

# Coursework 1

---

Web-app: Music@Napier

**SET09103 - Advanced Web Technologies**

**Module Leader: Mr. Simon Wells**

**01.11.2015**



Figure 1: Logo

**Student-ID 40219221**

**Exchange Student**

## Content

1. Introduction .....	1
2. Description of the web-app .....	2
3. Design section .....	3
4. Enhancements.....	5
5. Critical evaluation .....	6
6. Personal evaluation.....	7
7. References .....	9

## 1. Introduction

You all know the following scene: you listen to the radio and there was playing that song with which you fell into love. But then the radio host did not tell the details of this one song, only some gossip and tittle-tattle of the group. Now where can you have a look and research how this song is called? The result for this disaster is the music online catalogue “Music@Napier”. This is exactly the topic of the first web-app, which has to be created and developed for the first coursework.

So first of all this coursework shows a description of the web-app. Then, after explaining how this web-app is architected, there are some enhancements mentioned. Finally the last two categories deal with a critical and personal evaluation.

## 2. Description of the web-app

This developed web-app is an online catalogue, where the user can search and find songs. By a database it is possible to search or filter for genres, titles and artists.

But now step by step. Before the user can make a use in the proper sense, he has to log in first. In case that this web-app is a basic application, which is made by a student, who is not a web developer or web designer, there is only one account deposited. The login credentials are “admin” as username and “default” as password.

Then after entering the web-app with the username and the suitable password, there will open a page, which contains a table, where all included songs of the database were listed of course only when there are songs in the database. Now there are four possibilities. First opportunity is to filter by artists, second possibility is to filter by genres, the third one is to search for songs in the involved database and the last opportunity is to log out. In case of using one of the two filter-functions the user gets suitable tables of one artist or one genre. If the user uses the search-function he gets a table back, where the searched and entered word fits to the title, artist or genre of the data from the database. On the right side there is the possibility to click at the overview item. This function leads the user back to the listed songs. After using these functions the user should use the logout button. This logout-function leads the user back to the login-page.

### 3. Design section

After developing my first web-app, which contains only the basics of the workbook, a CSS-file and some HTML-files, I noticed after rereading the marking criteria that this is not enough to reach a good mark. Consequently I changed my web architecture and added a database. The topic of the web-app survived. How this crazy idea came into my mind? That is quite easy because Mr. Simon Wells, our module leader, told us in a lecture if anyone want to add a database to his web-app it is possible. So adding a database was my challenge. Of course I found a really helpful tutorial on the Flask website (Ronacher, 2013). Consequently the database, SQLite3, is the data basis. Therefore the developed web-app accesses to this database and represents the content of this database.

So the table below, Table 1, represents an overview of the URL history.

/	/login
	/logout
	/search
	/artist
	/genre

Table 1: URL history

Then the corresponding file-structure is shown by the following table, Table 2.

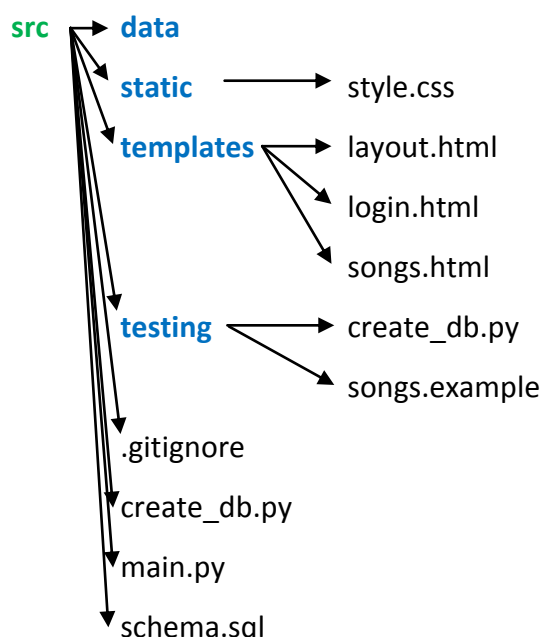


Table 2: file structure

Finally there are some comments and explanations to the mentioned files above.

The “src” directory contains the whole source code of the web-app.

The “data” directory is the place where the database is going to be located. A testing database with data testing directory can be created with the create\_db.py script.

“Sqlite” is the used database engine. “schema.sql” is a schema, which consists of a single table called songs. Each row in this table has an id, a title, an artist and a genre. The id is an automatically incrementing integer and a primary key, the other three are strings.

Afterwards some information about the “static” directory. This file contains all static data, for example CSS, that have to be stored in the file system.

The “templates” folder contents HTML codes, which describe the basic layout of the page. Here Python uses the external, already installed templates engine Jinja2.

Subsequent there is showed the testing folder, which involves all the example data to show that this web-app works. It is necessary because there has to be created an example-database before using the web-app.

„gitignore” specifies intentionally untracked files to ignore. In this case git should ignore the database.

The “main.py” is written in Python and uses the Flask microframework and contains all the routes of the web-app and their functionality.

## 4. Enhancements

After visiting this web-app the first time, everyone knows that there are many possibilities to expand. The following paragraphs show some nice and suitable enhancements.

The first improvement is the database. Whereas this database is a small and own developed one, which should show the functionality of this web-app, it would be nicer to connect the online catalogue with a bigger one. Then the user has a lot more songs to search and find.

After that it will be awesome if the existing login is for Napier students, which can see after login in the last searched songs and some other and new suitable suggestions of songs. First step to achieve this wish is to add a user database of the Napier University students. Then it would be nice to be able to create a playlist, where songs can be added and deleted.

In addition it is possible to found a place where students get music from other students by sharing their music. So every student can upload his data. Finally this web-app represents a data exchange application.

Furthermore the functionality can get improved in the direction of involved data. Because of the modern style of technology it would be lovely to have one web-app for songs inclusive song texts, and movies and books. Consequently this online catalogue does not only show and contain songs it will connect songs with suitable movies and books. So for example the web-app user searched a sad song. Then this web-app will suggest him suitable, sad and dramatic movies and books. In case that this song is played in a movie this film will be suggested, too.

Finally there are some improvements for the searching. As noticed this web-app can only find the whole searched word. So for example if you have a look at monkeys this function only find songs, which are called monkeys. Data which for example are called monkeybussiness are not detectable respectively were not listed.

## 5. Critical evaluation

- It is only possible to login as admin. There are no further accounts.
- The search-function only shows data, which title, artist or genre matched exactly to the searched term. So there should be paid attention to the case sensitively.
- The copyright of the data has to receive attention, too.

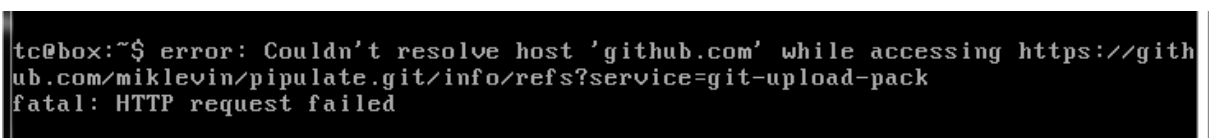


## 6. Personal evaluation

Every week the module leader uploaded a new chapter of the workbook, which is mostly suitable to the delivered lecture. These workbook contents the basic and important objects, which are necessary to develop a functional basic web-app. Consequently the students learned step by step how to create a web-app.

So after installing levinux and the necessary programmes like VIM, a command-line text editor, SSH, means secure shell, which is a tool to log in to a server, PuTTY, needed for the log in from a windows computer to the server, Git, a source control system (GitHowTo, 2015), Python, a very using programming language and flask, a Python based microframework for developing websites. Of course there were many tutorials necessary, in addition to the practice of the workbook, to handle with all these new programmes, even if the student has not attended the basic web technologies module.

Furthermore there arose some problems with the Git at the laptop. After writing the source code of the workbook it was not possible to push it on the GitHub repository. There were displayed an error comment, which is watchable in the figure, Figure 2, below. After asking the tutor and trying to find a solution it did not work either. As the result it was inevitable to do the practice at the campus. But because of an upload there were some problems at the computers in the Merchiston campus as well. Consequently the time flew and it was important to find a solution. One possibility was to go to another campus like the Sighthill campus, where the in question programme runs.



```
tc@box:~$ error: Couldn't resolve host 'github.com' while accessing https://github.com/miklewin/pipulate.git/info/refs?service=git-upload-pack
fatal: HTTP request failed
```

Figure 2: problematic Git comment

But these problems respectively challenges were not the only ones, there was another one, which is based on the images and pictures. To upload pictures and images to the static file in the GitHub repository it was used the “curl-command”. After knowing that the cloned URL, which was uploaded from an unblocked dropbox.com file, was a website, there had to make an additional step with a “cat-command” (nixCraft, 2015), which shows the file, in this case the picture-URL, from a file-system, here the uploaded HTML-dropbox-file.

Finally some arising challenges with the data, which are needed for the functionality of the online catalogue. After thinking und looking for suitable possibilities to add songs to a web-app, the best and most effective way was to implement a database. So first of all it was necessary to find a suitable tutorial to get to know how to write the source code for these demands.

## 7. References

Dr. Braun, S., 2015. *Webtechnologies (Lecture notes)*, Karlsruhe: s.n.

Dr. Wells, S., 2015. *Advanced Web Technologies (Lecture notes: SET09103)*, Edinburgh: s.n.

GitHowTo, 2015. *GitHowTo*. [Online]

Available at: <http://githowto.com/>

[Accessed 23 October 2015].

nixCraft, 2015. *nixCraft - Linux and Unix tutorials for new and seasoned sysadmin*. [Online]

Available at: [www.cyberciti.biz/faq/linux-unix-apple-osx-bsb-cat-command-examples/](http://www.cyberciti.biz/faq/linux-unix-apple-osx-bsb-cat-command-examples/)

[Accessed 25 October 2015].

Pratzner, A., 2015. *www.HTML-seminare.de*. [Online]

Available at: <http://www.html-seminare.de/>

[Accessed 25 October 2015].

Ronacher, A., 2013. *Flask web development, one drop at a time*. [Online]

Available at: <http://flask.pocoo.org/docs/0.10/>

[Accessed 27 October 2015].

w3schools.com, 2015. *w3schools.com*. [Online]

Available at: <http://w3schools.com/css/default.asp>

[Accessed 28 October 2015].

Figure 1: Logo .....	1
Figure 2: problematic Git comment .....	7
Table 1: URL history.....	3
Table 2: file structure .....	3