Team N

Title: Wikipedia Article Project (WAP)

Members:

Ruthvik Kodati	2019101035
Aman Goyal	2019101097
Adwait Raste	2019111027
C. S. Ramakrishna Tejasvi	2019114005
Keshav Bajaj	2019115010

Idea:

We plan on creating a single-player **web game**. The game will have 5 levels. Each level will correspond to a particular **degree of separation**. The easiest level will correspond to two articles having a degree of separation equal to 2. The last level will correspond to the articles having a degree of separation greater than or equal to six.
In every level, the user will be given two **Wikipedia articles** and his/her job would be to navigate their way from one given article to the other whilst ensuring that they reach the destination in the least possible number of jumps. A **jump** is defined as choosing a linked Wikipedia article from the current article at which they are located. For every level, the user has a base score equal to the **degree of separation * 500**. After the number of jumps exceeds the degree of separation, the user will **lose 100 points** for each jump till he/she hits 0. When they hit zero, the game resets. The final score will be the sum of scores from each level.
There will be a leaderboard where the top participants' final scores will be displayed.

Roles:

- **Frontend**(Designing the UI for the game and connecting the backend.)
    - Will be done by Keshav, Adwait, and Ruthvik
- **Backend**(Creating the storage and implementing the algorithms.)
    - Will be done by Adwait and Keshav
- **Scraping** and **Algorithm Implementation**
    - Will be done by Aman and Tejasvi

This is not set in stone and everyone will contribute to all parts but the majority of the tasks in a particular category will be done by the people mentioned.

Possible Entries:

Our daily tasks would be to first learn the frameworks mentioned below, the implementation or choosing the algorithm, or coming up with a design for the web app. This would be determined by the role that each team member has to play. We also plan on holding calls twice a week to ensure that we are all on the same page and are contributing to the goal daily. We will each be pushing a markdown file with our contributions to the GitHub repository every day.

Frontend Libraries

- Backend
- Bootstrap
- Node JS
- Express JS
- Javascript

Web Scraping - How to detect links from a particular Wikipedia article and store them
We have to think of a metric space to store the articles and the degree of separation between two articles will be the distance between the articles

We have to figure out an **efficient algorithm for detecting the minimum degree of separation** between two articles and also an efficient way to store the articles

We will be using a MERN stack or replace MongoDB with Firebase depending on feasibility.

Backend Server

- MongoDB or Firebase

Subtasks:

- Design the main UI and interface of the web app.
- Learn the frontend/backend libraries and technologies which are required to implement the frontend and backend.
- Research and learn new algorithms and data structures that will be required for the project.
- Find methods to scrape Wikipedia webpages and extract the required information.
- Combine the work and link the frontend to the backend into a single web app.

Timeline:

PHASE	DURATION	TASKS
Pre-Development Period	September 29th - September 30th	• Create the proposal. • Assign individual work. • Consider a possible framework. • Confirm with the Teaching Assistant • Find more resources.
PHASE	**DURATION**	**TASKS**
Week One	October 1st - October 7th	• Create a GitHub Repository. • Finalise the framework to use and confirm with the TA. • Confirm the algorithm to be used with the TA. • Confirm weekly progress with the TA.
Week Two	October 8th - October 14th	• Figure out the methods of scraping web pages. • Create a Figma prototype for the web app. • Learn the required frameworks. • Confirm weekly progress with the TA.
Week Three	October 15th - October 21st	• Scrape the articles. • Plan the domain and consult the TA • Setup the backend server. • Create a dummy webpage. • Confirm weekly progress with the TA.
Week Four	October 22nd - October 28th	• Start working on the algorithm. • Populate the database with the scraped articles and

		links. • Create a dummy leaderboard. • Confirm weekly progress with the TA.
Deliverable 1: • Completely populated database. • Working leaderboard. • 50% of the algorithm implemented.		
Week Five	October 29th - November 4th	• Finish the algorithm completely. • Create functions to implement the queries for the web app. • Link the frontend to the backend. • Confirm weekly progress with the TA.
Week Six	November 5th - November 11th	• Check the database for any errors. • Finish linking the functions to the frontend. • Implement the frontend for the game. • Confirm weekly progress with the TA.
Week Seven	November 12th - November 19th	• Create documentation. • Ask the TA and other external evaluators to try the game. • Check for suggestions and fix them. • Confirm weekly progress with the TA.
FINAL EVALUATION OBJECTIVES: • Achieve a working game with all the levels. • A clear documentation of the work done. • Submit final evaluation reports or whatever is required. • Create a presentation explaining the game.		