* A post reflecting on what you’ve enjoyed studying this year
* A post looking to the future that discusses what you’re looking forward to learning about in the upcoming academic year.
* A post discussing how your site is structured and in your own words a description of how Django works.

My favourite modules this year were:

1. AI (not the robotics side)
2. LC Data Structure
3. Logic and Computation

Logic and Computation

While I found this module to be challenging, I found it incredibly fascinating how day to day logic can be distilled down into a completely new language with rules that allow for complex manipulations into completely different (but fundamentally the same) rules. I also found it very insightful as it explained why my conditional statements in programming worked (and also why they didn’t). While the problems were difficult, it was very satisfying to work my way through them and slowly make progress in each question.

LC Data Structure

A relatively straight forward module however very fascinating to discover new (and old) data structures and really break down the advantages and disadvantages of each one, including the situations were its recommended. As I’m particularly interested in data science and optimisation, this module was particularly useful for me. The tree data structure was a new concept to me which I found most interesting as it can be incredibly powerful in certain situations. Doubly linked trees is something I have experimented with following our introduction to the tree, it is very fascinating to see how efficient this kind of data structure can operate.

AI

I thoroughly enjoyed learning about different optimisation algorithms in this module, along with the various methods to avoid local minimas. We learnt about a huge number of different algorithms each of which had certain advantages and disadvantages for each use case. Learning about machine learning was also interesting, specifically q tables and the various formulas that populate them over multiple iterations.

What I’m looking forward to learning in the upcoming year

I am very excited to learn more about C and C++ in systems programming. It is a fascinating language as it is the building block of many great languages. Being such a low-level language, the option to manage memory is a new concept to me. I’ve attempted to learn C++ in my spare time to some success, it can be used just like any other programming language like Java and C sharp - but the memory side of it is difficult to understand completely without proper guidance as I still have a lot of unknowns particularly when it comes to the stack and the heap.

Security and networks is another module that I’m interested in. I’ve used several different network protocols to move data in the past, and while I know the basics behind security, learning about vulnerabilities and good practices will make me a better programmer and designer.

I’m also looking forward to the group project next year. I have done some group work before but never really got the full experience of it. So I’m hoping with one large common goal to work towards that the group project will be a lot of fun.

How my site is structured/how Django works

How Django works:

Django is a web framework for creating websites in python. It is similar to many other web frameworks such as Node js and React. Django is a http server wrapped with a python API to form a flexible web framework. It also comes with built in admin and database support. The development process of writing a website is split into client and server code. The server decides which web pages to send to the client via the url entered which is mapped to a specific “view”. Each view usually has a different html page mapped to it which is what the client sees. Dynamic html can also be generated via the python interface within the html files. Python can also be used to help generate the html page, which is powerful as data can be transferred from the server when a page is loaded and used with python to help shape how the page looks. Navigating the website is simply a series of GET requests from the client computer. The user can write custom responses to each of these get requests, including parsing the data from the request and using it to decide which response to send back to the client.

How my site works:

It is a simple blog website demonstrating how data can be saved and stored in a database and then retrieved. Basic security has been implemented limiting the ability to add new blogs into it unless you are a admin. This is achieved via the “user.is\_authenticated” code in the html page. If the user is authenticated then a button will generate that when clicked will redirect the user to the post\_edit view. There is a security vulnerability however, because there is no authentication no the server side of the GET request, i.e. while the client code (the html page) will check to make sure your authenticated before generating the button, the server has no authentication of the request so any GET request it receives will be executed.

I used bootstrap to mainly decorate the page’s css. I also have my own stylesheet for more specific customisations.