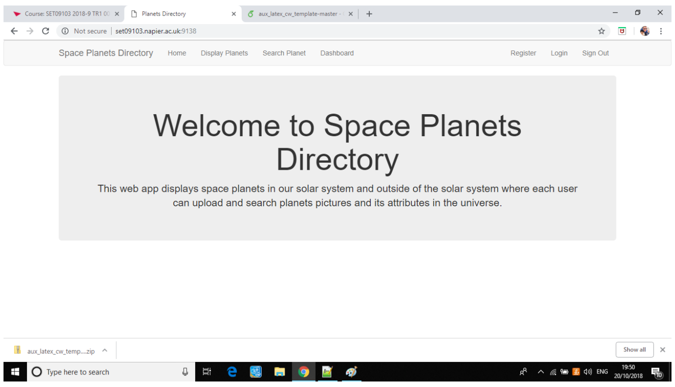
1 Introduction

The objective of the Advanced Web Technology Assignment is to design and create a online directory in a web app form which allow users to visit one or many routes. The online directory that I have created is called Space Planets Directory. It has a Space Planets theme where all visitors can see all hard coded planets and also uploaded planets on the web app and its attributes or search for the information of a certain planet by inputting the planet name in a input box and click submit. Users can also upload a planet but must sign up and log in to the web app to do so. I have used various of technologies to invent this web application such as vim, python flask and HTML templates. Vim was mainly used to create and edit python files and HTML templates. I have used python flask to implement the server side part of the app and have used various of HTML templates to implement the front-side of the app. I have also used css and JavaScript bootstraps to make my website more professional, user friendly and interactive.



Welcome to Space Planets Directory

This web app displays space planets in our solar system and outside of the solar system where each user can upload and search planets pictures and its attributes in the universe.

Figure 1: **Homepage** - The Space Planets Directory homepage

2 Design

This section describes all of the aspects of the design I have made for the web app.

2.1 Server Side

I have used Python Flask for all of the server side aspect of my app. The HTML is mainly created in HTML templates which has the HTML code that have generated the layout, navigation bar and text for each route of the online directory. These HTML templates is displayed on the web when is returned within a 'render template' built in function within each function in a python class called 'main.py'. I have developed the whole app on the module development server called 'set09103.napier.ac.uk' on putty in debug mode which have helped me to locate and fix bugs as soon as possible.

I have mainly used two resources to help me to create this web app. One is the Advanced Web Technologies workbook [1] and other resource I have used was a YouTube tutorial play list called 'Python Flask From Scratch' [2]. The workbook have taught me how to use sessions to manage user data between requests by storing small amounts of data in a cookie when every time the user logs in to the app so user can able to access the dashboard and upload planet route as long the user is still signed in. The workbook have also taught me how to render HTML templates within a function in my python class, redirect user to the dashboard every time user signs into the web app or redirect the user to the log in page when user is not logged in, error handling, responses and requests. The error handling is used to handle any errors within few of the functions within my web app such as the sign out function. Requests are used when a user clicks on the submit button which requests a response. The response is depending on what user inputted in the input boxes within login, search, register and upload planet page. The YouTube tutorial have taught me how to implement a basic blog css and JavaScript bootstrap within my app to make my website more attractive, user friendly, professional and interactive while each user navigates through my web app. I have used json to display all of the hard coded default planets in the display planets page. I have learn to implement this from a blog in the code handbook website. [3].

2.2 The HTML

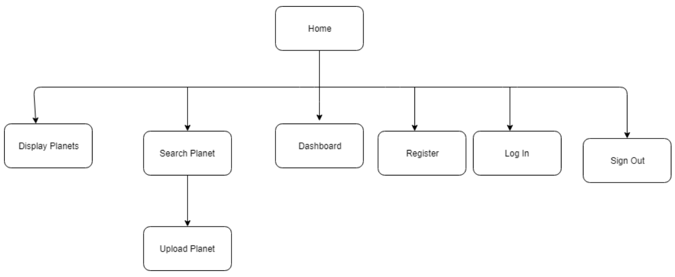


Figure 2: **HTML Structure** - HTML structure for the web app

I have used a navigation system for my web app to allow users to easily navigate through my web app.

Each page of the app uses HTML templates that is generated by using python Flask within if statements within functions in the 'main.py' class. Every page has a similar layout but different information to help users to not find my app complicated and remember what each function does in a more quick and efficient way as possible. The home page of my app is the most simple part of the app. The heading is on top left of the page and the homepage also contains a small easy to understand plain English description in large black font within a large grey rectangular box. Around the rectangular box is white background. All of the pages has a silver navigation bar top of the page and white background below it with black font title on left top. In pages such as searchPlanet, UploadPlanet, login and register, the pages contains labels, input boxes and a submit button. The information displayed is centered in the middle of the all pages in my web app.

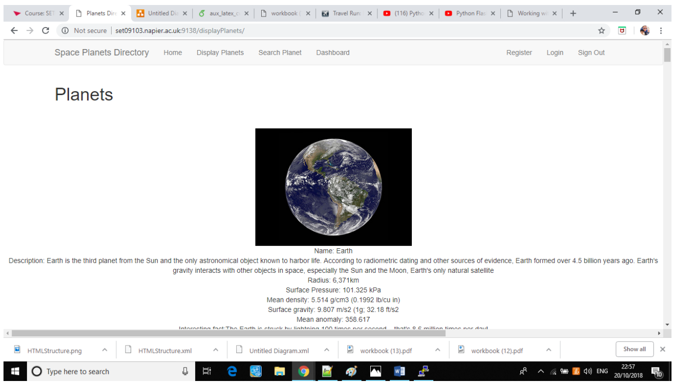
2.3 Python

I have mainly used python for all of the functionality throughout all pages within my web app. This includes opening a text file or json file, reading from a text file, displaying information on the screen from the json or text file and appending to a text file and storing each user data in cookies every time the user signs in by the use of sessions. A secret key is also generated for the password every time a user registers a account. I have used if statements and for loops in most of the functions so a response user only occurs if condition in the if statement is true. If the condition is false in a if statement, a error message is displayed on the screen.

2.4 CSS and Javascript

I have used CSS bootstrap that would allow the design of my web app to have a professional blog styled look. I have also used a JavaScript bootstrap that allows users to interact with the web app while navigating, searching, uploading, registering, signing in, signing out or just simply looking all of the planets that have been hard coded or uploaded to the web app.

The navigation bar that is present in all pages of the web app is very eye catching with its silver background and black font on tablet buttons. Every time a user hovers on a button, the button temporally changes colour. I have used ¡div align = "center"¿¡/div¿ to center align all information in all of the pages in the web app and also to center align the input boxes and labels in pages such as the login page.



Planets

Name: Earth
Description: Earth is the third planet from the Sun and the only astronomical object known to harbor life. According to radiometric dating and other sources of evidence, Earth formed over 4.5 billion years ago. Earth's gravity interacts with other objects in space, especially the Sun and the Moon, Earth's only natural satellite
Radius: 6,371km
Surface Pressure: 101.325 kPa
Mean density: 5.514 g/cm3 (0.1992 lb/cu in)
Surface gravity: 9.807 m/s2 (1g; 32.18 ft/s2
Mean anomaly: 358.617
Interesting fact:The Earth is struck by lightning 100 times per second – that's 8.6 million times per day!

Figure 3: **Display Planets** - A glimpse from my Display Planets page

3 Implementation

I have implemented this web app in a very user friendly methodology which is easy to understand and use. However, I have implemented some fascinating functions that would make any user hooked on to the site.

With the help of json and simple text files, Planets is able to be shown to the user, searched or even uploaded. Users can also register a account and log in with the help of reading usernames and passwords from or writing usernames, passwords and email addresses to a text file. The json file contains planets such as Earth, Mars and Mercury which i have hard coded myself. Each planet has same multiple of keys with different values. The keys for each planet are Name, Radius, Surface Pressure, Mean Density, Surface Gravity, Mean Anomaly and Interesting Fact. In the Display Planets page, I have implemented a function that reads from the json file and then load it. Using for loop, I have able to append each key value for each planet to the result variable and then return it so the hard coded planets can be displayed on the screen when user clicks on Display Planets button. The uploaded planets are written to the text file when the user submits a planet in the Upload planets page which can only be accessed if user is logged in. In display function of the app, the same text file is read from and turned into a list by using split function to split each text between commas. A for loop is established to loop through this list and append all values for each uploaded planet to a variable called result2. These uploaded planets are displayed on screen below the hard coded planets on the Display Planets page.

I have also implemented a search page which can be accessed by all users whether they are signed in to the app or not. The user can enter a planet name in the input box and then click the submit button. If the planet name available, the planet name and along with the planet image and other attributes of the planet is displayed on the screen. In the back end part, this works within a search function which renders a HTML template depending on what if statement condition is true. Within another if statement, the function reads from a json and text file and checks if the inputted text is equal to any value in json or text file. if it is, the planet is displayed. If is not, a error message is displayed on the screen.

The register page allows a user to make a account by inputting a username, password and email address in the respected input boxes next to its labels and then click submit. If the username and email address are both not in a text file called 'accounts.txt', the account have successfully been made. Each password is encrypted by a secret key stored in the 'app.secretKey' variable. If the condition not met, error message is displayed on the screen. This was mainly implemented by a register() function in the 'main.py' class where after the if statements are met, the username, email address and password is written to a text file called 'accounts.txt' separated between commas. A HTML template is returned which has the normal layout with the success message in the centre of the page.

The login page allows users to sign in by inputting their username and password in the respected input box next to its labels and then click submit. If the username and password are both in the 'accounts.txt' file, the user have successfully logged in. The user data is stored in the cookie and removed when user signs out which is implemented by the use of sessions. This is implemented in the login() function within the 'main.py' class. The 'accounts.txt' is read from and the text between commas is split up by the split function and is then stored in a list. I have then implemented a for loop to loop through the list and check if the inputted password and username is in the list by using if statements within the for loop. If condition is correct, user signed in and its data is stored in cookies by the implemented session. The user is also redirected to the dashboard where there is a message that says 'welcome' username. The username is displayed as I have used Session.username in the dashboard HTML template. The dashboard has a Upload planet button which allows the user to click on the upload planet button and then the user is directed to the Upload Planet page. When user clicks on the sign out button, the user is signed out and the user data is popped off from the session. This works only if the user is already signed in. If not, error message is displayed. I have used various of if statements in the signout() function in my 'main.py' class.

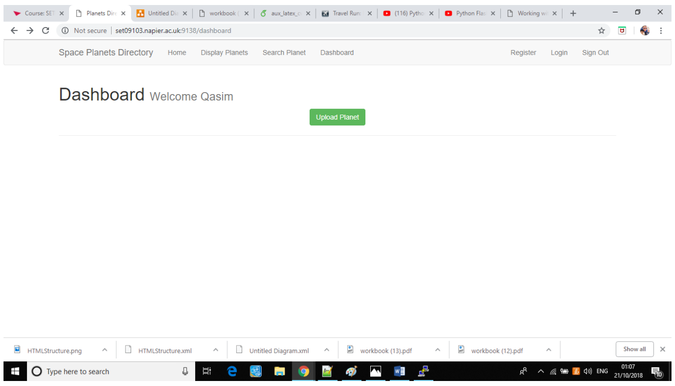


Figure 4: **Dashboard Screenshot** - Screenshot of Dashboard while I am signed in as a user called Qasim

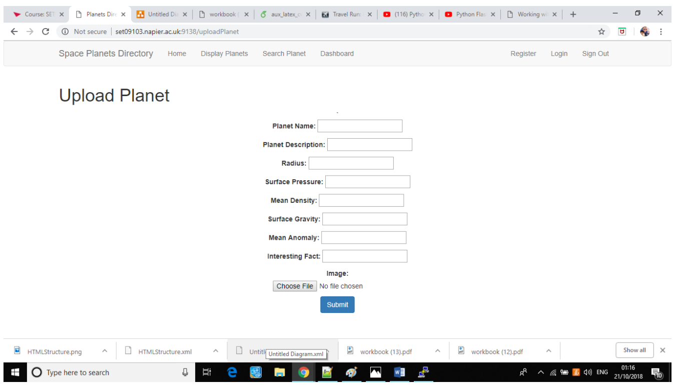


Figure 5: **Upload Planet page** - Screenshot of Upload Planet page

4 Implementation Evaluation

4.1 Does it meet the Specification?

There is no stress for a user to navigate the whole Space Planets Directory. The user can either click on the Search Planet button to search for information for a certain Planet or click on the Display Planets page to see all Planets and its attributes on one page. If the user find no interesting Planet, the user can upload a planet within the Upload Planet page as long the user is signed in. There is multiple of routes and the application is very interactive and user friendly.

I have used a blog style format for all of the pages within the web app. White background, silver navigation bar and black bold font. I have implemented my web app in a blog style app despite the app being a online directory because I have visited various of blog websites and realized that these websites are most easy to use, navigate and websites that you would visit multiple of times. This layout was successfully generated by inheriting the 'layout.html' template in all of my templates by using extends keyword. The 'layout.html' template contains the javascript and css bootstrap which is located in the bootstrapcdn website and also extends to the '$_n avbar.html' template. The'_n avbar.html' templateimplementsthesilvernavigationbarthathastabletbuttonswithblackfont. Inmy$

The web server has the hard coded planets in the json file and uploaded planets in a text file along with its attributes. This is very useful as it can help each user to search a certain planet from the planet name in a search bar with no struggle.

I had to deal with many challenges while developing this web app. The hardest challenge was looping through the text file and json file to read or write from these files whether it be when planets are displayed on screen, plants being uploaded, user accounts being registered and user signing in. It was really challenging. This was really challenging as sometime it was hard for the function to find username or password in the 'accounts.txt' file within if statement within the for loop. This was difficult because there was some glitches while the register function writes the username and password to the 'accounts.txt' every time a user registers a account so I had to edit these if statements many times before I have managed to solve the problem.

I had similar problem within search function. The problem was that every time a user inputs a planet name in the search bar, the search function struggles finding the planet name in the planets text file as there was glitches while the uploadPlanet() function writes a planet attributes to the planets text file. Fortunately, I have managed to solve this problem after many logging, debugging and editing the code.

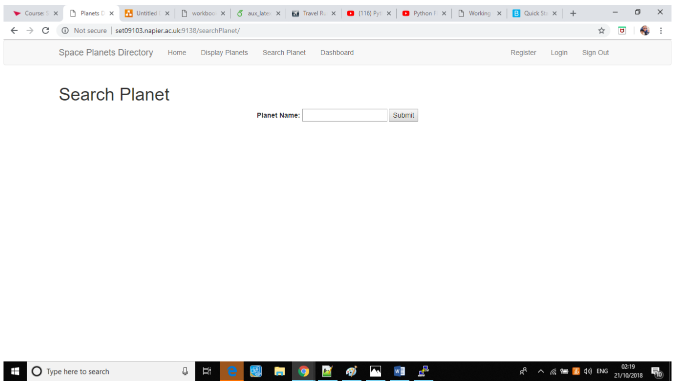


Figure 6: **Search Planet Page** - Screenshot of Search Planet page

4.2 Potential Enhancement

I like to add a Edit and Delete function for each Planet that a certain user have uploaded to the server. However, this is bit difficult to implement without using SQL which is unfortunately not allowed in this assignment.

I also would like to implement a report function where a user can report if there is any faults in any of the planets such as duplicate names, spelling mistakes, spam and ect to a admin of the web app. However, is a big challenge to implement without using a database.

I also would like to add a message function which allows each user to communicate with each other. I also would like to add a forum function that would allow users to create topics and initiate space exploration related discussions with other users.

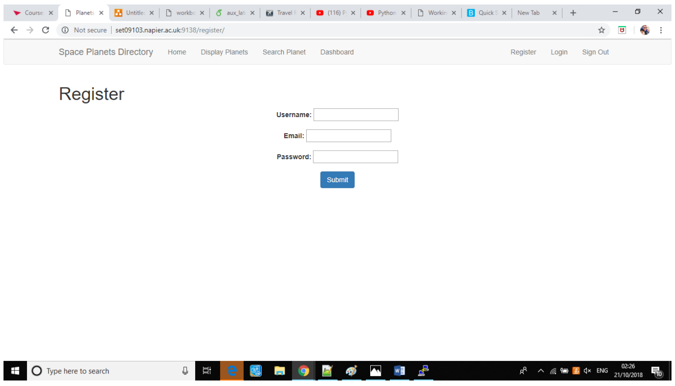


Figure 7: **Register Page** - Screenshot of Register page

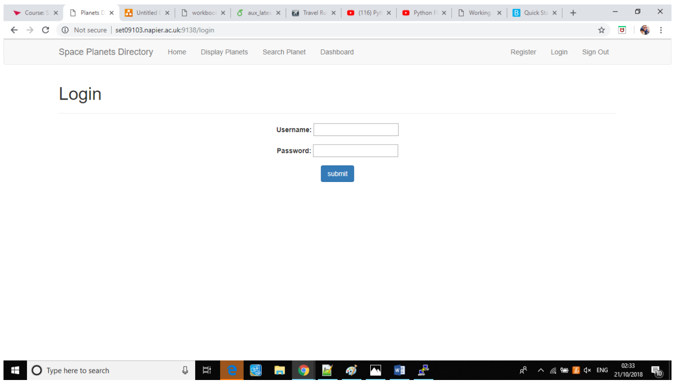


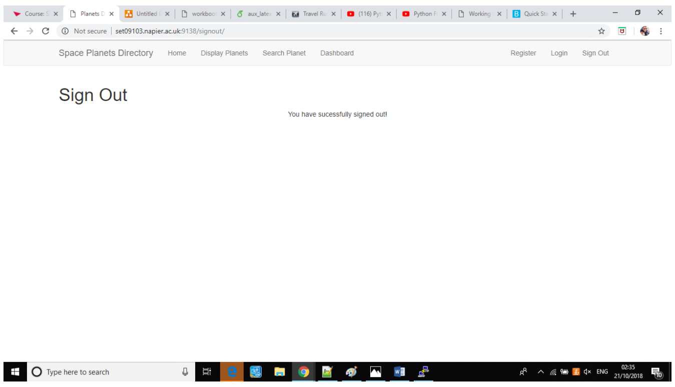
Figure 8: **Login Page** - Screenshot of Login Page

5 Personal Evaluation

The functions that I implemented in this web app has been a huge achievement. Despite developing websites before, I have never used python Flask before so It had been a amazing learning experience while developing this app. For most of the features of the app, I had to research on how to use Python Flask to implement these features Flask such as researching how to implement a CSS and JavaScript bootstrap. The procedure I used before I decided what CSS and JavaScript Bootstrap to implement in my web app is that I have read few user interface design articles and then decided professional blog style layout is most suitable for my app as is user friendly, easy to use and would encourage visitors to visit my web app multiple of times and invite their friends or family members to do so as well. However, I do not intend to put this web app live unless I make some enhancements such as like what I have mentioned before such as delete, edit, report, message and forum feature. I would also like to tweak the design of the web app a bit to make it more unique such as make the navigation bar blue and background black with a space theme logo in left part of the navigation bar because during my research, I have found out that users are mostly addictive to websites with blue and black combination such as Linked-in. I was not able to implement this due to my little experience of user interface design which mean I was fearful of generating debugs that will be difficult to locate and fix.

The biggest challenge I faced was the uploadPlanet() function. This was a big challenge because initially, I wanted to read the input values for each planet that been uploaded by a user to a json file. I have attempted to do this few times but It did not work work so I researched on how to write these uploaded planet values to a text file within if statement in a for loop. My first attempt to write code that would read these uploaded planets values to a text file in the uploadPlanet() function had led to the searchPlanet() function failing to find planet name that was inputted by user in search bar in the text file as as there was extra commas that had been read to the text file by uploadPlanet() function. Fortunately, after few debugs and editing the code, planet values managed to be written to the text file every time a user uploads a planet. I have managed to use the same procedure to make the register() function to write the inputted username, password and email address from the respected input boxes to a text file called 'accounts.txt' file every time user click on submit button and successfully creates a account.

Overall, i think this space planets online directory is efficient and unique as there is not much online directory's that has a Space Exploration theme. Despite, being a challenge of writing to a text file or reading from a text file or reading from a json file, I have managed to make this web app a bug free web app which makes me really proud that I am the founder and developer of a space planets online directory web app. This web app is great for users who likes to learn about various of planets that he or she never have heard of or maybe learn about extra information about a certain planet that he or she are interested about. The app is also suitable for space geeks who likes to upload planets that they have knowledge about to teach my web app visitors around the globe.



You have sucessfully signed out!

Figure 9: **Sign Out Page** - Screenshot of user signing out

References

[1] S. Wells, "Advanced web technologies workbook,"

[2] T. Media, "Python flask from scratch,"

[3] C. Handbook, "Working with json in python flask,"

[4] Bootstrapcdn, "Bootstrapcdn,"