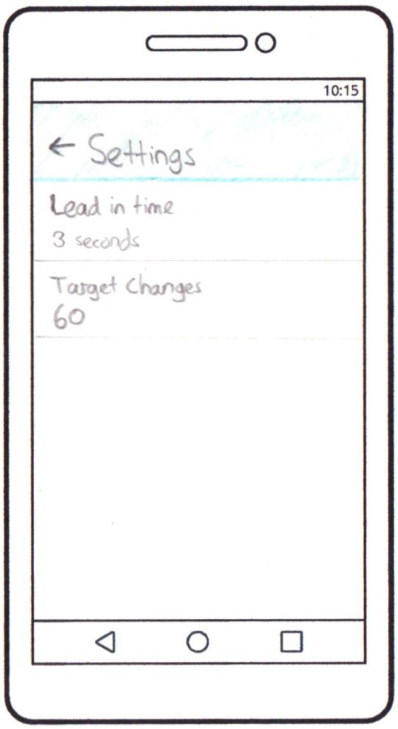
Screen 4 - Chord List



Displays the list of all the chord changes that the user wants to practice, and have added to the app. There is fab button to add a new chord change to track.

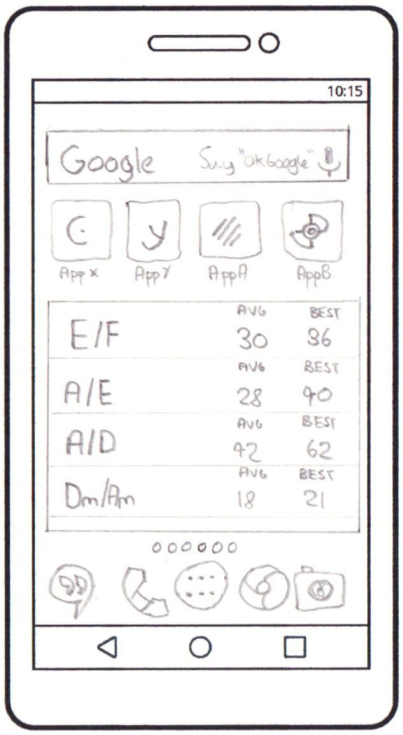
The display includes some info on how well they are progressing with that change, for example the average and best number of changes as shown in the mock up.

Screen 5 - Settings Screen



Allows the user to change some defaults that are used by the app, for example the amount of seconds to count them in on the timer.

Screen 6 - Widget



Allows the user to put their list of chord changes on their home screen, showing their progress.

The widget is shown here on the home screen.

Key Considerations

How will your app handle data persistence?

The app will use Firebase Database for storing the user's chord list and chord history. The settings will be stored locally on the device using a SQLite DB via a Content Provider.

Describe any corner cases in the UX.

If the user clicks the back button on the Practice screen they will be taken back to the Chord Screen or the Main Screen depending on whether they came from ‘Quick Chord’ or ‘Choose Chord’.

After the user has input a score in the Time’s Up Dialogue, the Practice Screen will be shown again but with a new chord change. In this way, the user can keep practicing multiple chords during a session until they decide they need a break and want to stop (or their fingers hurt!)

If the user has not added any chords to practice yet (e.g. on first installation), a basic set of chords will be loaded. In this way they are able to use the ‘Quick Chord’ feature right from the start. They can remove/customise the pre-loaded chords on the chord list screen.

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

- Butterknife - To simplify the binding of view elements and eliminate boilerplate code
- Schematic - To generate a content provider with less boilerplate code
- Google Support Library - For AppCompat components and material design elements
- Firebase Database - For persisting the users chord list and progress
- Firebase Analytics - For tracking metrics of how the app is used

Describe how you will implement Google Play Services.

The app will make use of some Firebase services:

Firebase Ads - To display a banner ad on the Welcome Screen of the app
Firebase Analytics - To track metrics on how users are interacting with the app, for example the average minutes per session spent practicing. This will provide valuable data to improve the app experience.
Firebase

Next Steps: Required Tasks

Task 1: Project Setup

- Create a new project in Android Studio
- Setup gradle build with required library dependencies
- Setup minimum and required SDK versions
- Add required permissions to Manifest
- Research and familiarise with Firebase APIs
- Create required accounts on Firebase

Task 2: Implement UI for Welcome, Practice Screens

- Build UI for Welcome Screen
- Build UI for Practice Screen
- Launch Practice Screen via Intent from ‘Quick Chord’ button on Welcome Screen
- Integrate with Firebase Ads to display a banner advert on the Welcome Screen

Task 3: Implement Timer

- Implement 60 second countdown timer for Practice Screen
- Build Time’s Up Dialogue and display after timer is finished
- Show Practice Screen after confirming score in Time’s Up Dialogue

Task 3: Persist Chord Score Data

- Persist the user's progress on their chord changes. Includes the time of practice, score reached, and the chord pair

Task 4: Implement Chord List Screen

- Build UI for Chord List screen
- Link ‘Choose Chord’ button from Welcome Screen to launch it via an Intent
- Retrieve persisted chord data to populate the chord list
- Allow the user to choose a chord from the list to practice that chord specifically
- Build Fab button so new chords to practice can be added

Task 6: Implement chord chooser logic

- Implement logic to decide the next chord to display to the user. Uses an AsyncTask and takes the user's chord list to select the next chord they should try
- Integrate the logic so it is used to choose the chord to play from the 'Quick Chord' button, and also to display the next chord after a practice has finished

Task 5: Implement Settings Screen

- Build UI for Settings Screen
- Launch Settings Screen from app bar menu on the Welcome Screen
- Persist settings choices to local SQLite DB via Content Provider
- Make use of the values in the settings in the relevant parts of the app they relate to

Task 6: Implement Widget

- Build UI for widget
- Implement widget to show the users chord list and progress

Task 7: Integrate Firebase Analytics

- Integrate Firebase Analytics into the app to gather interesting metrics from use, e.g. minutes spent practicing etc