Project report
Web Engineering 2019
Group 14

E. Waterink (s3417611), T. de Vries (s2367610), I. Oralin (s4171446)

October 2019
v2.0

Introduction

This document contains information regarding the architectural choices of the project.

Technology Stack

For the backend we was decided to use programming language Python as two members of the team have experience with it, and the third member can learn it fast and easily due to its simplicity. As a framework we was decided to use Flask as it is easy to use and it is suitable for a small projects like this one.

As a database management system we decided to use MySQL. The reason we chose this over a no-SQL database is due to the (relatively) small amount of data we have to deal with (it is in the thousands). In other words, we are able to get very fast query responses.

We also made use of the 3^{rd} party API QuickChart. With this we can easily make graphs, and in our case histograms. We create a URL containing a JSON/JavaScript object that includes all data and display options and get the histogram we want to display.

Database Design

The CSV file with songs has many duplicate information about the same artist for all songs he made. So we was decided to split data into three tables: Artist, Song, Release (the last one does not contain any useful data for current API but it was kept just in case). To make things easier, some field names were changed, e.g. 'terms' to 'genre'.

Songs contain a foreign key referencing Artist (id of the Artist) and Release (id of the Relase).

Architecture of the project

The Flask application plays the role of a server which receives HTTP requests from the API user. After the request is received, the server forms an SQL query and sends it to the MySQL DBMS which, in turn, responses with needed data. Thereafter, the server encodes the received data in JSON or CSV format and sends it to the user in the response.