**search-re-search**

**Outline**

**2XB3 Final Project**

**Group 8:**

**Donisius Wigie**

**Michael Ilao**

**Rohit Saily**

**Objective**

Provide a flexible yet rigorous means to find academic papers that are the most useful in relation to some idea. This includes:

1. Allowing the user to specify their ideas with simple input.
2. Search results are ordered and presented for convenience of the user by:
   * Ordering by quality, determined by some scoring algorithm.
   * Papers with ideas that do not overlap are presented towards the beginning to provide the user with a diverse set of resources.
   * Allowing the user to specify precise filters to remove papers with properties they definitely do not want and influence the scoring algorithm.
3. Building interactive graphs that show how papers are connected based on citations, so the user can see the history of ideas in the paper and find related papers with ease. The interactivity involves:
   * Nodes of the graph will show basic information, upon clicking the user can see additional information.

**Tools**

The project will be programmed using Java and the Eclipse IDE. In addition to this, we plan on making use of various technologies and tools (conceptual and actual) that will allow us to complete this project efficiently.

* **Github:** allows us to collaborate on the same project easily.
* **Google’s Java Code Style Guide**
* **JavaDoc**
* **JUnit Testing**
* **Swing**: Java’s library for programmatically creating UI.
* **Model-View-Controller (MVC):** The design paradigm which we will base the structure of the entire project on.

**Staffing**

Due to the scale of the project and the number of group members, each member will take on a combination of roles and the work will be split by the different components of the code; however, each person will be assigned specific roles associated with what they are ultimately responsible for and will be required to revise that aspect of the project towards the end. The specialized roles are listed below:

|  |  |  |
| --- | --- | --- |
| **Person** | **Roles** | **Responsibilities** |
| Donisius Wigie | Documentation | In code comments for the final program and JavaDoc creation. |
|  | Programmer - Controller | Produce the controller aspect of the MVC structure as Java classes. |
|  | Researcher | Find libraries, standard library functions, or algorithms that can be used to fulfill objectives programatically. |
|  |  |  |
| Michael Ilao | Log Admin | Log meetings, log project progress and changes. |
|  | Programmer - Model | Produce the model aspect of the MVC structure as Java classes. |
|  | Designer | Determine the classes needed and the API for each class. |
|  |  |  |
| Rohit Saily | Project Leader | Manage the project, ensure all milestones are completed, and produce the prototype. |
|  | Programmer - View | Produce the view aspect of the MVC structure as Java classes. |
|  | Quality Assurance | Produce JUnit tests to validate that API works as intended for all classes. |
|  | Exceptions and Exception Handling | Create and implement exceptions as well as determine how exceptions should be handled and create the code for doing so. |

**Project Plan**

|  |  |
| --- | --- |
| Date | Deliverable |
| Week 1 | Begin designing:   |  |  | | --- | --- | | Michael | * Create a log containing beginning times/end times for meetings, progress made in the project, changes made in the project. * Create rough requirements for modules which may be used in the final program. * Come up with modules which will be used in the program. | | Rohit | * Assign tasks and roles to each team member and divide tasks evenly. * Create dates for milestones to be completed and a plan for each team member to meet the deadlines. * Come up with modules which will be used in the program. | | Donisius | * Research tools and libraries which may be useful for the project. * Come up with a way to store or parse large json files in a time/space efficient manner. * Come up with modules which will be used in the program. | |
| Week 2 | * Revise the modules proposed by each team member and remove modules to avoid junk and confusion and add/modify modules which are missing or need refining. * Revise the rough requirements created in the past week and come up with a requirements document which will be the plan for the implementation of the modules. * Think of a way to incorporate the modules into the final program to work together. * Begin implementation.  |  |  | | --- | --- | | Michael | * Log the meetings, results from the meetings, and design changes to the project. * Think of ways the modules may be improved. | | Rohit | * Make sure the assigned jobs meet the deadlines, and ensure that all milestones are completed to this point. * Think of Junit tests as code is written. | | Donisius | * Create the javadocs comments as code is written. * Continue researching new tools/algorithms. | |
| Week 3 | * Process the information in the json file. * Begin working on the scoring algorithm which will order the papers by quality. * Begin working on the interactive graphs using the citation information in the json files. * Either have a way to search through the json files efficiently or have a way of storing the information so that searching for the papers will not take so long. * Finish a citation generator using information from the json files.  |  |  | | --- | --- | | Michael | * Log the meetings, results from the meetings, and design changes to the project. * Think of ways the modules may be improved. | | Rohit | * Make sure the assigned jobs meet the deadlines, and ensure that all milestones are completed to this point. * Think of Junit tests as code is written. | | Donisius | * Create the javadocs comments as code is written. * Continue researching new tools/algorithms. | |
| Week 4 | * Finish up a rough working prototype that contains the scoring and interactive graph features.  |  |  | | --- | --- | | Michael | * Log the meetings, results from the meetings, and design changes to the project. * Think of ways the modules may be improved. | | Rohit | * Make sure the assigned jobs meet the deadlines, and ensure that all milestones are completed to this point. * Think of Junit tests as code is written. | | Donisius | * Create the javadocs comments as code is written. * Continue researching new tools/algorithms. | |
| Week 5 | * Improve upon areas which either do not work or do not work in a satisfiable manner. * Make the rough working prototype more efficient in terms of time and space. * Make the scoring feature suggest papers more accurately.  |  |  | | --- | --- | | Michael | * Log the meetings, results from the meetings, and design changes to the project. * Think of ways the modules may be improved. | | Rohit | * Make sure the assigned jobs meet the deadlines, and ensure that all milestones are completed to this point. * Think of Junit tests as code is written. | | Donisius | * Create the javadocs comments as code is written. * Continue researching new tools/algorithms. | |