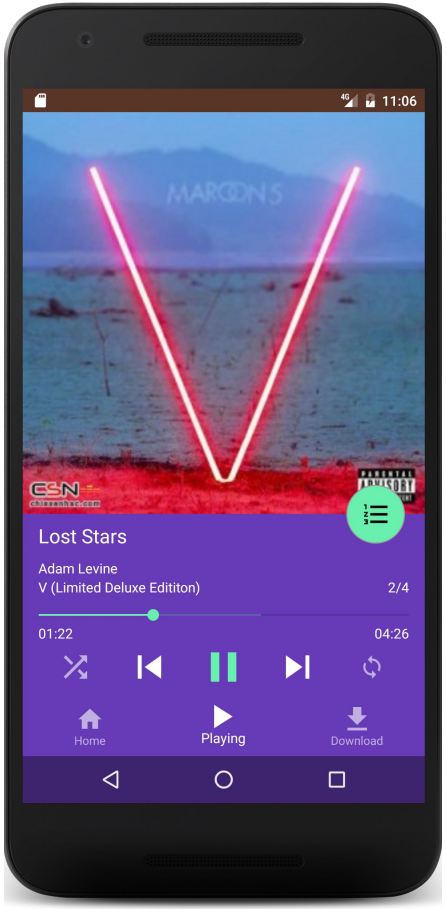


Online Music Player

Mobile Application Development

Group 5:

- Đỗ Đặng Ngọc Kha
- Nguyễn Đức Tùng
- Đặng Vũ Lâm

Content

1. Introduction
2. Architecture
3. Activities
4. Networking
5. Optimization
6. Demo
7. Conclusion

Introduction

- Online music player application, that connect with MP3Zing public music
- *What does the app do?*
  - Play music online and offline
  - Download music from the server
- *Why do we need it?*
  - Easily listen to music from MP3Zing
  - Lower the necessary for internal storage for Music content

Architecture

Loose MVC Architecture

- **Model**
  - SongItem
  - Playlist
  - SongAPI
  - RequestQueue
- **View**
  - XML fragments, views
- **Controller**
  - Activity - MainActivity
  - Fragments classes

Models

- **Playlist**
  - Contain list of SongItem objects
  - Describe a set of song that can be played in sequential order
- **SongItem**
  - Store a song's metadata and URL
- **SongAPI**
  - Unofficial API of MP3Zing
  - Can search songs by name
  - Return JSONObject with song’s information
- **RequestQueue**
  - A singleton Volley request queue

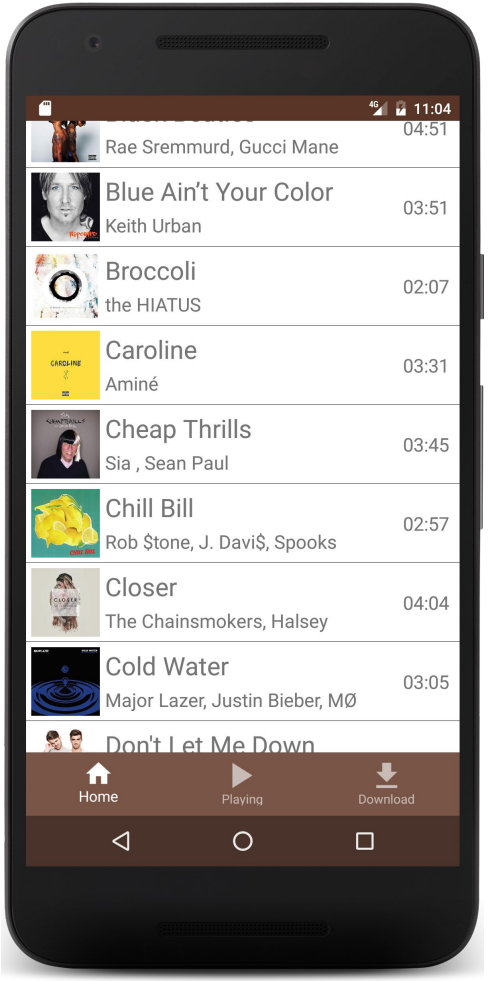
** Volley: a networking library make easier, faster network call*

Activities

- **MainActivity:**
  - Control component
  - Loads fragments to display in different tabs in the app.
  - Handle events

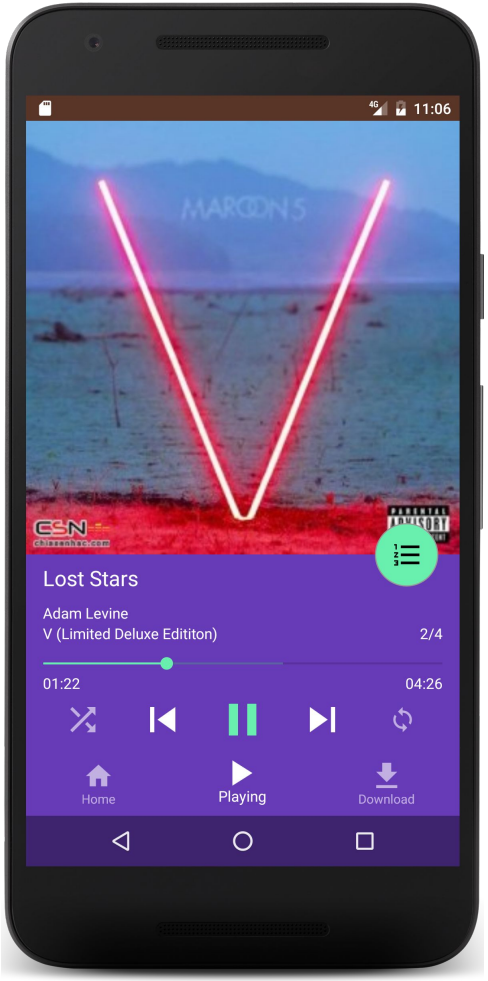
Fragments

- **Songs Fragment**
  - Show list of songs got from SongAPI
  - Hold to download



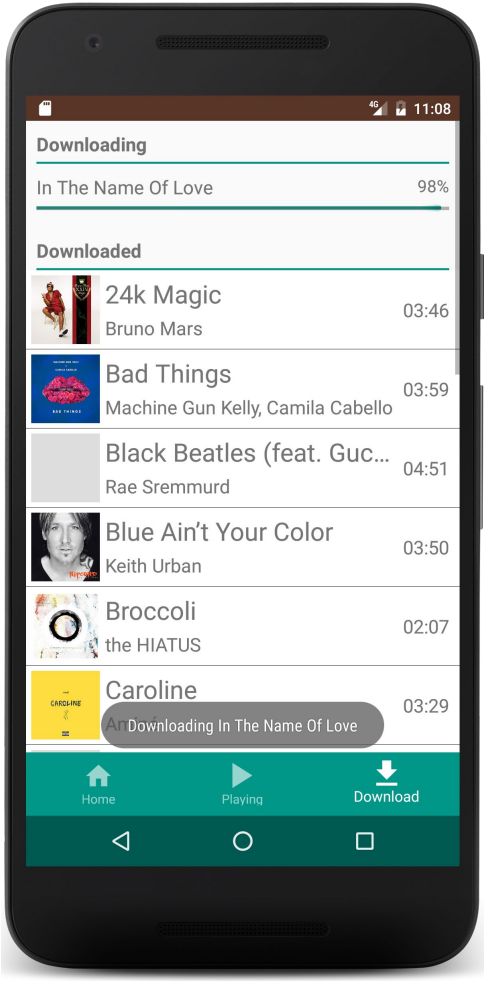
Fragments

- **Playing Fragment**
  - Show UI to control music player



Fragments

- **Download Fragment**
  - Show downloading and downloaded songs
  - Downloaded songs can be clicked to play



Network

We connect to MP3Zing server to get songs information

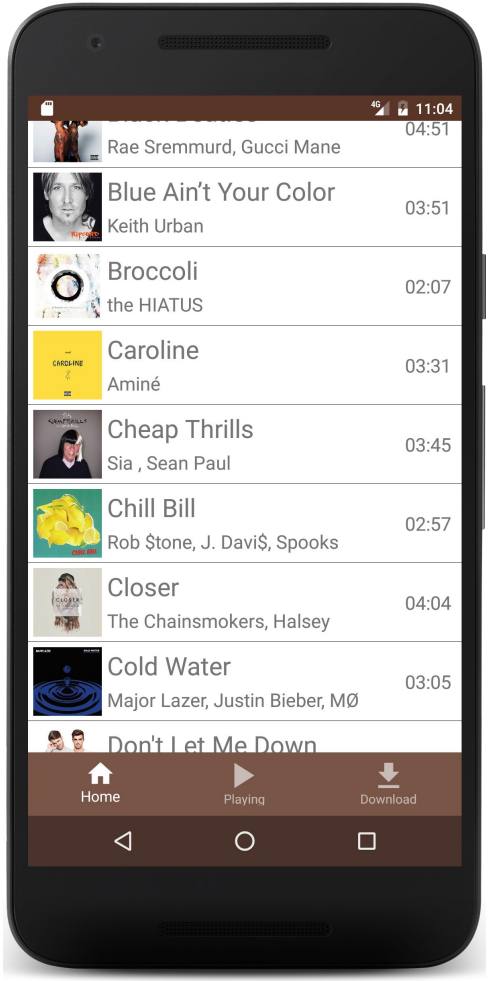
Our unofficial MP3Zing API (2 steps):

1. Get Song ID from http://j.ginggong.com/jSearch.aspx by song name
2. Get song info in JSON from http://api.mp3.zing.vn/ by song ID

- Async access (Volley queue)

Optimization

- **Songs Fragment**
  - In the beginning, the whole song list are loaded and update to the UI at once
  - Now, each song item are added to Volley RequestQueue, and update to the UI consecutively
  - The same thing applied for the song artworks



Optimization

- **UI Overdraw**
  - Mostly blue and green

DEMO

Conclusion

- *What was done*
  - Architecture design
  - Layout of the app
  - Playback with offline music
  - Download music from server

- *What has not been done*
  - Stream music online
  - Show song list in artist, albums, …
  - Searching function

- *Possible future development*
  - Cache for song list
  - Play music when sleep