Nini Nguyen
Biocomputing II
MSc Bioinformatics with Systems Biology
Birkbeck, University of London
2020

Group Project - Reflective essay
bbk_Chromosome6 - Florence Lai, Oliver Cant, Maham Ahmad, Nini Nguyen

1. Approach to the project

A) Interaction with the team

For the duration of the project, all members worked cohesively as a team. When assigning roles in the group we tactically selected members to the different tiers based off of what they were most comfortable with handling: Python/SQL - database layer, Python - business layer, HTML/CSS - CGI layer. We exchanged phone numbers and formed a group chat, so that we could contact each other as a group more easily.

B) Overall project requirements

The project required us to construct a genome browser that would contain the Genbank entries for our chosen chromosome (6). We had several discussions to exactly determine what we thought the project required us to carry out. The main aspects were:

- A gene summary table to display the output of all entries and corresponding information (gene ID, accession no., protein product, location, CDS and protein/DNA sequences)
- A search bar for the user to search for individual/related entries
- A hyperlink on the accession codes so that it can display the further required information such as restriction enzyme sites on the DNA sequences.

C) Requirements of my contribution

As the fourth member of the team I was assigned the role to carry out code/software testing; which required me to be well adapted to all layers of the browser. As a software tester, I was aware that I had to test each function of the code to ensure it worked as expected and that the correct output was achieved. If there were any errors/failed tests, I was to be able to find and resolve the issue with the author of the code. Code testing was either to be done after the code had been written or during the developmental process.

2. Performance of the development cycle

We first tried to draw a schematic diagram of the browser; with the required components of the project and how we thought results should be returned to the user. It was agreed that the genome browser will display the gene summary list of all chromosome six genes including details for each gene: gene ID, accession number, location and protein product. From this table of genes, you will be able to click on to a Genbank accession number where the user will be transferred to another page that will display all the gene information for that particular gene. This page will display all the information for that selected accession number and its related gene record details including codon usage frequency and the choice from a selection of restriction enzymes that will be located with stars marked on the DNA sequence. Alternatively, the user will also be able to search for an entry of their choice by inputting either: a Genbank accession code, gene ID, protein product or a chromosomal location map coordinate. All related gene entries will then be displayed based off of the input details from the user.

3. The development process

A Github account was created so that all members of the group could upload their files for everyone to be able to use interchangeably for their code if required or necessary.
We agreed on what variables we needed to have and how the data should be formatted as an output and what type of input the user will have to type in for the browser.
During the development of the browser, all face-to-face teaching had stopped due to the coronavirus pandemic and therefore we were unable to have any further group project meetings. A video call meeting was then planned so that we could finalise everybody's responsibilities for their assigned layers. Over the next few weeks each member individually worked on their layers, occasionally handling any queries with everyone in the group chat if necessary.

4. Code testing

Unit testing was chosen as the method to test the code as it is a useful method when wanting to determine what aspect of the code is wrong and how to resolve the issue. Each unit testing module should run independently, where an expected outcome is asserted during the testing and then compared with the actual functions from the project code.
Due to the nature of the developmental process, I essentially had to wait until group members had completed their code for their layers so that I could test their functions. Compared to test driven development where the codes are written whilst tested, unit testing required codes to be able to run and work independently. Hence, the need to use the *setUp* and *tearDown* functions to create dummy entry codes, and to place variables for the records created when the functions are tested.
For the blapi (business layer api) I had a lot of help from Florence as she tested her own code since I was struggling to complete the database coding. Doctest for python was therefore implemented to test the blapi, which passed all tests for the getEntry and getAllEntries function. I was given the shell output for the tests on the code from Florence and subsequently wrote the docstrings documenting the test verification.

5. Known issues

We had a few minor bugs in some parts of the code that were dealt with and resolved.
The database API testing presented assertion and name errors whilst I attempted to unit test the code, I was aware that unit tests were supposed to be able to run independently and therefore had problems executing the dbapi successfully because some areas of the code used other imported modules and databases.

6. Problems and solutions

As a whole, the team worked very well as a group and the final product of everyone's efforts resulted in a working browser, despite the lack of testing on my behalf.
I found the software testing challenging, however Florence very kindly offered to help me complete the testing for her layer of the project, this helped ensure that the blapi code was working as it should.
Testing the CGI layer presented some problems as I was aware that unit testing required the testing of functional units of code, whereas the front end layer didn't have any. Alternatively, Florence attempted to test the code using doctest; however the validity isn't too reliable as the test will always pass successfully. Therefore, the final option we had was to directly check the information from the website and verify the correct output.

7. Alternative strategies

Alternative strategies I would have used for my contribution to the project:

- Perhaps begin with test driven development (TDD) as a team to understand better how the code is formulated, enabling me to better understand the output of functions more clearer for unit testing
- Ideally to test several different forms of input when testing functions, and asserting false values with errors to ensure that if there is any incorrect input from the browser, the programme will be able to identify this
- To attempt using several different testing platforms/modules instead of trying to apply one method of testing for all aspects of the browser.

8. Personal insights

Overall, I am very pleased with the enthusiasm, hard work and efforts all members contributed to the team. The project was substantially challenging, however we demonstrated well the importance of team work and how platforms like Github allow us to easily share code with team members to allow us to work on the same project. It helped me gain insight and allowed me to have a better understanding of web development and the collation of different skills that need to be applied to put together a working project such as the genome browser.