GitHub Username: jusbielak

Words Guide

Description

App enables user to read lyrics of songs in original language and see its translations in other languages. Thanks to the Words Guide you will be able to check lyrics and understand songs that you like, even if you don’t know its language at all.

.

Intended User

The app will be useful for every music fan, who is curious about songs lyrics.

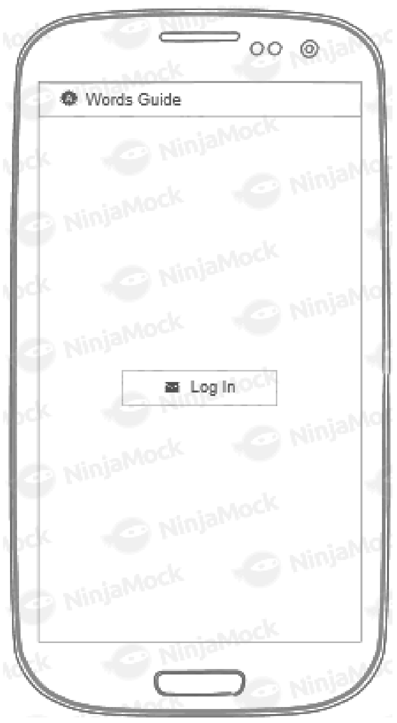
Features

List the main features of your app:

- Search song original lyrics
- Check song translation
- Explore top songs in various countries
- Save and display favorite songs

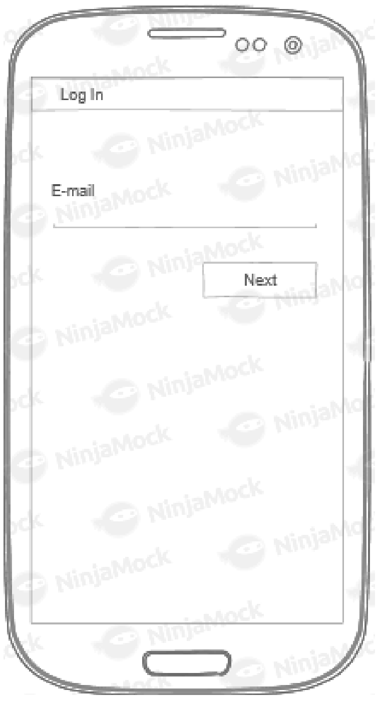
User Interface Mocks

Screen 1



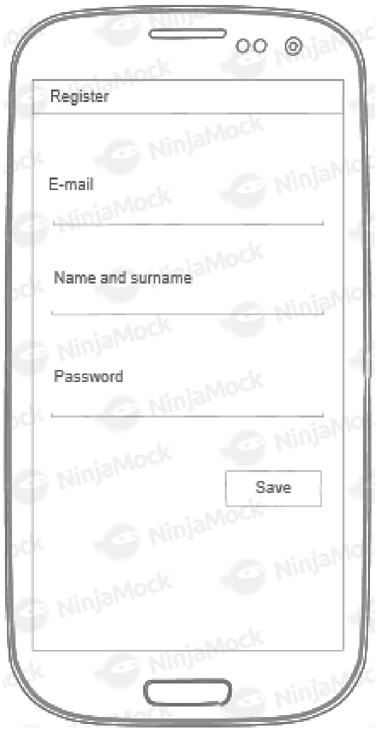
Application startup page, which enables user to select log in via e-mail address.

Screen 2



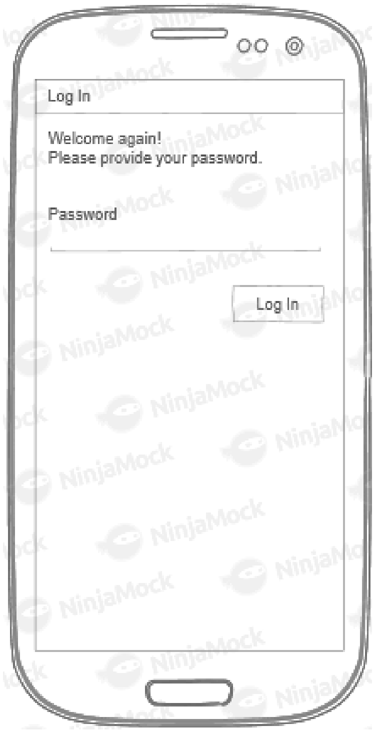
Next step of loging in - page with edit text for typing e-mail.

Screen 3



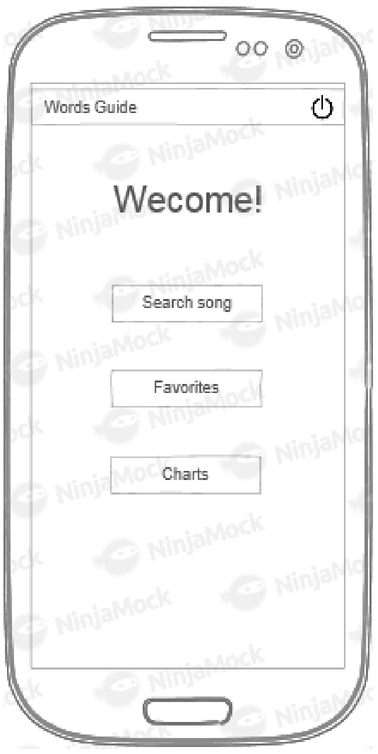
Next step of loging in - page that will be displayed to unregistered user.

Screen 4



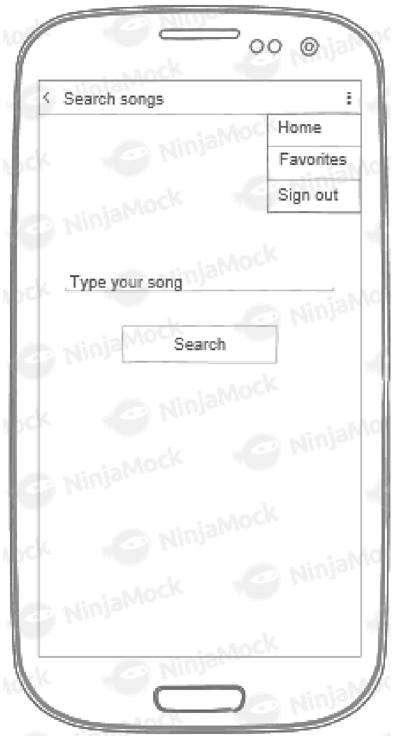
Next step of loging in - page that will be displayed to the registered user.

Screen 5



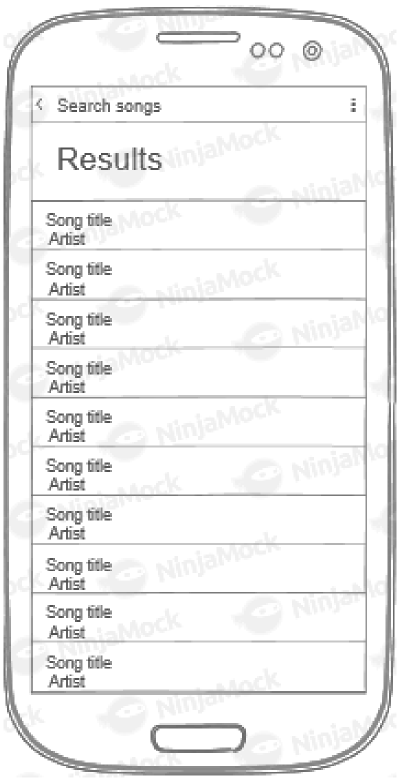
App home page screen.

Screen 6



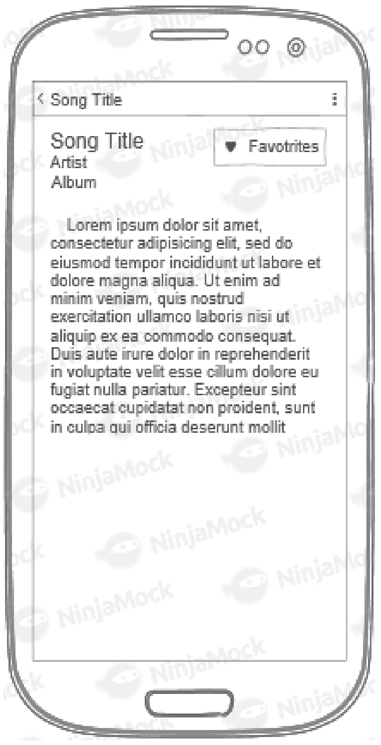
Songs searching – main page.

Screen 7



Page displaying results of searching.

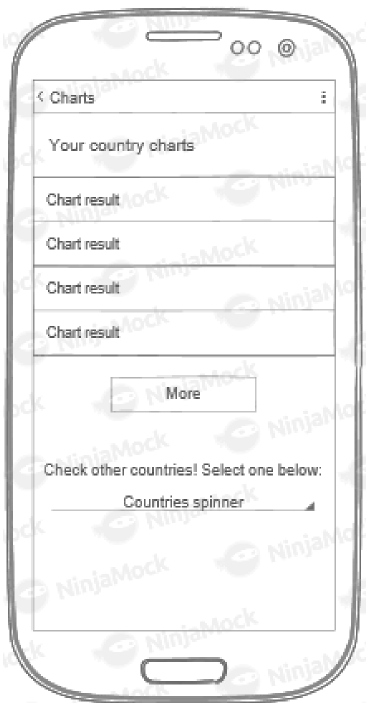
Screen 8



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit

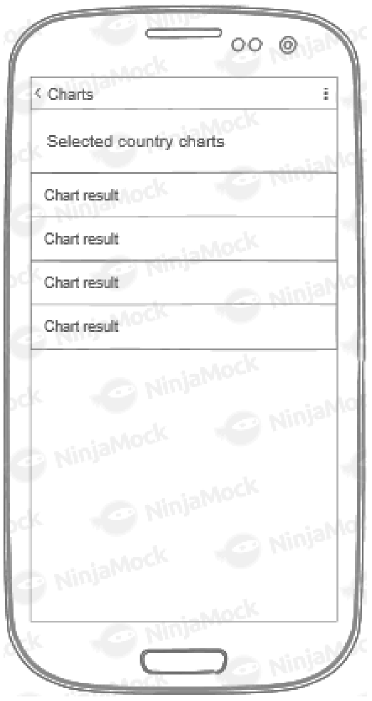
Page displaying song details.

Screen 9



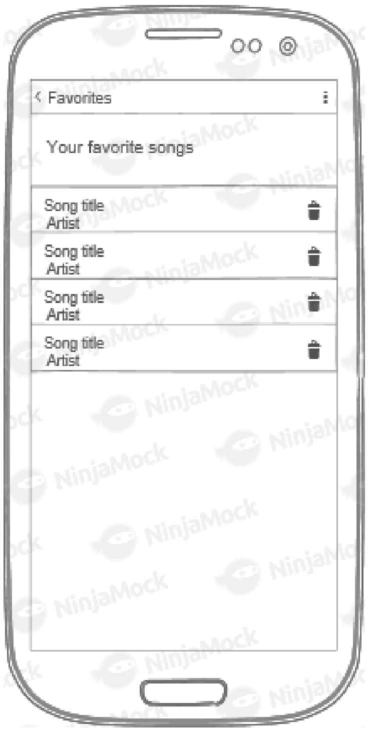
Charts – main page.

Screen 10



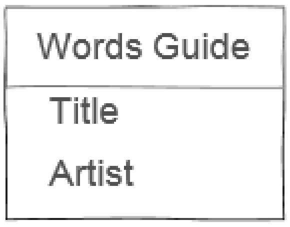
Charts – results details page.

Screen 11



Favorites – main page.

Screen 12



App widget – title and artist of recently displayed song.

Key Considerations

How will your app handle data persistence?

Data persistence will be handled by the Firebase Realtime Database.

Describe any edge or corner cases in the UX.

- Data may not be processed and displayed properly if the JSON file received in response may be malformed

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

- Butterknife - simplify binding views,
- Retrofit - handling API communication,
- Picasso - loading images,
- Firebase Realtime Database - saving user's data,
- Firebase Authentication - register and authenticate user,
- Location - find user's location,.

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

App will use Musixmatch API version 1.1 (https://api.musixmatch.com/ws/1.1/) to get required data.
Below operations will be used in the app:

- /chart.tracks.get - get top tracks for selected country
- /track.get - get track with given ID
- /track.search - search track
- /track.subtitle.get - get track subtitles

Google Location service will be used to get user's location and display songs charts for this country.

Next Steps: Required Tasks

Task 1: Project Setup

The Application is written solely in the Java Programming Language. All strings are kept strings in a strings.xml file, dimensions in a dimens.xml file, colors in colors.xml, styles in styles.xml. RTL layout switching on all layouts is enabled. Content descriptions for all images and buttons are provided.

1. Create a project in Android Studio 3.1.3 and use Gradle 4.4.
2. Provide required libraries:
    - Butterknife: com.jakewharton:butterknife:8.8.1,
    - Retrofit: com.squareup.retrofit2:retrofit:2.4.0,
    - Picasso: com.squareup.picasso:picasso:2.71828,
    - Firebase Realtime Database: com.google.firebase:firebase-core:16.0.1, com.google.firebase:firebase-database:16.0.1,
    - Firebase Authentication: 'com.firebaseui:firebase-ui-auth:4.1.0,
    - Google Location: com.google.android.gms:play-services-location:15.0.1.
3. Generate API key for Musixmatch and provide it to the app.
4. Create project in Firebase Console.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for HomeActivity
- Build UI for SearchActivity
- Build UI for SongActivity
- Build UI for FavoritesActivity

Task 3: Implement Firebase Authentication

- Add UI sign and AuthStateListener
- Setup sign in and sign out
- Handle sign in cancelation
- Handle sign out

Task 4: Implement Location

- Add ACCESS_COARSE_LOCATION permission
- Add Location Services client

Task 5: Create Model Objects

- Create Song complete model for handle API response
- Create Song DTO for persisting favorite songs in the database

Task 6: Create AsyncTask

- Create AsyncTask to perform on demand requests for searching songs, fetching lyrics and charts

Task 7: Create user’s favorites songs section

- Create layout for displaying user’s favorite song
- Add button for add song to favorites
- Add button for remove song from favorites

Task 8: Create application widget

- Create layout for displaying widget
- Display title and artist of recently displayed song