# SET09103 Coursework Assignment 1 – Report

Brian Giggle – 40121723

## 1 Introduction

The web application that I chose to create was to be a collection of video games, this application will display a sortable and searchable collection of games. I chose to create a site based off a collection of video games as I found a resource that could provide a large amount of data freely available (www.giantbomb.com) this resource is provided freely to anyone who signs up for an API key.

Further to learning how to develop a web application using Python and flask, I also decided to do some client side implementation of an angular JS application, this would allow me to give the user a richer experience when viewing the content that my flask application would be providing.

## 2 Design

The structure of the web application revolves around a RESTful API, which provides a JSON data source for the client side application to consume and render it to the clients browser. The client side application makes requests to the python flask application running on the server and the server returns the collection or single entity that was requested.

To avoid creating many large request to the GiantBomb API I have cached the data for collections in a small database that is used when retrieving data for the collection, this also allows me to be able to search and order the data in a way that is easy for the user.

```
@app.route('/Games/', methods =['GET'])

def getGames():

   start = int(request.args.get('start', ''))

   end = int(request.args.get('end', ''))

   filter = json.loads(request.args.get('filter', ''))

   limit = end - start

   games = getGamesSQL(start, limit, filter)

   return jsonify(games);
```

As can be seen form the code above the /Games/ route of the web application takes in start and end parameter along with a filter JSON object, this object is used to filter the results of the sql query. The results of the getGamesSQL function is then turned into JSON and returned to the client.

When retrieving just one game, my python flask application makes a call to the GiantBomb API retrieves the data then sends this on to my client application to be displayed, this is a simple task and uses Pythons “request” library, to create a request for a specific game given its ID.

## 3. Enhancements

### 3.1 SQL security

Currently my web application will execute any sql that is in the filters which is a big security concern, I would like to add the functionality to protect my data by making sure the filters do not contain anything malicious.

### 3.2 Improved search

I would like to add the ability to search on more fields than just the title, I would also like to off the ability to search by platform or any other of the given fields. This would require a slight restructure of my data in the database to allow extra fields to be made searchable. Possibly moving away from a SQL data source to a more search friendly data store such as Elastic would allow much improved search and reduction of waiting times.

### 3.3 Storing user preferences

Giving the ability for the user to save searches or favorite particular games could allow the user some flexibility.

### 3.4 Data visualisation

Given much more time I would have liked to allow the user to visualise the data not just as a list of games but also be able to chart things such as average rating by year, or games per console per year and other such abilities. I would possibly acieve this by using Highcharts (a JS charting library) with custom controls that the user could choose which data to chart in which way, this would also require improved search capabilities to provide more complex and structured data.

### 3.5 Error handling

I would also like to add error handling, that would handle errors and a meaningful way and present them to the user.

## 4. Critical Evaluation

The web application performs quite well, data is retrieved relatively quickly and is presented to the user in a readable way.  I feel the use of the AngularJS framework allows the user to have a good experience when using my web application. The use of a client side framework allows the application to offload some of the processing to the clients, in some cases this can create a lesser user experience, if for example the user is browsing from an underpowered machine they could see a slight performance drop.

## 5. Personal evaluation

I feel I have learnt a great deal about using python and flask during the course of this tutorial, I also decided to draw on some of my previously developed skills by using Angular JS and bootstrap whilst developing my application. I have also learnt a lot about the use of linux and VIM as I had not used either of these before in a meaningful way.

## 6. Resources

Flask: http://flask.pocoo.org/

Levinux: http://mikelev.in/ux/

SQLite3: https://www.sqlite.org/

AngularJS: https://angularjs.org/

JQuery: https://jquery.com/

Bootstrap: http://getbootstrap.com/

Angular Bootstrap: https://angular-ui.github.io/bootstrap/

GiantBomb Api: http://www.giantbomb.com/api/

## 7. References

Dr Simon Wells, Notes and workbook:
http://moodle.napier.ac.uk/pluginfile.php/955615/mod_resource/content/10/workbook.pdf

SQLite Documentation:

https://www.sqlite.org/docs.html

AngularJS Documentation:

https://docs.angularjs.org/api

Angular Bootstrap Documentation:

https://angular-ui.github.io/bootstrap/

GiantBomb api Documentation:

http://www.giantbomb.com/api/documentation