Sprint 1 Report

Introduction

The task for sprint 1 was to create a command line utility for searching University of Guelph courses offline. For this sprint, the team split up the project requirements into a number of smaller tasks, of which are documented below into their respective sections. As an additional consideration, the team sought to develop solutions without the use of external libraries.

Researching VBA

A 30 minute session by Tinson to research potential solutions involving VBA and how it may help this sprint task resulted in determining that VBA appears to be irrelevant to our task. Tinson notes that VBA could be used to remove HTML in cells in Excel, however the team's solution did not require Excel whatsoever.

Converting HTML to JSON

The members responsible for this task are Eric and Tinson. The two members both spent an hour researching solutions. Eric took to investigating solutions utilising other libraries while Tinson focused on a regex solution. After concluding research, the two members took to their own to experiment and develop a parser following what they've learned. Eric focused on a parser using BeautifulSoup while Tinson worked on a parser using the re library. On the morning of September 14, an announcement was made by the course instructor against the use of BeautifulSoup. Following this announcement, the decision to proceed with the regex parser was made. Eric and Tinson later connected to further refine the regex parser for the remainder of the team to use.

Python for Reading and Returning Text

The members responsible for this task are Saarthi and Rithik. It was determined that Saarthi would work on functions for reading text while Rithik would implement functions for returning text. Saarthi spent a total of 8 hours on the following tasks: reviewing Python syntax, learning how to read JSON into Python, creating functions to read JSON files, search user queries, and display results, implementing a CLI loop for parsing user input that calls search/filter/exit functions, writing skeleton functions for filtering queries, adding logic to ignore the "*" character in user input.

Tinson implemented the base functions for displaying text and assisted in the formatting of certain string elements while Rithik built on these functions to include a scrollable page view to allow for viewing 5 results at a time to ensure none gets cut off. Additionally, Rithik also implemented colour within the results by using yellow text for courses that were still open and had capacity available while using red text for courses that were closed and at full capacity. This allowed for easier readability and navigation.

Implementing Flags

The members responsible for this task were Luka and Michael. Both members were tasked with the implementation of both the filtering and sorting on the courses depending on the output that the user wants to see. The filtering was mainly done through the implementation of flags which would be inputted by the user and applied to the list of courses to only display the courses that match the filtering options that are provided by the user. Rithik also stepped in for this task to implement a multi-filter selection feature which allows the user to filter by numerous different values such as filtering by the status of the course along with the number of credits the course is weighed at and only courses that match these filters will be displayed to the user. Additionally, the user could also use the same filter twice. For example, by using the name filter twice, the user can search for all 'CIS' courses that also include the keyword 'Information' in the name. The user is also given the option to sort the list of courses based on their preferences. The user can either have the courses unsorted, sorted ascendingly or sorted descendingly based on the lexicographical values of the course name.

Testing

Ryan is the member assigned to testing. He was tasked with testing the parser and course search programs to ensure that they were bug free. Alongside this, an early attempt at creating an automated testing script was conducted, with a small file of 4 tests being created to test the html parser. In terms of manual testing, very few bugs were discovered, with extensive testing done using both Windows and Linux operating systems to ensure cross compatibility of the programs. Any bugs found were quickly fixed and the patches subsequently pushed to the sprint 1 repository.

General Usage Guide

The parser.py file must be ran first (CMD: "python parser.py"), with the "Section Selection Results WebAdvisor University of Guelph.html" in the same directory as the parser file. This will create a courses.json file. This file is not for the user, but for the command line interface. The CLI.py file can now be ran (CMD: "python CLI.py"), as long as it is in the same directory as courses.json, and it will read the JSON file and allow the user to perform searches, filtering, and sorting on the list of courses via a readable and easy to use command line interface.