

Final Project Stream Two

uk fishing industry

Christine Wilde | Full Stack Software Development | 29.09.2018

Contents

[Project Brief 4](#_Toc525979739)

[The Prototype 8](#_Toc525979740)

[Back-end setup 12](#_Toc525979741)

[Front-end setup 17](#_Toc525979742)

[Testing results 20](#_Toc525979743)

[The complete product 37](#_Toc525979744)

[Conclusion 41](#_Toc525979745)

[Preparing the Project for Heroku deployment 42](#_Toc525979746)

[Appendix A: References 50](#_Toc525979747)

# Project Brief

The goal of the stream two project is to build a front and backend Website that pulls data from a database and presents the data in a well-presented graphical format using technologies such as Python, D3.js and DC.js.

The subject chosen is the UK fishing industry and the quota restrictions that the UK currently abides to via the EU fishing laws in EU waters. One of the key mechanisms to control fishing due to fear of ‘over-fishing’ is to implement a fishing quota on nations. The aim here is to analyse the UK fishing industry statistics over a 5 year stretch to gain an understanding of how we are performing due to press coverage that always highlights a concern. For example a well-known British MP Boris Johnson has stated:

"[The EU is] Pinching our fish" – Boris Johnson, The Telegraph, 16th June 2016.

The Telegraph itself has reported that the 23% quota given to the UK now goes back into a Dutch crawler that therefore lands back in Holland.

The aim of project two is to look at the actual UK government figures to obtain a factual picture of the state of our fisheries.

I have therefore chosen to build a fishing dashboard that highlights the fishing statics in graphical format and eye-catching design.

Project Keywords

Fish - Pelagic

Most of the world's large fish populations are pelagic. They include tuna, herring, mackerel, sardines and anchovy.

Fish – Demersal

According to the Scottish Government demersal fish live on or near the seabed and feed on bottom-living organisms and other fish. They include cod, haddock and monkfish.

Fish – Shellfish

This includes lobsters, crab and scallops.

Landings

This is defined by the OECD as the catches of marine fish landed in foreign or domestics ports.

\*Please see appendix A for references

Statistics

Cod levels over the years from total landings (after previous years reports of a dire situation for cod).

Technical inclusions

The list below presents the technologies used to accomplish the Fishing Dashboard Website from the initial prototype versions to the completed Website product.

**Back-end**

D3.js

* This JavaScript library was used to build the fishing graphs and make them interactive based on the data pulled from Mongo DB.

DC.js

* DC. js JavaScript helped to plot the graphs more easily.

Crossfilter.js

* The Crossfilter library was used to enable the two-way binding i.e. filtering and aggregating data across datasets.

Queue.js

* The queue() JavaScript function was used in order to pull data from the API.

Mongo DB

* Mongo was chosen since it the intention was to use a noSQL database that is growing in popularity and to become even more familiar with it by using it in the project. Additionally, JSON file format is versatile and useful for browser /server communication.

Flask

* The Flask framework application routes were used to deliver the stored Mongo DB data to the dashboard front-end Website.

**Front-end**

(Semantic) HTML5

* Website HTML structure – DOM

CSS

* General and advanced Website display including animation (external style sheet)

JavaScript

* JavaScript language was used to build the graphs via D3.js and DC.js and allow the Website to be responsive

Bootstrap

* The free front-end framework in order to contribute to designing an eye-catching Website

Project deployment

* GitHub, GitBash, GitHub Pages

Hosting the software project files and version control. Creating repositories and branches for project files.

* Heroku

Heroku was used to host the project’s files, application and database on the Web.

# The Prototype

The below charts were chosen for the fishing project. It was decided that one chart should feature at the very top and show the data in full over the years. The best chart for this was the line chart.

Next a feature was created where the user could select the values by UK region – this should also be placed at the top of the page together with the numeric figure totals. Both the value in sterling and the number caught in total should be displayed.

Further down the page users can then select fish by name and by main UK ports. It was important to add the fish names since the project is to question if cod is still showing high numbers throughout the years.

Charts required:

The lineChart was necessary to show the amount of fish types through the years and to answer the project question “How well is our fishing going”? The charts chosen are depicted below.

**Title: Quantity of fish through the years**

Y = Quantity of fish

X = Date in years

**rowChart to answer “What type of fish are being caught”?**

Title: Fish types

row = Fish types

X = Quantity of fish

**pieChart to answer “How much fish is caught by each region”?**

Title: Fish by UK region

**rowChart to answer “What are the main ports that catch the fish in the UK”?**

Title: Main regional ports

Row = Main port

X = Value

**selectMenu to Select by fish name**

Title: Fish types

**SelectMenu to Select by fish name**

Title: Fish name

**Metric 1 for The quantity of fish**

Title: Total quantity of fish

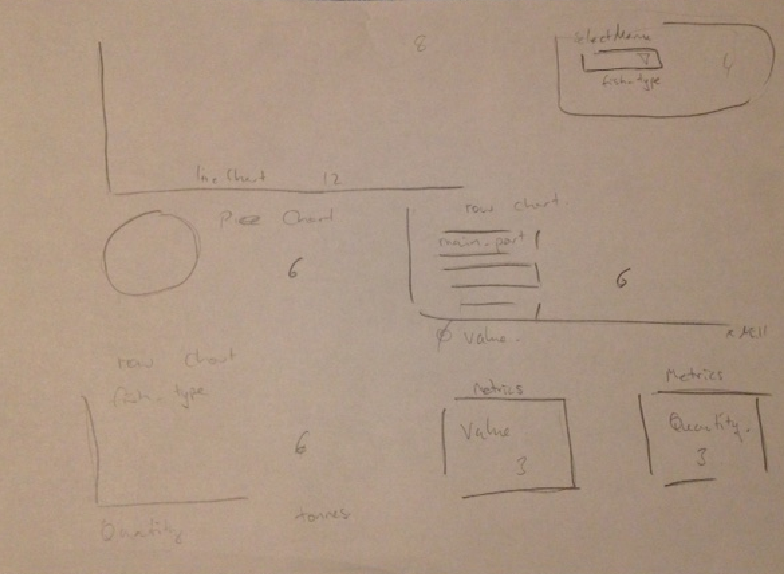
**Metric 2 for The value of fish**

Title: Total value of fish

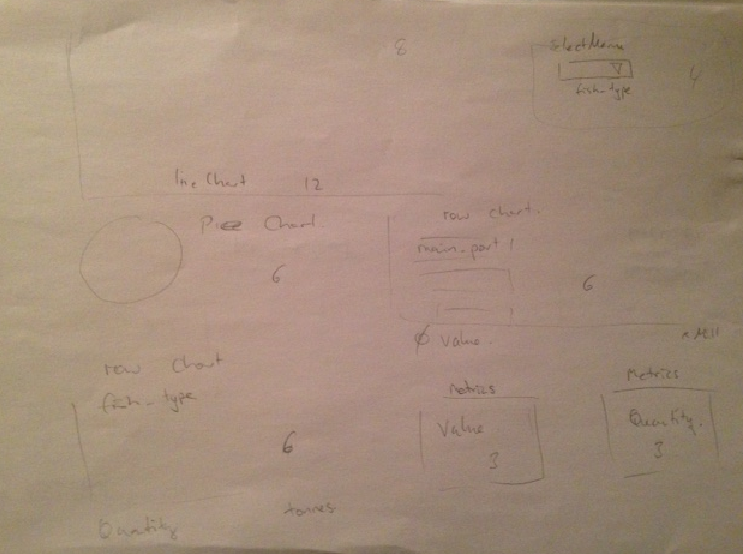
The below pictures show the sketches that were created at the beginning of the prototype design. It’s an important step that can save a lot of time. In some of the sketches the about the Bootstrap grid system was considered and how each graph can be laid out according to the grid framework.

The next task was to add more details such as the chart titles, chart D3.js names and the database fields it should pull from.

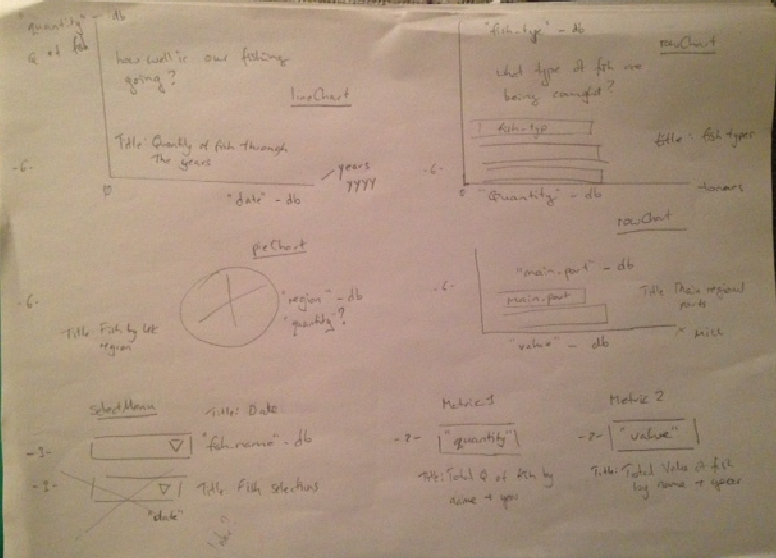
**Figure 1. Early sketch of the main graphs required**



**Figure 2. Sketching the graphs and focusing on the grid layout**



**Figure 3. Adding more details (e.g titles and chart names)**



# Back-end setup

The below steps were followed to get the Mongo DB up and running with the fishing statistical data. Mongo DB was used since the database is set in the versatile JSON file format. Using Mongo means that each record is saved in a document where data is depicted by a key and value. It does not require primary keys and the table relationships that SQL tables do.

The mongo directory was opened as an Administrator using Windows cmd prompt at the following location: cd Program Files\MongoDB\Server\3.6\bin

To open the Mongo shell the following command was typed: **mongod**

The fishing csv data file was placed in the same bin folder location i.e. Program Files\MongoDB\Server\3.6\bin.

 Another command line was opened again as administrator and this time the **mongo** command was used to open up the database.

**Figure 4. Mongo DB**

Machine generated alternative text:
Admin cmd prompt - mongo 
icrosoFt Windows [Uersion 6 .3 
.9600] 
(c) 2013 Microsoft Corporation. 
All rights reserved. 
: •cd Program .6\bin 
: \Program .6\bin >mongo 
ongoDB shell version u3.6.ø 
onnecting to: mongodb://127.Ø 
ongoDB server version: 3.6.ø 
eruer has startup warnings: 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
.56 
:11.66ø.øøøø 
.661 .øøøø 
.661 .øøøø 
:11.662.øøøø 
:11.663.øøøø 
:11.664.øøøø 
:11.664.øøøø 
:11.665.øøøø 
:11.668.øøøø 
.669.øøøø 
.67ø.øøøø 
.671 .øøøø 
.671 .øøøø 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
CONTROL 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
[initandlisten] 
WARNING: 
WARNING: 
WARNING: 
Access control 
Read and write 
This server is 
Remote systems 
is not enabled For the database. 
access to data and conf iguration is u 
bound to localhost. 
will be unable to connect to this ser 
Start the server with 
——bind_ip (address > to specify 
addresses it should serve responses From. or with — 
bind to all interfaces. IF this behavior is desired. 
server with 
——bind_ip 127.ø.ø.I to disable this warn 
The File system cache OF this machine is conf igured 
See http://dochub.mongodb.org/core/wt—windows—system—file—cache 

 The command to import the fishing csv data file was as follows:

mongoimport -d fishingUK -c projects --type csv --file fishing\_data\_landings.csv --headerline

* The database is called **fishingUK**
* The collection name is **projects**
* The file type is **csv**
* The file name is **fishing\_data\_landings.csv** and it contains a header

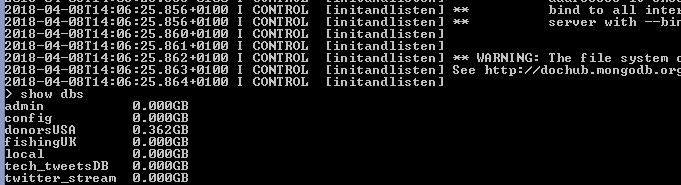
**Figure 5. Mongo DB directory**



 Next the following commands were run to ensure the data imported successfully:

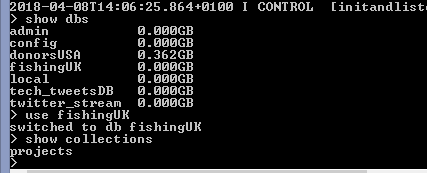
* **show dbs** (this shows the available databases that have been created).

**Figure 6. Querying Mongo DB databases**



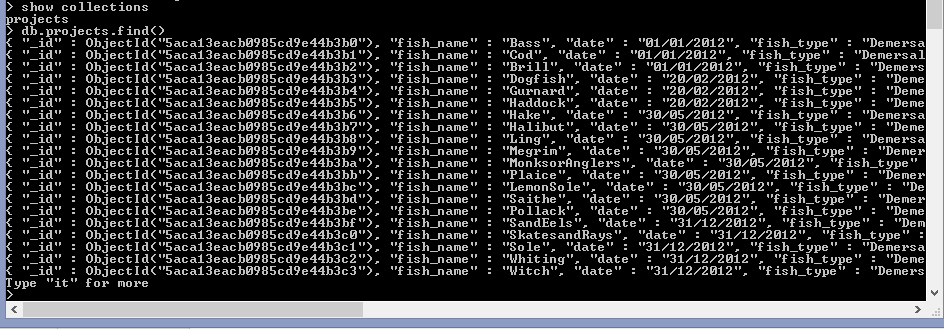
* **use fishingUK** (to switch to the fishing database).
* **show collections** (to show what collections have been created).

**Figure 7. Querying Mongo DB collections**



* **db.projects.find()** (to show all the records -Json format).

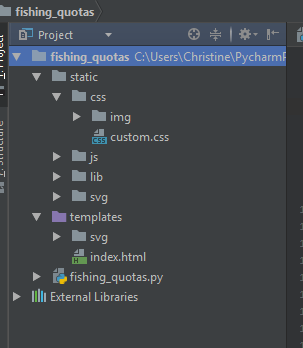
**Figure 8. Querying Mongo DB database records**



# Front-end setup

The below folder structure was created in Pycharm.

**Figure 9. Folder structure**



The custom files for the project were found under: CSS custom file (\static\css ), the JavaScript custom file for the graphs (\static\js) and a svg folder for the fish animation (\static\svg).

All other library folders such as Boostrap and tool tips .js and .css files are found under the lib folder e.g (static\lib\js).

This was to maintain separation of library framework files and the project custom ones.

Index.html

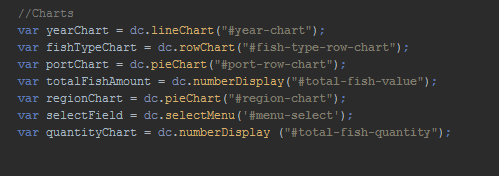
The index html file contains the graphs that were chosen in the prototype phase.

Bootstrap allows a grid system based on 12 columns and that was used to nest the graphs into the desired sequence that was planned in the prototype phase.

Graph.js

This file contains all the JavaScript that allows the graphs to be interactively constructed with the relevant data.

**Figure 10. DC.js charts in graph.js file**



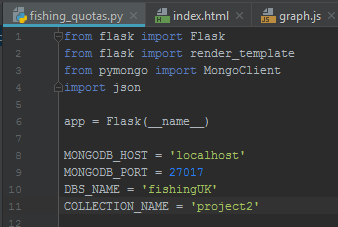
Custom.css

The custom.css file has all the CSS required to style the individual graphs, background and dashboard in general.

The name chosen for the project was fishing\_quotas and created in a virtual environment.

The fishing\_quotas.py Python file contains the relevant imported packages required to make the application work. Flask is used here to make the connection to the Mongo DB and allow the project’s fishing data to be pulled using the computer’s local host connection. This worked very well with no issues.

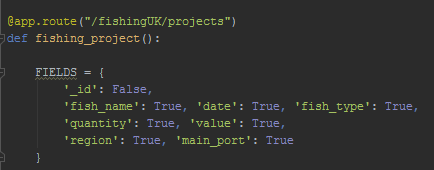
**Figure 11. fishing\_quotas.py file**



The two routes were defined. First the route to the home page template called index.html which contains the html structure for the fishing dashboard.

The second route defines the fishing database details where the fields that need to be pulled are ultimately outlined.

**Figure 12. fishing\_quotas.py routes**



# Testing results

The below table summarizes the testing results whilst building the Pycharm project.

|  |
| --- |
| MongoDB file import |
| Issue: Spaces in file  Machine generated alternative text: File Edit Format View Help  fish_name, date, fish_type, quantity , value, region , main_port, ,  Bass , 31/12/2012, Demersal, 5 , England , Newlyn, , ,  Brill, 31/12/2012, Demersa1,Ø, 2, England, , ,  cod, 31/12/2012, Demersal, 2, 3, England, Newlyn„,  Dogfish , 31/12/2012, Demersal, 1, England , Newlyn , , ,  Gurnard , 31/12/2012, Demersal, 1, 1, England , Newlyn, , ,  Haddock, 31/12/2012, Demersal, 3 , 3 , England, Newlyn, , ,  Hake, 31/12/2012, Demersal, 1, 1, England, Newlyn, , ,  Halibut, 31/12/2012, Demersal, England , Newlyn , , ,  LemonS01e, 31/12/2012, Demersal, 2, 5 , England , Newlyn , , ,  Ling, 31/12/2012, Demersal, England, Newlyn, , ,  Megrim, 31/12/2012, Demersal, 1, 2, England , Newlyn , , ,  MonksorAng1ers , 31/12/ 2012, Demersal, 3 , 9, England , Newlyn , , ,  Plaice, 31/12/2012, Demersal, 2, 3 , England, Newlyn, , ,  Pollack, 31/12/2012, Demersal, 1, 3, England, , ,  Saithe, 31/12/2012, Demersal, England , Newlyn , , ,  SandEe1s , 31/12/2012, Demersal, England , Newlyn , , ,  SkatesandRays , 31/12/2012, Demersal, 2, 3 , England , Newlyn , , ,  sole, 31/12/2012, Demersal, 2, 14, England, , ,  Turbot, 31/12/2012, Demersal, 3 , England , Newlyn , , ,  Whiting, 31/12/2012, Demersal, 2, 1, England , Newlyn , , ,  6 items  I item selected 43.8 KB  The above file caused mongodb error: Failed: fields cannot be identical: '' and ''    The file was subsequently cleaned (per below) and another import attempt done. The cleaned file imported successfully.  Machine generated alternative text: File Edit Format View Help  ish_name, date, fish_type, quantity , value, region , main_port  Bass , 31/12/2012, Demersal, 5 , England , Newlyn  Brill, 31/12/2012, Demersa1,Ø, 2, England, Newlyn  cod, 31/12/2012, Demersal, 2, 3, England, Newlyn  Dogfish , 31/12/ 2012, Demersal, 1, England , Newlyn  Gurnard , 31/12/2012, Demersal, 1, 1, England , Newlyn  Haddock, 31/12/2012, Demersal, 3 , 3 , England , Newlyn  Hake, 31/12/2012, Demersal, 1, 1, England , Newlyn  Halibut, 31/12/ 2012, Demersal, England , Newlyn  LemonS01e, 31/12/2012, Demersal, 2, 5 , England , Newlyn  Ling, 31/12/2012, Demersal, England , Newlyn  Megrim, 31/12/ 2012, Demersal, 1, 2, England , Newlyn  MonksorAng1ers , 31/ 12/ 2012, Demersal, 3 , 9, England , Newlyn  Plaice, 31/12/2012, Demersal, 2, 3 , England , Newlyn  Pollack, 31/12/2012, Demersal, 1, 3, England, Newlyn  Saithe, 31/12/ 2012, Demersal, England , Newlyn  SandEe1s , 31/12/2012, Demersal, England , Newlyn  SkatesandRays , 31/12/2012, Demersal, 2, 3 , England , Newlyn  sole, 31/12/2012, Demersal, 2, 14, England, Newlyn  Turbot, 31/12/2012, Demersal, 3 , England , Newlyn  Whiting, 31/ 12/ 2012, Demersal, 2, 1, England , Newlyn |
| Initial Flask Tests |
| Virtual Environment    Requirements File    Simple Hello World test    Localhost    Test Route for Fishing\_quotas and load the index.html file test    Localhost    Load empty database |
| Data retrieval error |
| Issue: Server error indicating issue with data      A quick search on Python bugs highlights the issues (<https://bugs.python.org>)    The fishingquotas.py file is passing through the accumulating mongo DB object ID which is not required and causes errors.    Under the fields listed “\_id: False” was added to stop flask passing through the Mongo DB ID for the fishing data. This fixed the issue. |
| Graph name error |
| Issue: DC.js and D3.js error on graph  The dashboard cannot show any data in graphs due to errors.  DC.js error:    D3.js error:    Index.html shows a typo error – “ear-chart” should be “year-chart” |
| Date value error |
| Issue: variable numFishByDate that groups on field “date” is causing errors    The code was viewed in detail on repl.it. The date format was incorrectly constructed. The date format stored in Mongo DB was DD/MM/YYYY and not DD-MM-YYYY. |
| After correcting the date format the Line chart started working and retrieved the data. |
| Further dateDim error after adding more graphs |
| Issue: dateDim error again after adding further graphs    Both Pycharm and dev tools highlighted the syntax issue although dev tools showed this more clearly. |
| Crossfilter |
| Issue: Missing crossfilter  This means that applying data selection on one graph will not apply across to other graphs (two-way binding) so this will not work. Crossfilter needs to be applied. |
| Clearing cache! |
| Issue: Last page cached  Every re-load of the browser page presented the cached pages from previous testing. Ctrl+Shift+Delete could be used to clear the cache however a more convenient solution was to temporarily disabled it during testing under Chrome More tools>Developer Tools>Network    The browser now displays the new/refreshed pages |
| Syntax Error |
| Issue: Syntax error after adding region chart  Unexpected token due to ‘comment characters’ mistakenly left in code |
| Missing dimension |
| Issue: Missing Region dimension  The below dimension for Region pie chart was missing thereby causing Chart errors |
| Region pie chart error |
| Issue: Syntax error  Per below a syntax error was shown which stopped data from being retrieved for the charts      After further cross examination it was found that the syntax error was due to a semi colon further up in the code by the dimensions:    After removing the semi colon the pie chart successfully retrieved the data. |
| Adjusting the numeric totals |
| Formatting: Changing the quantity totals to numbers per preference  The “,d” format was chosen for quantity since the preference was to show the full number of quantity with leading zeros |
| Pie chart styling |
| Formatting: Pie chart style  The styling of chart titles was best coded using CSS inline styling, example: |

# The complete product

The nav bar

**Figure 13 – The nav-bar**



The navigation bar was constructed using Bootstrap and fixed to the top of the page: navbar navbar-inverse navbar-fixed-top.

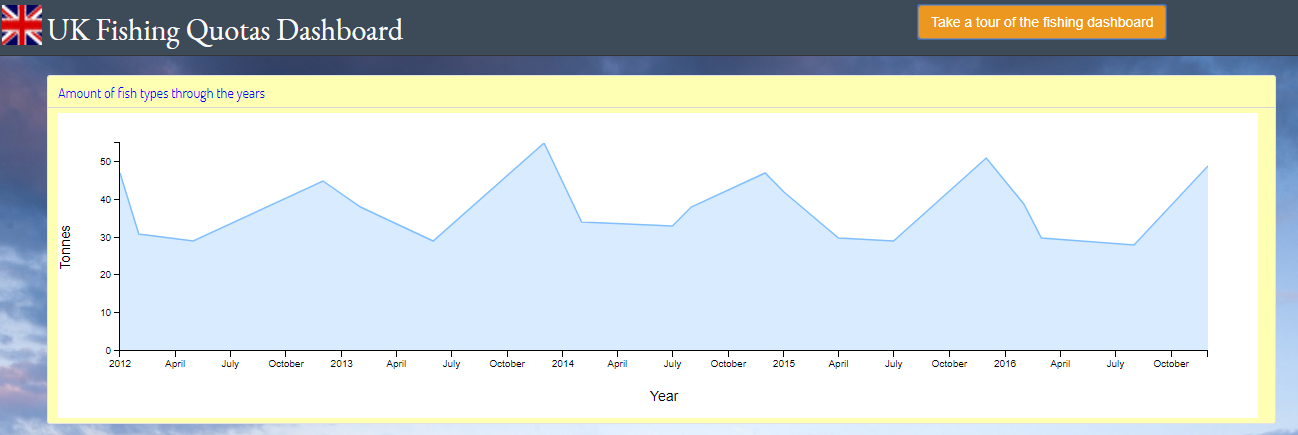
The idea was to keep the nav bar smart and more conservative – the tour button which was the tool tips functionality had to stand out so this was done using Bootstrap warning button: btn btn-warning.

Line Chart – Amount of Fish Types Through the Years

This chart featured as the main chart at the very top of the Webpage. It pulled data from field “date” and served as the dimension (x) layout, then the metrics grouped the amounts for each region.

The data updates when selecting the region or port chart options.

**Figure 14. The Line Chart – Amount of Fish Types Through the Years**

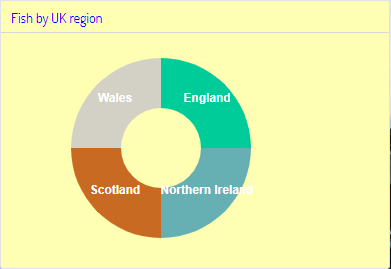


Pie Chart – Fish by UK Region

This provides the user the option of obtaining statistics for individual UK regions.

Scotland has the largest number of caught of fish with 1.88 billion tons over a 5 year span. Wales has the least with 78.3 million tons. Overall there has been healthy landings of fish throughout the last 5 years for the UK.

**Figure 15. The Pie Chart – UK Regions**



**Figure 16. The Quantity and Value Totals**

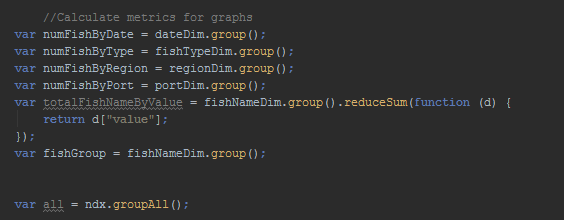


Pie Chart – Main Regional Ports

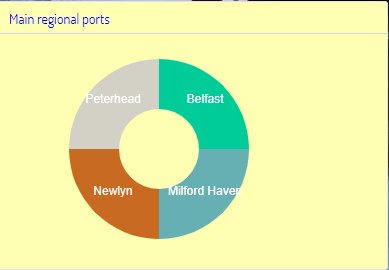
The main regional ports pie chart is there to highlight where the fish is caught in the individual UK regions. Selecting a port then presents the relevant UK region in the regional pie chart above it.

The metrics in the graph.js file allowed for this grouping and calculation of values to happen:

**Figure 17. Metrics in graph.js file**



**Figure 18. The Pie Chart – UK Ports**

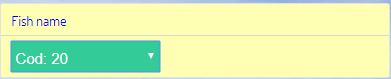


Select Menu – Fish Names and Row Chart – Fish Types

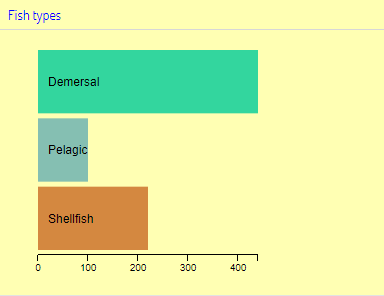
The fish name select menu was important feature to answer the project question of how well cod was doing. Once the user clicks on the name ‘cod’ under the list the quantity and values are shown highlighting that there has been no downward drop in the amount that the UK catches, in fact cod has one of the highest numbers.

The fish types row chart and information button are there for information purposes to provide further knowledge on what the UK is retrieving in terms of fish categories. Cod, haddock and crabs are among the most popular. Both cod and haddock are demersal fish while crab falls under the shellfish category.

**Figure 19. The Select Menu for fish names**



**Figure 20. The Row Chart – Fish Types**



# Conclusion

Using D3.js, DC.js and crossfilter libraries made the presentation and data calculation extremely easy to compute and present. They execute all the heavy lifting and allow users to focus on extracting answers from a large dataset.

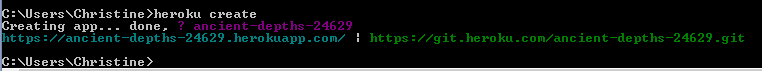
So with the help of these technologies used we can see that the statistics from the government show that there is no downward spiral of fish landings in the UK – quite the opposite, there is a healthy number of fish caught over a span of 5 years!

# Preparing the Project for Heroku deployment

Heroku new app setup

The first step was to open an admin command prompt and type **heroku create** in the command line. This creates a new Heroku application for use.

**Figure 21. Heroku app creation**



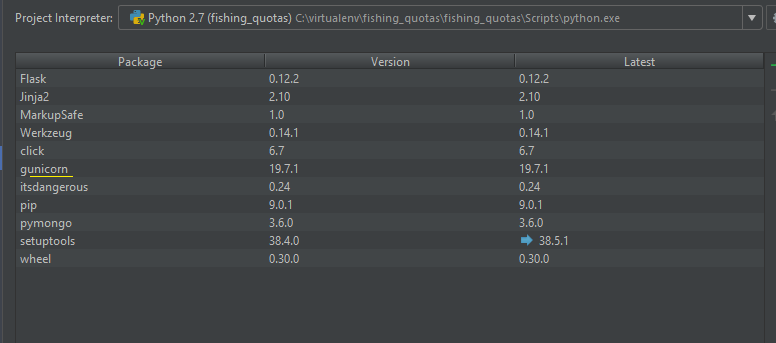
**Figure 22 Heroku new app**

Machine generated alternative text:
0 
Personal 
ancient-depths-24629 
Overview 
Free Dynos 
Resources 
Deploy 
Metrics 
Activity 
Access 
Settings 
Upgrade to Hobby... 
Web Dyno Autoscaling: Available now for Performance web dynos and Private Spaces. 
web gunicorn fishing_quotas: app 

Next gunicorn was installed. Gunicorn is used for running HTTP servers on UNIX based operating systems, such as Mac OS X and Linux distros.

Pycharm settings was then updated and gunicorn installed.

**Figure 23. Pycharm settings**



Requirements file

A requirements txt file was created to keep all the installed package versions.

The following commands were run:

* **Activate VM :** (fishing\_quotas) C:\virtualenv\fishing\_quotas\Scripts>activate
* **Run pip freeze --local > requirements.txt**

This has applied the following output dependencies to the requirements.txt file:

click==6.7

Flask==0.12.3

gunicorn==19.7.1

itsdangerous==0.24

Jinja2==2.10

MarkupSafe==1.0

pymongo==3.6.0

Werkzeug==0.14.1

Procfile

The Procfile is a file that’s used by Heroku to tell it what to do with the application once it’s been deployed.

 Therefore a new file called procfile was created and saved as a txt file.

A procfile.txt file was then created and added to the Pycharm project folder containing the following command: web: **gunicorn fishing\_quotas:app.**

Heroku will then execute this command once the procfile is detected.

Since Windows is running on the local machine another file was created namely the procfile called Procfile.windows containing the command: **web: python fishing\_quotas.py.**

Since gunicorn doesn’t have Windows support the server will run directly from the app.

To ensure Windows users run the window procfile the following command was initiated (after ensuring Mongo DB was running):

**heroku local -f Procfile.windows**

Heroku git

The Heroku git project deployment and the scaling (since this is a moderate sized project only 1 Heroku dyno was required) was then initiated by running git commands on the Heroku app.

**Figure 24. Heroku Git Deploy**

Machine generated alternative text:
heistineeChrisLenouo 
git branch — 
-all 
master 
heistineeChrisLenouo 
-pc 
-pc 
Ml NGW64 
Ml NGW64 
N/ P ycharmPro j e c t s/ F is hing_quo tas 
N/ P ycharmPro j e c t s/ F is hing_quo tas 
(master) 
(master) 
git push heroku master 
ounting objects: 4. done. 
elta compression using up to 2 threads. 
ompressing objects: Iøøz (4/4). done. 
pit ing 
otaI 4 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote: 
emote . 
objects: Iøøz (4/4). 974 bytes 
: 26.øø KiB/s. 
(delta 3). reused (delta O) 
Compressing source Files.. 
. done. 
Building source: 
Python app detected 
Installing pip 
Installing requirements with 
Discovering process types 
ProcF ile declares types web 
Compressing.. 
Done: 34.6M 
Launching... 
Released u4 
done. 
https : // . herokuapp.com/ deployed to 
Hero ku 
• UeriFying deploy... 
done. 
o https : //git . 
eaFø7cd. .2FIOF8b master master 
heistineeChrisLenouo—PC MINGW64 
N/ P ycharmPro j e c t s/ F is hing_quo tas 
(master) 

**Figure 25. Heroku Scaling**

Machine generated alternative text:
heistineeChrisLenouo 
-pc 
N/ P ycharmPro j e c t s/ F is hing_quo tas 
heroku ps : scale web=l 
cal ing dynos... 
done. 
web 
at I :Free 
now running 
heistineeChrisLenouo 
-pc 
N/ P ycharmPro j e c t s/ F is hing_quo tas 
(master) 
(master) 

Runtime.txt

Lastly a runtime.txt was created in Pycharm to tell Heroku what version of Python is running. Per below the version running on this PC is 2.7.15.

Therefore python-2.7.15 was added to the runtime.txt file.

**Figure 26. Python version**



mLab

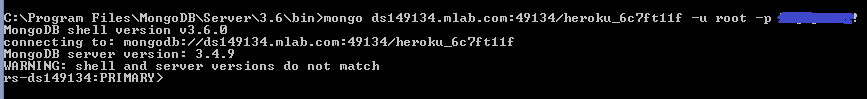
mLab was downloaded and used as an add-on to Heroku. mLab is a cloud database that allows the MongoDB to be hosted on the Web.

**Figure 27. Heroku mLab add on**

Machine generated alternative text:
Personal 
ancient-depths-24629 
Overview 
Free Dynos 
Resources 
Deploy 
Metrics 
Activity 
Access 
Settings 
Upgrade to Hobby... 
Open app 
Hide 
More C 
Leam More 
Web Dyno Autoscaling: Available now for Performance web dynos and Private Spaces. 
web gunicorn fishing_quotas: app 
Add-ons 
The addon mongolab has been installed. Check out the documentation in its Dev Center article to get started. 
Q Quickly add add-ons from Elements 
ml_ab MongoDB Mongodb 
Find more add-ons 
Sandbox (Free) 

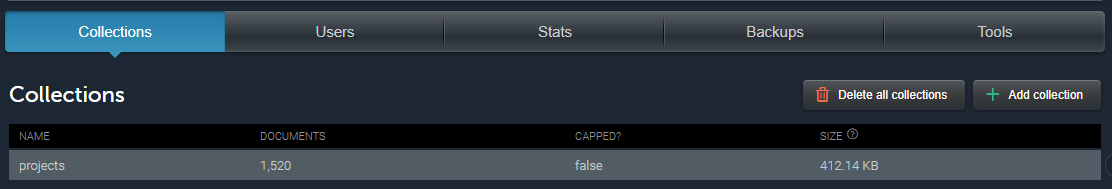
After logging into mLab, a new user (in addition to the mLab database created account user) was created and tested.

**Figure 28. mLab new user creation.**



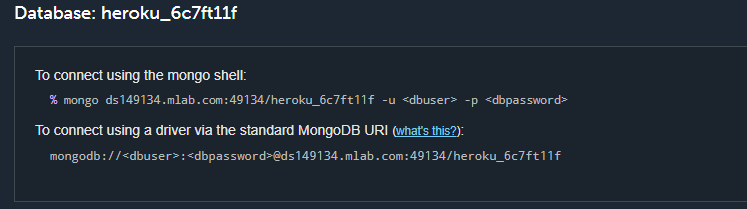
A new collection was then created (called projects) and the fishing quota data uploaded.

**Figure 29. mLab collection and records**

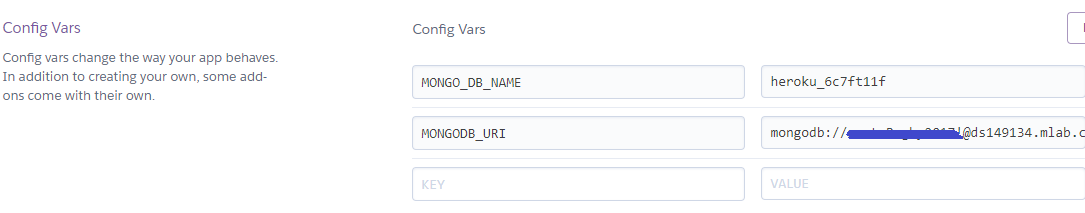


The last few steps included taking the URI path depicted on the mLab and mapping this together with the database name as the saved environment variables to the Heroku dashboard.

**Figure 30. mLab URI path**

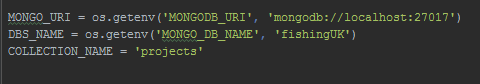


**Figure 31. Heroku Environment Variables**



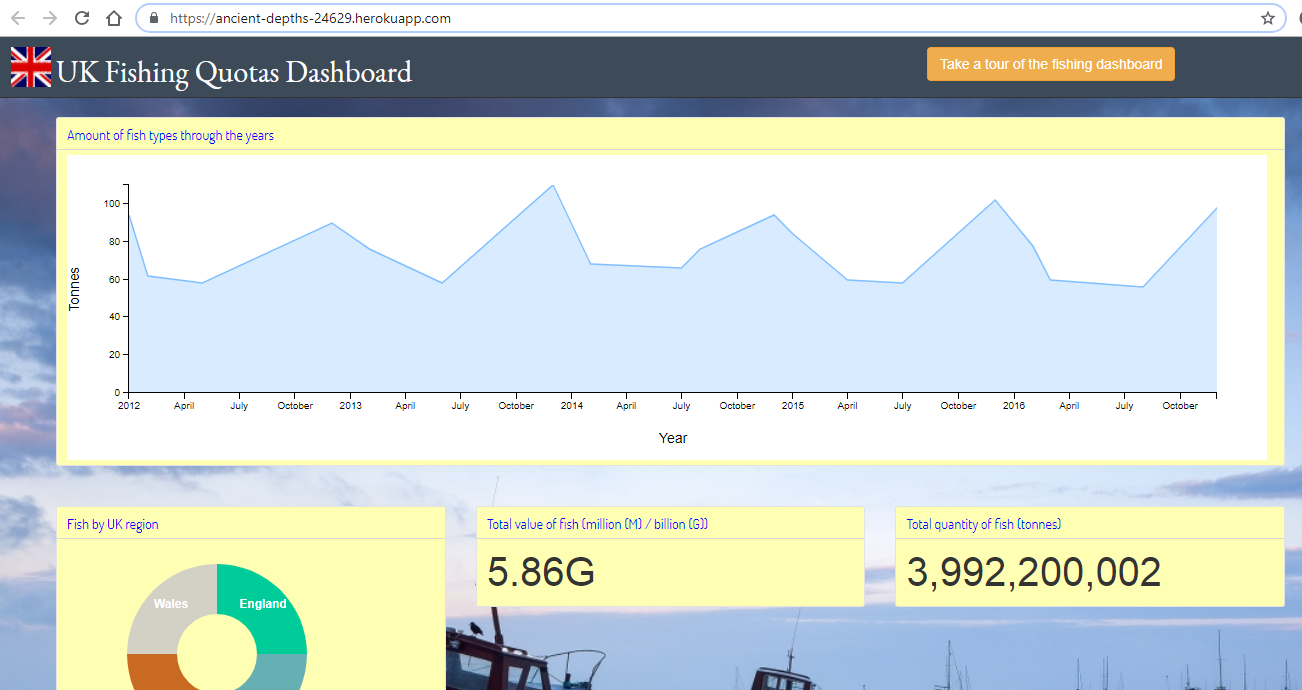
The last setup was to updated the fishing.py file on the Pycharm project. After deploying to Heroku the new environment variables will be used. If they are not available then the local host variables would be used instead.

**Figure 32. Fishing.py environment variables**



Once the “Heroku open” command was initiated (ensuring Heroku login was first executed) the Heroku application opened in the browser.

**Figure 33. Heroku Live Application**



# Appendix A: References

<http://www.gov.scot/Topics/marine/marine-environment/species/fish/pelagic>

<https://data.oecd.org/fish/fish-landings.htm>

<https://www.gov.uk>

<https://fullfact.org/europe/eu-pinching-our-fish/>

<http://www.telegraph.co.uk/comment/11305123/No-end-to-the-EUs-crazy-fishing-policy.html>

<http://bl.ocks.org/zanarmstrong/05c1e95bf7aa16c4768e>