# Musical UX Smart Enhancer

**Students:**

Covasa Alexandru-Iulian

Covasa Corina-Gabriela

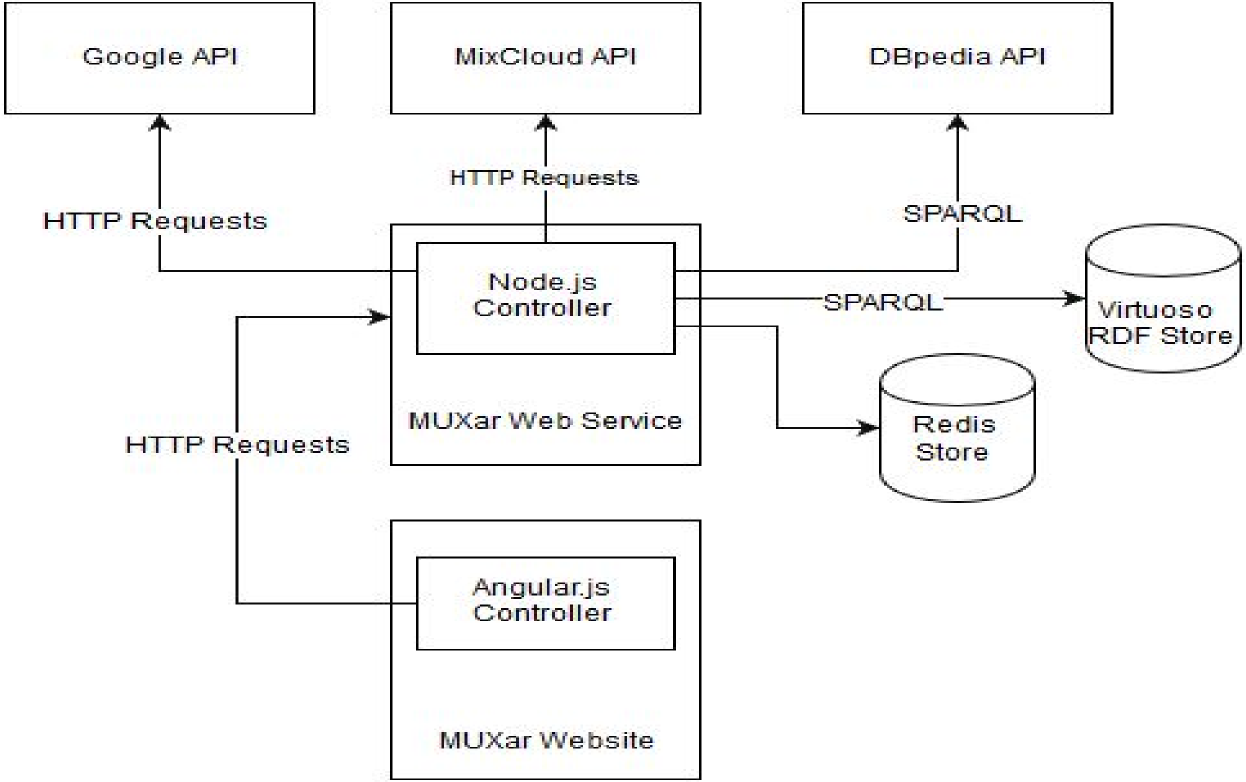
Plesciuc Sebastian-Laurentiu

**Coordinator**: Prof. Dr. Sabin-Corneliu Buraga

# Introduction

- MUXar is an application that brings new experience for the users, facilitating not only the searching and listening to new songs, but also giving smart recommendations based on his preferences.
- And this will be possible using the power of semantic web.

# Application Architecture



# Application Architecture

The project is based on a **client-server** architecture:

- server – it will be implemented in **Node.js** and is the main controller of the application linking the front-end with external services.
- client – consists in an **AngularJS** controller that redirects each user's request to the server

Other services used are:

- **Redis store –** component that stores session data for the user
- **Virtuoso store –** holds information in **RDF** format
- **External services –** such as **DBPedia, MixCloud, Google API**

# Main concepts

As our application is based on **semantic web** and our project is about music, the following entities with properties are suitable to be used:

- **artist**(id, name, age, placeOfBirth)
- **album** (id, title, artist, songs)
- **song** (id, name, artist, duration, genre, album, composer)
- **playlist**(id, song)
- **history**(id, song)

# REST API description

Our REST API is described using **Swagger** tool. A capture with details about the entities(resources)/URL and method can be seen below:

| Resource | URL | Method |
|---|---|---|
| List of artists | /artists | GET |
| Artist | /artists/id | GET |
| List of songs | /songs | GET |
| Song | /songs/id | GET |
| List of playlists | /playlists | GET |
| Playlist | /playlists/id | GET |
| Playlist | /playlists/ | POST |
| Playlist | /playlists/id | DELETE |
| Playlist and song | /playlists/id/addSong | PUT |
| Playlist and song | /playlists/id/deleteSong | PUT |
| History | /history | GET |

# Use cases

1. The user searches for a specific artist/song
2. The user adds/deletes a new song to/from a playlist
3. If user listened to a specific genre a recommendation will be made with songs of similar genre
4. If user has participated to an event, will receive recommendations similar to that event
5. User would like to know what artists have collaborated with ArtistX.

# Questions?