SENG 2021 | Team Melli
Deliverable 2

Group Members:

Afia Hoque
Arsham Emad
Kasra Mahabadi
Varun Kashyap
Yuechen Gong

Part 1 : Software Architecture

1. What external data sources will your system be accessing?

   As the theme of our project is Music, the main external data source our system will be accessing is Wikipedia through the 'MediaWiki' API. This API is a RESTful (Representational state transfer) Web Service that allows users to perform certain actions - like searching, which will be most relevant to our system.

   Furthermore, as users will be able to listen to music, the system will need to access the youtube database through the 'Youtube Data API'. Similar to the MediaWiki API, the Youtube Data API would allow the system to search the youtube database. The result set identifies matching video, channel, and playlist resources.

2. *Software components: the selected Web stack showing major software components that comprise your solution. These will include both components that need to be developed and third-party components (e.g. web browser).*

   The project will be built on 'dotCMS'a Content Management System (CMS). 'dotCMS' is an open source 'HybridCMS', offering the functionality of a headless and traditional content management system.

   The main disadvantages of building a website on dotCMS are:

   Security concerns; as it's an open source platform, the site is more vulnerable to cyber attacks as parts of the code have been written by someone else. In saying this however, 'Tune Talks' will not be handling sensitive information from our users which is why we have chosen to move forward with wordpress irrespective of this disadvantage.

Site speed; Due to the excess of generic code and plugins associated with using a CMS platform such as dotCMS, hence it could potentially increase loading time of the site resulting in a worse user experience. However, as most of the site functionalities will be run through the API's, site speed will not be influenced to the point of reducing user experience.

Frontend Technology Stack - Angular

Our project will use Angular for it's frontend stack. Angular supports MediaWiki API and Youtube Data API integrations, which is the external data sources our project will be relying on.

Developer Stack

Each developer has their own personal productivity stack to ensure cohesion when creating the project. The editor being used is Visual Studio Code, with version control being managed through branching on a Github repository. Deliverable management is occurring through 'Trello'.

Backend Technology Stack - Node.js

The backend of our system will use Node.js - which uses Javascript as its scripting language.

3. Relating choices to components: decide which language should be used for which component of the software architecture. This will be largely determined by the Web stack but at the same time you can make variations.

   The technology components include the CMS platform 'dotCMS' is based on 'Leading Java Technology', hence the Frontend will be coded using Javascript.

   Additionally, as we have chosen to use Angular, it requires Java and HTML when creating websites. Customs tages created using HTML are needed when making Requests through dotCMS for the website.

   We will also make use of JavaScript as it is the main language used in our backend construction.

4. The choice of a platform: decide on machine or machines requirements (Linux, Windows etc.) for the final system.

   For the platform that will be required to run our final system we decided to go with Linux. Our testing and practice runs will be on a linux environment therefore we saw that it would be the best fit. However, our system is going to be designed to be accessible from any machine as long as the machine has access to the web browser Google Chrome. This means that although we will mainly use Linux, machines with operating systems such as Windows and Mac OS will still be fine to use along with Google Chrome.

5. Make a summary of the key benefits/achievements of your architectural choices.

   For each of the decisions that we made about the different software architectural components of our project, we weighed the pros and cons and decided that the above decisions were the best for us.

   Firstly, our choice of the Youtube Data API was due to its extensive use in the music area. The advantages of it include being able to search by specific fields such as artists, publication dates, genres and location. Users can search not only for specific videos but also playlists and artist channels, and search by language. The reason the Youtube Data API was chosen over the Spotify or Apple music API is because our system will be a stand-alone platform rather than an add-on to Spotify or Apple Music. Furthermore, unlike the Spotify and Apple Music API, Youtube Data API and MediaWiki offer free solutions for the project.

   We also chose to use DotCMS to build our system. This decision was justified due to its many benefits including high flexibility and ability to be customised. We are in need of customisation abilities as we need to change a lot of things to fit our specific music themed functionalities such as music recommendations and music forum posts. Another advantage of DotCMS is that it is fairly easy and smooth to use with a simple and easy to learn interface. It also allows you to make tweaks and changes to your system without a hassle which is something that we anticipate we will need going forward. We also considered some disadvantages such as the bugs that are often encountered on this platform, and it has less built in frameworks in comparison to other options, however we still decided that overall it was the best choice for us.

We considered the alternative of WordPress, but we found that it was limiting in its scope as DotCMS has more flexibility in terms of development. As DotCMS is a ‘hybridCMS’ it gives us the flexibility of a ‘headless CMS’ (A back-end only CMS with the primary function of being a content repository rather than a front-end or presentation layer).

When choosing which Frontend Technology Stack to use for our project, three frameworks were considered: Next.js; Nuxt.js; and Angular.

Nuxt.js is extremely light weight, and most commonly used for single page applications. As ‘Tunetalks’ will have multiple pages and functionalities - using multiple API’s and accessing large music databases, Nuxt.js did not meet our requirements, hence was not used. Next.js had similar issues, it's a minimalist framework for server rendered applications. Next.js is better suited for sites that require very good SEO such as ECommerce sites.

As ‘Tunetalks’ is a large app, Angular is the best option due to its stability. Furthermore, taking into consideration development timelines, Angular is considered to be quick to develop due to greater community support as it’s been more widely adopted by developers.

For our backend, we will be using Node.js. We had a look at several options such as Python and PHP, but after doing research and reading about the best technologies to use to build a backend, we finalised our choice of node.js. We saw that it suits our project as it has a large number of free tools, it has high efficiency and the code is easy to reuse and share. Our developers also had some knowledge and experience with it which made our decision easier.

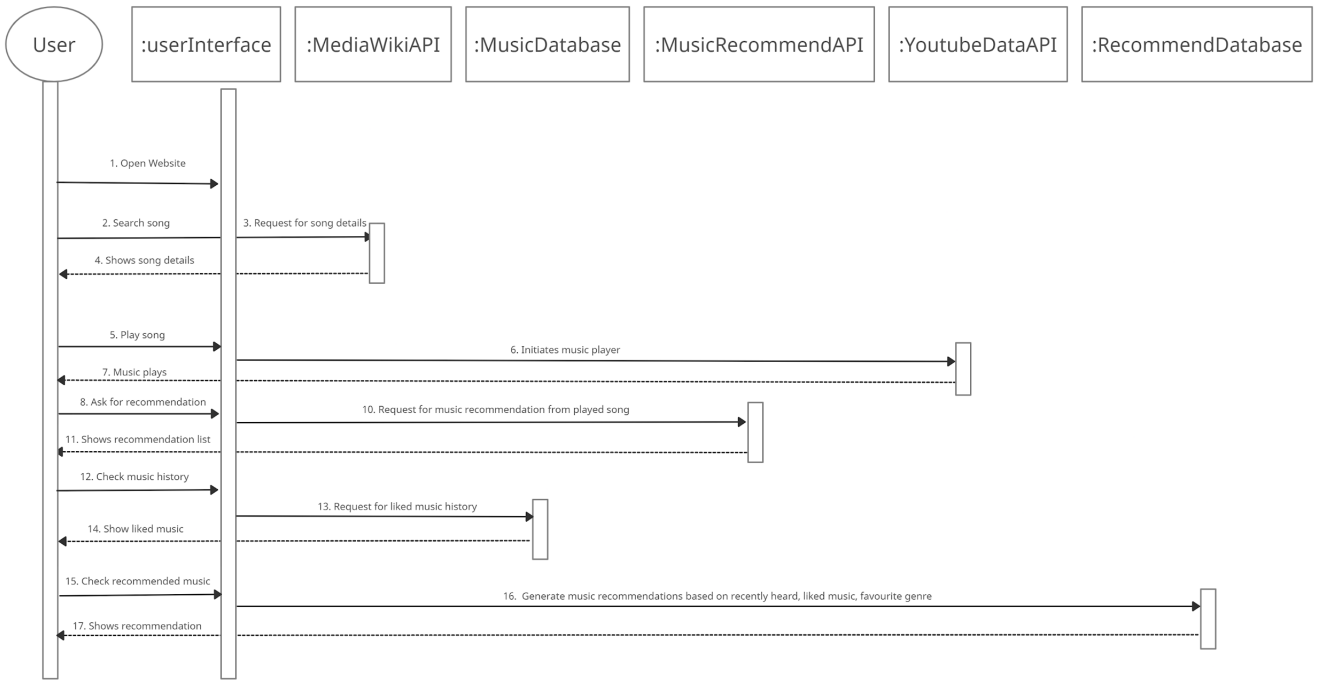
Part 2: Initial Software Design

The main user stories ‘Tune Talk’ will focus on are the recommendation of new songs from similar genres and the ability to share posts about music status on popular social media. For the diagrams, please refer to the google drive links to see the originals.

Feature: Hear new music recommendation
As a user of ‘Tune Talk’
So that I can find new songs to listen to
I want to hear new music recommendation

Scenario: User wants to find music recommendation from playing/selected songs
GIVEN I am listening to a song on ‘Tune Talk’
WHEN I search for music
THEN a results will show the song details
WHEN I play the song
THEN the song will begin to play and more recommendations appear
WHEN I click ‘Find more music’ on the bottom tab
THEN a list of 10 new songs by similar artists is presented
WHEN I check my liked history
THEN a tab opens with my liked song selections
WHEN I view the ‘Music Recommendation’ page
THEN I will find lists of music recommendations based on liked music, recently played, favourite genres

https://drive.google.com/file/d/1vpgxyTUkop6QwsF7ltJq5nZ4FQOUXKo6/view?usp=drivesdk



Feature: Make posts on the forum
As a listener who enjoys analysing music
So that I can talk with other people about music
I want to make posts on the forum for everyone to see

Scenario: Makes a post on the forum
GIVEN I am on the ‘Tune Talk’ dashboard
WHEN I click on “Forums” tab
THEN I will be redirected to the “Forums” page
WHEN I open up a “Popular post”
THEN I should be taken to a discussion page which has title, author’s post and comments
WHEN I click on “Write comments”
THEN I should see a text input field should open up
WHEN I write my comments and click “Post”
THEN I should see a refreshed page which shows my post in the discussion page
WHEN I give a “like” on another user’s comment
THEN I should see the like numbers updates and shown as “liked” by me
WHEN I click on “New Post” symbol
THEN a text input field should open up with title field and post field and music link
WHEN I fill the informations and click on “New Post” symbol

THEN I should see the page refreshed with my post on top

https://drive.google.com/file/d/1AYg3hHEhyl0zNCJFSdnwyDvUe8NUwalf/view?usp=drivesdk