# CS 308 - Term Project MVP

This project aims to develop an online system that can collect liked-song information about its users from various different sources-platforms, present various different kinds of analysis about the musical flavor that people prefer and perform different kinds of recommendations based on collected information.

Under no circumstances is it expected from you to perform music streaming. Doing so will not improve your grade.

The MVP functionality that the developed platform is expected to realize can be listed as follows. Please keep in mind that this functionality will grant you 60/100 over the term project and the rest will be determined based on a curve-grading on the additional work that project teams have realized. Possible extension points are also presented mostly as the last sub-item in each main item.

For some of the functionality you should be using an external musical info API such as https://www.theaudiodb.com/ For your project's demonstration purposes please use the rate limited free versions. Any functionality that requires hundreds of external API calls should not be required for your system.

- **Data format:** The basic data format that the platform is expected to collect is the song information that was published as a part of an album, single or via other medium. The song information is going to present at least the name of the track, the performer(s), the album or media that it has been listed in and the rating from the user.

  On top of those there may be challenges when designing the corresponding data format such as:

  - Some tracks may contain multiple performers. Project team should find a way to handle the multiple composer information properly since keeping it as plain text would have its own side effects during the analysis phase.

  - The same song may be included multiple times in multiple albums, also its different edits may have been published. These choices should be properly handled by the project team to unify the different versions of the same song as much as possible during the analysis phase.

  - A song may have various different properties on top of the basic ones such as its length, its tempo, its genre (which may be more than one), its mood, being live/studio/radio recording, the instruments in the song, the number of times song has been listened to by the user, its release year, the time when the song is added to user database, etc. The properties in this item are not included in the MVP but the more project team handles the more points it is going to get.

- Data collection: Data collection can be performed over various different mediums of project team choice. However, the following functionality should be provided
    - Manual user input via web/mobile application: A user can input manually a single song by a neat UI presented by the project application. The information to be collected is explained in the previous item.

    - Batch input via text file/csv/json/etc. transfer: A user can input multiple song information at once by uploading a file format of project team's choice. This format should be consistent with the format used for export functionality explained later.

    - Transfer via a self hosted or cloud database service of choice: A user can connect the project to a cloud database platform and/or a self hosted database of its choice and import all the song information in the database to the system. The schema used in the external database can be assumed to follow the schema provided in the previous item.

    - Ability to rate non-rated songs/albums/performers: A user may add many songs to the system that has not been rated by herself/himself. System should provide a neat interface that allows the user to rapidly rate the songs/albums/performers that have not been rated yet. It should be noted that there may always be non-rated songs/albums/performers in the song database which should be properly handled by the project team. Another possibility is to be able to change rating in time that will be retrospectively recorded so that the user can see how her/his liking of the songs/albums/performers changed over time.

    - Ability to remove a song/album/performer from the system. The removal should be done in a cascading way when required.

    - Song information can also be transferred from other systems such as Spotify, last.fm, Allmusic, etc. not only for liked songs but for missing song information and properties as well. When done so, incompatibilities in the data format should be handled by the project team properly. <u>The properties in this item are not included in the MVP</u> but the more project team handles the more points it is going to get.

- Analysis of musical choices: Over the collected song information the system should perform a series of analyses and present them via different mediums such as statistics, tables, charts, etc. via a dashboard. The following functionality should be provided:

    - Users should be able to see statistical information about their likings of songs/albums/performers and additional data that the system holds. This statistical information should be filtered by a date constraint. For example the user should be able to list his favorite 10 albums from the 90s era. Another example may be the user should be able to list his favorite 10 songs that were added during the previous six months.

    - Users should be able to display tables and charts according to some predetermined criteria offered by the system. An example might be viewing a table of the last 10 months' average ratings on a selected group of artists. An example chart may be the number of songs added by the user that belong to a selected group of performers. Another interesting chart may be the average rating the user provided during the last month on a daily basis as a line chart.

    - Project team may even further let the user customize the analysis medium such as letting the user select and build pivot tables, interactively editing the charts and being able to select the chart type etc. The properties in this item are not included in the MVP but the more project team handles the more points it is going to get.

- Recommendations: The system should provide recommendations based on the collected user information. Following functionality should be provided.

    - System should provide music recommendations based on different kinds of analysis it performs. A very obvious selection is to provide recommendations based on user ratings. This recommendation can be done in song/album/performer granularity.

    - The system should also make recommendations based on the temporal properties. One option would be to recommend songs from highly rated albums/performers that don't have any activity recently (example: "you haven't added a song by <<some performer>> lately that you have previously rated high; you may want to listen to <<another song>> of this performer" ). Another option is to use genres (if collected) and recommend similar genres that the user is adding songs from in the past few days.

    - The system can also make recommendations over the friendship activity. System should recommend some songs/albums/performers based on the activity of the added friend. When recommending, the system should explain the source friend identity (if allowed) of the recommendation as well.

    - It may be possible to transfer recommendation information from other sources such as spotify, etc. Additionally, recommendations may also be done over the overall activity of the users of the system based on some grouping such as geolocation, genre-listeners, performer-fans, etc. The properties in this item are not included in the MVP but the more project team handles the more points it is going to get.

- Additional features based on user activity should be as follows:
    - Authentication: System is going to accept straightforward password based authentication. Optionally, the project team may also support third party authenticators such as google.

    - Friends and friendship: It should be possible to add friends that are using the system. It should be possible to limit the amount of activity that would be included in the recommendation analysis, per friend basis. For instance, a typical scenario can give the capability of a user to not allow its rating information to be used in the recommendations for a specific friend.
        - (Optional, not in MVP) Friend group analysis may also be a good idea. Users may be allowed to form friend groups that would allow them to perform analysis based on common likings.
    - Result sharing: It should be possible to share the analysis results explained in the previous items as a post in social media. The choice of the social media platform is left to the project team.
    - Data exporting: It should be possible to export the current database of song ratings kept by the user as a single file followed by a series of filtering operations. For instance a user may export as a file all the ratings to songs by a single performer.
    - (Optional, not in MVP) App-Integrations: Users may connect their profiles to third party applications such as Spotify, CDDB, allmusic, etc. to collect information. Additionally library integrations can also be done as well to perform operations such as recommendation and analysis. All of those integrations are more than welcome as long as they are properly documented.

- Development artifacts: Your code repository should contain a refined level of information about the key components of the system. Your github repository should have a proper Readme.md that would point the inspector to the related artifacts derived during the system development. Examples are: your backlog in Gherkin format, diagrams for the key aspects of your system, for instance your data schema as an E-R diagram, a sequence diagram for a third-party integration, analysis pipeline for the recommendation engine, information on how to deploy/run the system.

Good luck! Have fun!

13.10.2023

CS 308 Instructor team