Edinburgh Napier
UNIVERSITY

Beatsprod Report 2

Eloi Filaudeau
40403956@live.napier.ac.uk
Edinburgh Napier University - Advanced Web Technologies (SET09103)

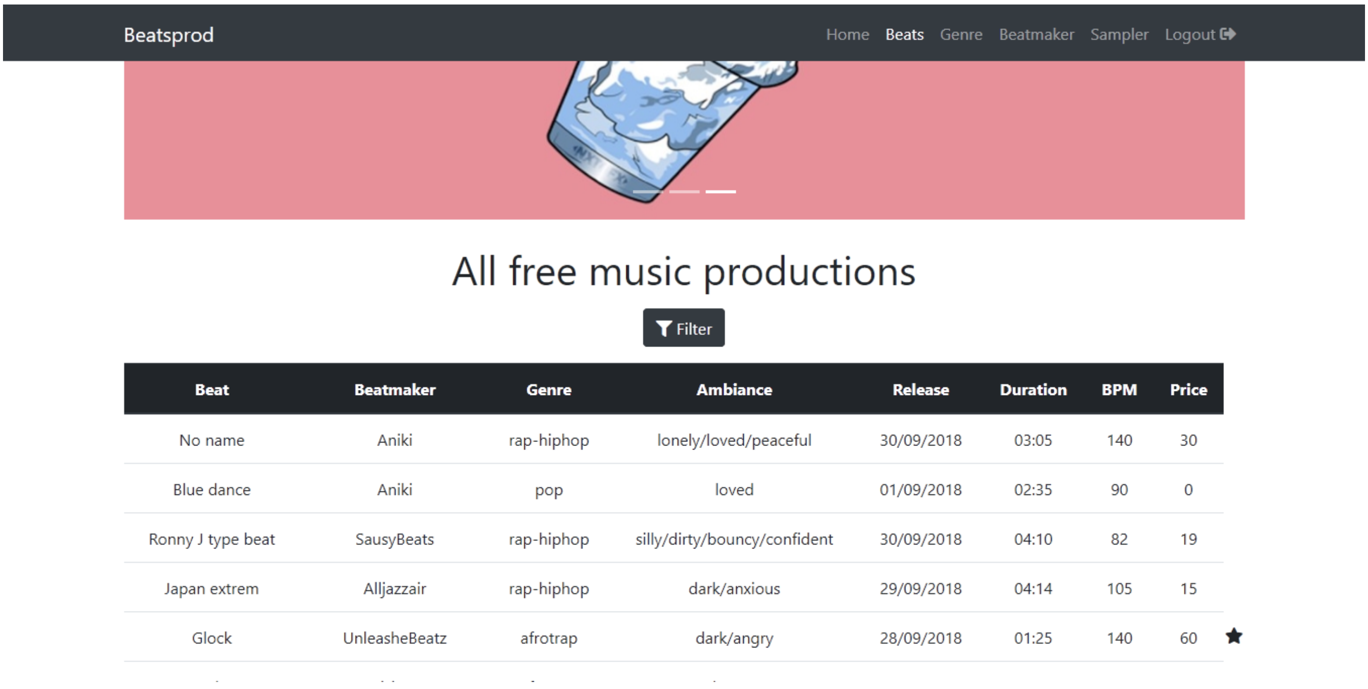
Abstract

In this document, you will find the results of my work on the second Advanced Web Technologies coursework. In this coursework, I have to develop my own web application using Python-Flask micro-framework to demonstrate my understanding of these technologies. This report is structured in 2 parts. In the first part, you can read the explanation of the work done about the development of my web application. And in the second part you can see my evaluation of the work I have done.

Keywords – coursework, web development, application, flask, python, music, keyboard, midi, report

1 Introduction

The first web application I developed, in the context of first coursework, aims to reference beat-type[1] musical productions by author, genre, etc. For the second coursework, I wanted to create a possibility of login in an account and a new functionality for beatmakers, the possibility to play music online. It's very interesting for a beatmaker to be able to play music easily anywhere. So I would like to facilitate this with Beatsprod web application. In the screenshot below you can see the login page which is the first page we get when we go to the web-app. The core features of my application are, firstly the referencing of beats with many search criteria and secondly the possibility of playing music with midi keyboard or computer keyboard online.



All free music productions

Beat	Beatmaker	Genre	Ambiance	Release	Duration	BPM	Price
No name	Aniki	rap-hiphop	lonely/loved/peaceful	30/09/2018	03:05	140	30
Blue dance	Aniki	pop	loved	01/09/2018	02:35	90	0
Ronny J type beat	SausyBeats	rap-hiphop	silly/dirty/bouncy/confident	30/09/2018	04:10	82	19
Japan extrem	Alljazzair	rap-hiphop	dark/anxious	29/09/2018	04:14	105	15
Glock	UnleasheBeatz	afrotrap	dark/angry	28/09/2018	01:25	140	60

Figure 1: **Web app overview**

[1] A *beat* is the instrumental part of a music, composed with rhythmic and melodic loops.

2 Design

The structure of my web application is the same as the first coursework. That is to say, it is URL based app that used the name of the web page or of the resource you are watching to construct the URL of the current page. Indeed, when a user search a beat by his beatmaker, the URL begin by ”/**beatmaker**”. As you can see on the navigation map (see Figure 2), as we get closer to the beat we are looking for, the URL is completed with the parameters chosen for the search. This hierarchy is important allows to show the user his path to the current data and he can use simply the URL to navigate into the application. I use this type of URL because a beat is characterized by many information, and the most important are the author, the genre and the name of the beat. So I just construct my URL with this information, to categorize the beat. But there is in this new version 2 other pages. The first is the home page with account informations (like username, mail) and the second is a virtual keyboard which I will detail in the rest of the report.

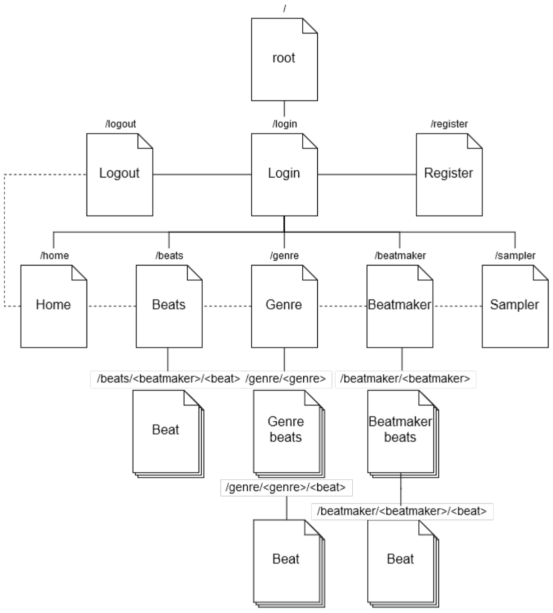


Figure 2: **Navigation map**

With regard to security, since a user of the application must create an account, it is necessary to ensure that the information entered is protected. That's why I first use REGEX [1] to force the user to use a strong password (Password must be contains at least one numeric, one uppercase and one lower case and six to twenty characters). In addition, passwords are encrypted via Bcrypt[2] [2] and stored in a database. I had also used functions to avoid XSS vulnerabilities in python (see Listing 1), but I realized that flask handles them on its own.

Listing 1: Python equivalent for PHP function *htmlspecialchars*

```
username = cgi.escape(username, True)
email = cgi.escape(email, True)
password = cgi.escape(password, True)
```

But now that I have an account system, it is important to add sessions for any user who logged in to protect access in personal information. Therefore, I create a *username* cookie for the recognition of the session opened by the user connected. With regard to the visual design and not the functional design, I use Bootstrap for implement more quickly and more properly the style part (CSS) and maybe media-queries, and Javascript for the animations of course. For the icons of the web I use FontAwesome and I create an animated favicon for the tab browser.

2.1 Login/Register page

My web application is structured in five main pages and a login/register form. I removed the contact form because it is currently useless. When we arrive in the website the first page we seen is a login form. If you are not registered you can also register you in another form (see Figure 3) with your username, mail and a strong password (a tooltip specifies the content of the password). You can not use a mail without '@' and you must fill in all fields before register you. As I say before, the password is encrypted with Bcrypt and the use of a hash protect the password to dictionary and brute-force attack and since I force the user to choose a strong password, it protect in part from rainbow table[3] too.

[2] *Bcrypt* is a password hashing function designed by Niels Provos and David Mazières, based on the Blowfish cipher.

[3] A *rainbow table* is a precomputed table of password hashes used for cracking password

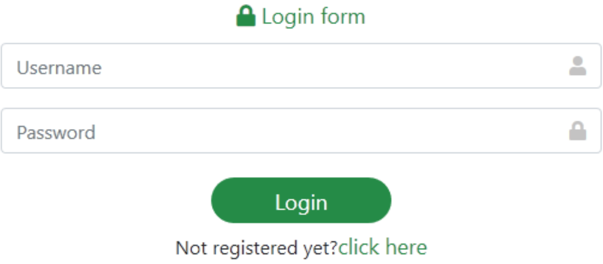
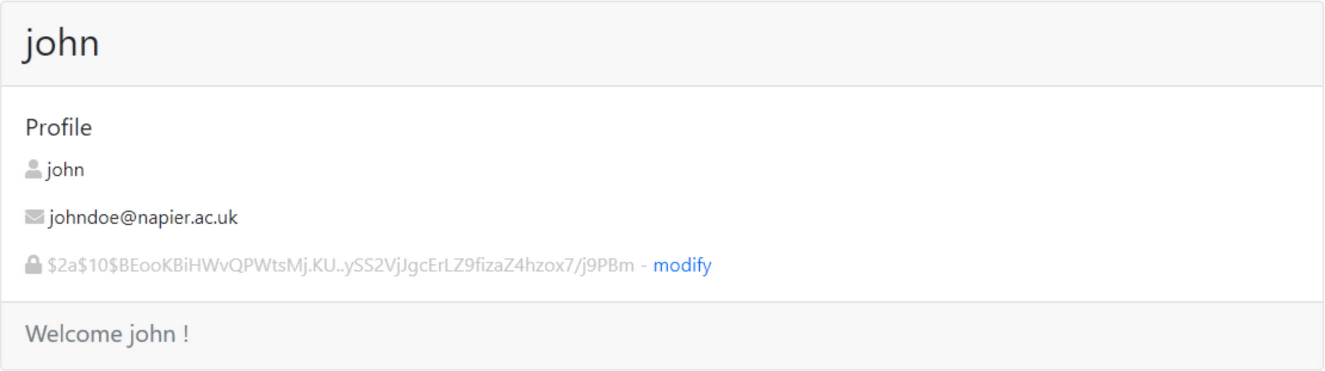


Figure 3: **Login page**

2.2 Home page

Once you are logged, a session is created and get your username in a cookie to check in every pages if you are logged in, and you arrive in a page that resume your personal data (see Figure 4). In this page you can change your password if you want, when you click in modify, a modal window appears and you can enter your actual password and choose a new one. If you are the admin of the web app you can see, under your personal info, the list of all users and it is possible to delete any user if you would (see Figure 5).



john

Profile

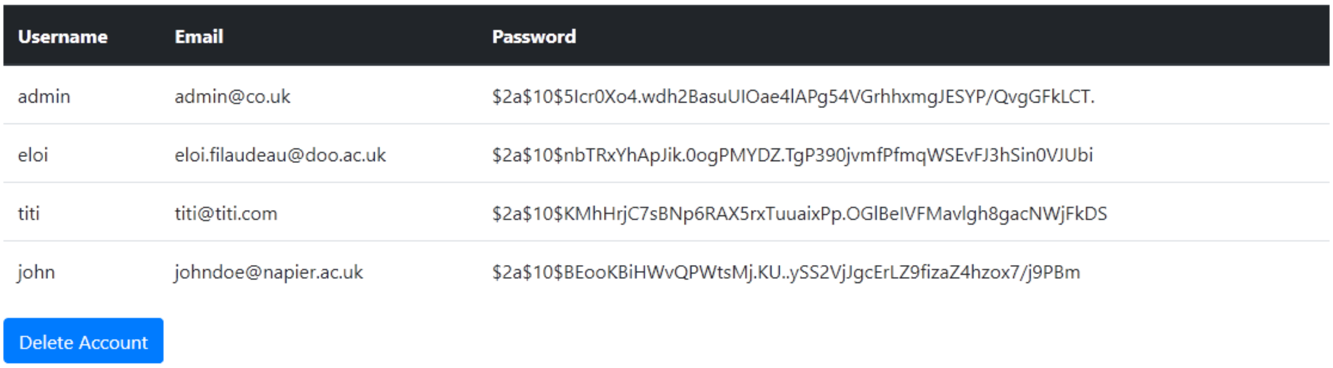
john

johndoe@napier.ac.uk

$2a$10$BEooKBiHWvQPWtsMj.KU..ySS2VjJgcErLZ9fizaZ4hzox7/j9PBm - modify

Welcome john !

Figure 4: **User information**

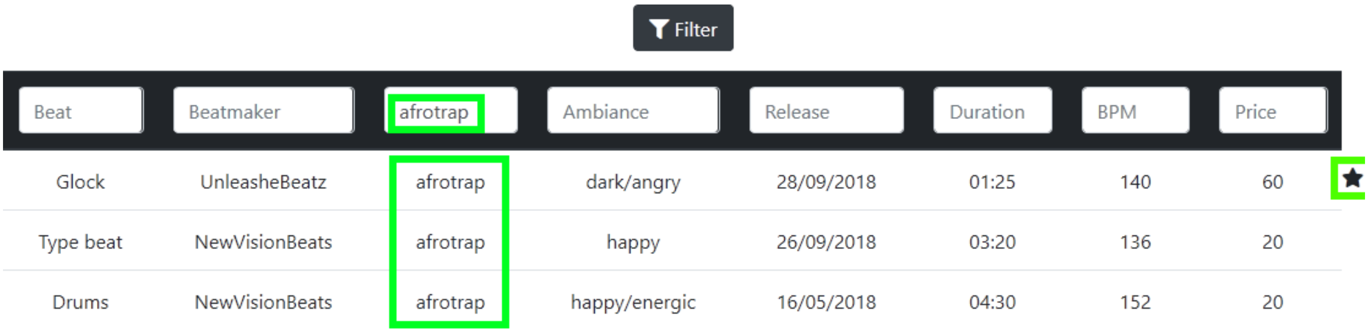


Username	Email	Password
admin	admin@co.uk	$2a$10$5Icr0Xo4.wdh2BasuUIOae4lAPg54VGrhhxmgJESYP/QvgGFkLCT.
eloi	eloi.filaudeau@doo.ac.uk	$2a$10$nbTRxYhApJik.0ogPMYDZ.TgP390jvmfPfmqWSEvFJ3hSin0VJUbi
titi	titi@titi.com	$2a$10$KMhHrjC7sBNp6RAX5rxTuuaixPp.OGlBeIVFMavlgh8gacNWjFkDS
john	johndoe@napier.ac.uk	$2a$10$BEooKBiHWvQPWtsMj.KU..ySS2VjJgcErLZ9fizaZ4hzox7/j9PBm

Figure 5: **Admin panel** - Users list

2.3 Beats page

Here is the page in which you can search for a beat, this display a carousel (for the attractiveness) and a table in which we can see the beats and their characteristics. If you want to search about a specific criteria like ”beatmaker”, ”genre”, ”duration”, ”BPM”, ”price”, ”release date”, ”ambiance” or ”beat name”, you can add a filter by clicking on the filter-button. This action displays form fields on the top of all the columns of the sorting table and in these you can write your search in a specific fields (see Figure 6). I have create a system of mark to select your favorites beats. When you add a beat a little star appears next is row in the sorting table.



Beat	Beatmaker	afrotrap	Ambiance	Release	Duration	BPM	Price	
Glock	UnleasheBeatz	afrotrap	dark/angry	28/09/2018	01:25	140	60	★
Type beat	NewVisionBeats	afrotrap	happy	26/09/2018	03:20	136	20	
Drums	NewVisionBeats	afrotrap	happy/energic	16/05/2018	04:30	152	20	

Figure 6: **Sorting table** - Filter form

2.4 Genre page

The fourth page is the **Genre** page, it categorize the beats by their genre, like *Rap, Reggae, Trap, etc.* You can choose a genre among six thumbnails (see Figure 7) and after you click on it, you can see all the beats that match this genre in a sorting table. You can now choose a beat and click it to read the description sheet and listen to it and you can add/remove it to your favorite (see Figure 8). You cannot buy it at this time it is just a button. Every beat is accompanied by his description: ”duration”, ”BPM”, ”price”, ”release date”, etc.

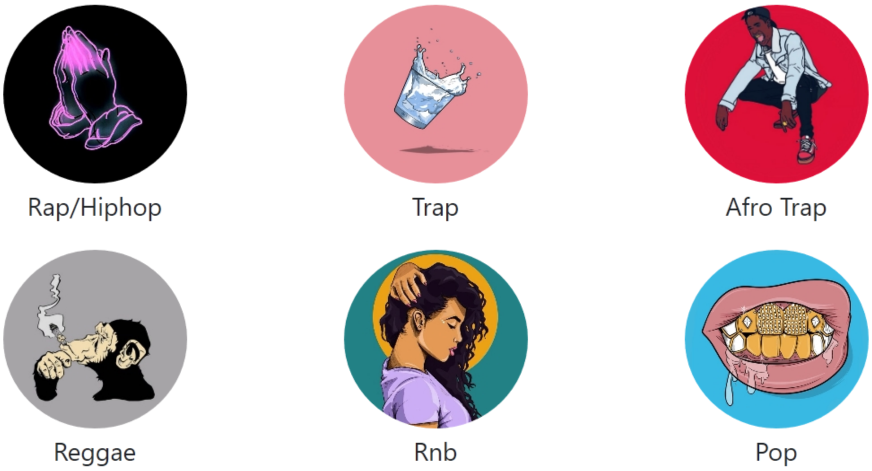
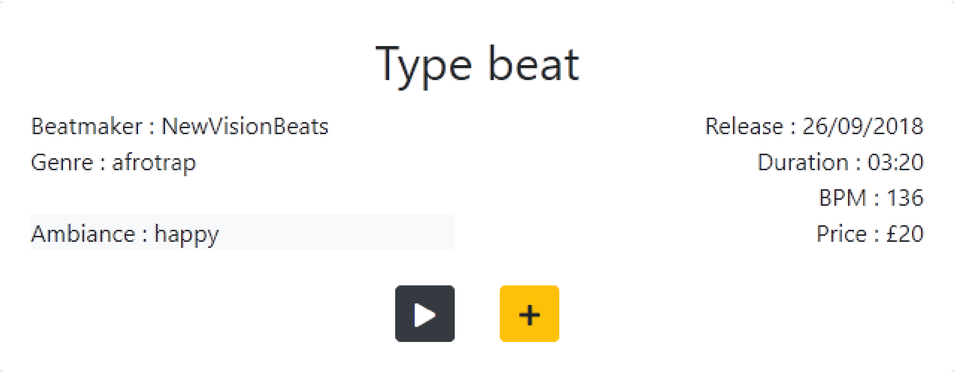


Figure 7: **Genre thumbnails**



Type beat

Beatmaker : NewVisionBeats

Genre : afrotrap

Ambiance : happy

Release : 26/09/2018

Duration : 03:20

BPM : 136

Price : £20

Figure 8: **Beat description**

2.5 Beatmaker page

The fifth page is about the **Beatmaker**, it shows simply a list of all the beatmakers and you can choose one and see his beats and their characteristics. If you click on a beat, you are redirected in the beat page as for genres.

2.6 Sampler page

This page is the real new functionality of my website. In short, this is a virtual device for playing music online, like a digital audio workstation (Flstudio, Cubase...) but who can use online everywhere, when you want, without any pre-installed software or instrument, just with a computer. As you can see on the screenshot below (see Figure 9) this instrument is structured in three part. At first we have a keyboard with 25 keys (2 octaves) with letters in any keys and you can play with this keyboard on a computer keyboard thanks the key with the same letter. The second part of the instrument is a sampler with9 pads that you can also play with the computer keyboard. And the third part is a little configuration and effects panel with a volume and an effect meters and 2 effects: *highpass* and *lowpass*.

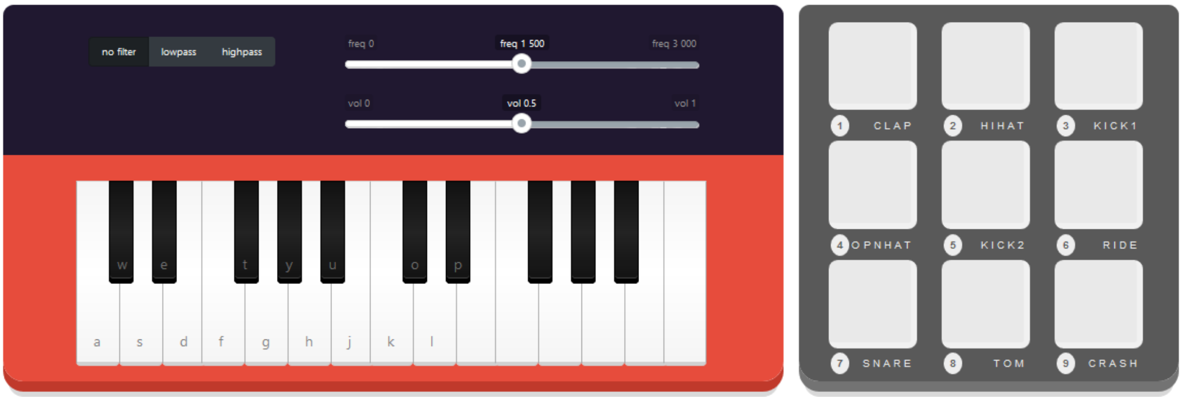


Figure 9: **Virtual instrument**

To implement this different parts I use 2 APIs.
The **Web Audio API** [3] for the sound part and effects, that use an Audio context, a system of nodes for the effects and an oscillator for create note with a certain frequency (see Figure 10). For the volume and effect master I use the IonRangeSlider plugin to get a nice and personalized slider. The Audio API uses a system of nodes, every effects (gain, reverb, lowpass...) is a node and we have to connect them to play sound (see Listing 2).

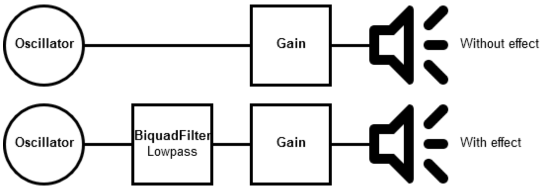


Figure 10: **Audio API** - explanation

Listing 2: extract of sampler.js - highpass function

```
case "highpass":
        osc.disconnect(); //disconnection of all nodes
        osc.connect(biquadFilter); //oscillator connection with filter node
        biquadFilter.connect(gain); //filter node connection with volume node
        biquadFilter.type = "highpass"; //filter initialization
        biquadFilter.frequency.value = 1200; //filter initialization
        break;
[...]
```

The **Web Midi API** [4] for the midi device recognition and gestion (see Listing 3). In fact, in addition to computer keyboard, you can also use a midi keyboard to play music with the pads and the keys. This API is only developed on Chrome, but with addons and plugin you can use midi on Firefox, so my application is fully functional on Firefox and Chrome which represents about 70% of the web browser market. If you would use this API on Firefox you must add the Jazz-midi [5] for midi device recognition, the Web-midi-api [6] plugin for the API recognition and finally install Jazz plugin [7] for your OS. This new Midi API is a revolution for CAM (Computer Assisted Music) and beatmakers because the midi[4] is the most important format of data in this domain.

[4] The *midi* is a data format that contains information about each of the notes to be played (duration, volume, channel number, velocity...)

Listing 3: extract of sampler.js - Main midi functions

```
function onMIDISuccess(midiAccess);
function onMIDIFailure(error);
function onMIDIMessage(message);
```

3 Enhancements

I thought of some improvements that I would add with more time. First, I would like to add a new effect for the virtual keyboard, the reverb, with a convolver node that I could add in the audio circuit, I had started to implement it. Then I would like to use the Soundcloud API to enlarge the database of beats and to play the beat with an integration of the Soundcloud audio player. I have documented how the API works and I think it is also possible to create a connection with Soundcloud account because many artists have a Soundcloud account for their own productions. Furthermore, I would add the possibility to add your own productions on the platform. And maybe the possibility to change your mail must be added to the app. Finally, the current mark system to select your favorites beats is the same for all users, so I have to had a favoriteBeats database for each user to separate the choice of each one.
Currently, my database calls and web page construction are done on the server because I didn't have time to work on it better, but I plan to make the page be built by the browser using Ajax requests. I think also that the password encryption can be improved with a salt to avoid the hacking with rainbow-table, but with a strong password is it really useful?

4 Evaluation

4.1 Critical Evaluation

I am particularly proud of my work on my virtual instrument, which allows anyone to play both with the keyboard and the mouse (for pads) and above all offers the possibility of using a midi keyboard. This allows anyone to play music online without installing any software (except for Firefox which requires a plugin). I have put a lot of work and time into the comprehension and development of this keyboard and I am happy with the result. But I think I have to clean the code and improved my ability to use Web Audio and Web Midi API to develop a best code.
The music files are always store locally and it can quickly become heavy for the site and slow it down, so the next step is the implementation of Soundcloud web player (see Listing 4). In the current database I have only 27 entries but I decided to use only one music for each genre as a demo. To fix the fact that storing music locally could slow down the application, I blocked preloading the music from the web player to allow the page to load faster.

Listing 4: Soundcloud player in Javascript

```
var track_url = 'http://soundcloud.com/forss/flickermood';
SC.oEmbed(track_url, { auto_play: true }).then(function(oEmbed) {
  console.log('oEmbed response: ', oEmbed);
});
```

I have also created my own error pages with an error handler for a better rendering (see A).To check the routing validity, I performed unit tests for each of the routes by verifying that the return was either "200 OK" or "302 FOUND" (see B). I have store the data about the beats and users in a CSV file save in a database file thanks to a Python script [8]. I have placed the connection functions to the database in two files separate from the application's routing, one for the search beats part and another for the log and sampler part.

4.2 Personal Evaluation

I am very happy with the work I have done, because I have learned a lot, I have spent more time on this and I like the result. I have improved my use of Flask and Jinja2, which offer a very interesting combo. I also liked using Bootstrap it is a real time saving and it look nice thanks is grid system. I will be able to add these technologies to my skill set. In addition I discovered 2 new interesting APIs for the using of audio and Midi and it is a domain I especially appreciate. I think that the midi API has a future because many people are waiting for it to be implemented on browsers other than Chrome. It's a small revolution for music on the web.

References

[1] "Python regex - http://apprendre-python.com/page-expressions-regulieres-regular-python." Accessed: 20/11/2018.

[2] "Bcrypt for flask - https://flask-bcrypt.readthedocs.io/en/latest/." Accessed: 20/11/2018.

[3] "Web audio api - https://webaudioapi.com/." Accessed: 25/11/2018.

[4] "Web midi api - https://webaudio.github.io/web-midi-api/." Accessed: 25/11/2018.

[5] Jazz-Soft, "Jazz addon for firefox - https://addons.mozilla.org/fr/firefox/addon/jazz-midi/." Accessed: 18/11/2018.

[6] Jazz-Soft, "Web midi api addon for firefox - https://addons.mozilla.org/fr/firefox/addon/web-midi-api/." Accessed: 18/11/2018.

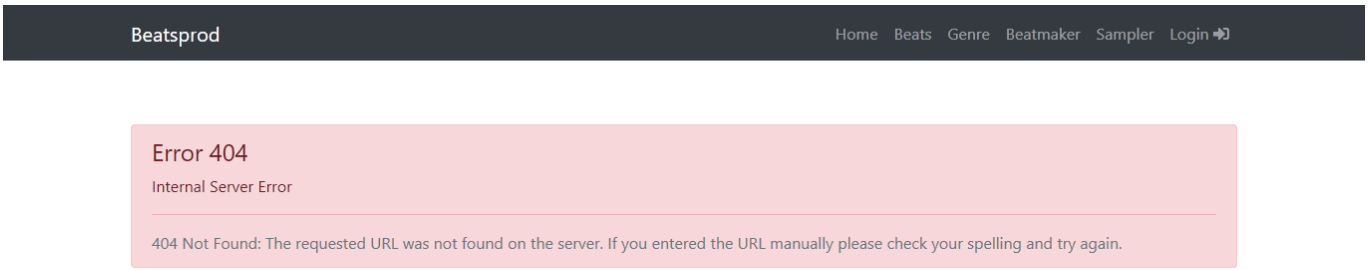
[7] "Jazz plugin - https://jazz-soft.net/." Accessed: 15/11/2018.

[8] A. Guillonneau, "transfodata.py,"

Appendices

A Error handler

```
@app.errorhandler(404)
def page_not_found (error):
    message = "Internal Server Error"
    return render_template('error.html', nb=404, message=message, error=error)
[...]
```



Error 404

Internal Server Error

404 Not Found: The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.

Figure 11: **My own error**

B Unit tests

```
import unittest
import app

class TestingTest(unittest.TestCase):
    def test_root(self):
        self.app = app.app.test_client()
        out = self.app.get('/')
        assert '200 OK' or '302 FOUND' in out.status
        assert 'charset=utf-8' in out.content_type
        assert 'text/html' in out.content_type

    def test_home(self):
        self.app = app.app.test_client()
        out = self.app.get('/home')
        assert '200 OK' or '302 FOUND' in out.status
        assert 'charset=utf-8' in out.content_type
        assert 'text/html' in out.content_type
[...]
```