World View Project Report

Contributors:

- Ivan Bury
- James Panya
- Aiden Zavala
- Gregory Del Bene
- William Mowbray
- Jake Zellmer

Project Description:

World View is a social platform that allows you to create and share your own websites or “worlds” with other users. After creating an account, you have access to an editor that allows you to build your world using HTML and host it on our servers. After creating your world, you can view other worlds either by searching or through the explore page which will randomly take you to another user’s world. Beyond the basic functionality of creating your world and exploring other worlds, World View has some social features that allow you to interact with other users. First, there is a friends page that stores a list of your friends along with shortcuts to their worlds. You can add to this list by searching other users or you can remove users from your friends list. Additionally, the direct messaging feature makes it easy to connect with other users and learn more about them or ask about how they coded certain aspects of their world. By going to the messages page, you can select from a list of available users to see your message history with them and send or receive new messages.

Project Tracker:

https://github.com/users/SpurSlicer/projects/2

Demonstration Video:

https://drive.google.com/file/d/1JeABjJcyINoQnsuxBUr2E-aMtFkb3_3E/view?usp=sharing

Version Control:

https://github.com/SpurSlicer/World-View

Contributions:

- Ivan Bury: Ivan handled the direct messaging and friends features. The direct message system allows the user to visit a page containing a list of other users where they can click on a user to view their messages with that person and send/receive new messages. The friends feature lets users search for other users on the site and add them as a friend. Once a friend is added, they appear on a list of the user's friends. This list gives the user shortcuts to view their friends' worlds or unfriend them. Message and friend data is stored in a SQL database.
- James Panya: James handled stylizing for most of the pages and creating a dynamic navbar for ease of browsing through the website. James created a small mockup, within an application called Figma, to help the team stylize what some of the pages should look like. The pages that James contributed to the page's styling were Log In and Register.

James helped design a mockup for what the ‘Home’ page would look like. The dynamic navbar allows a user to browse through the website with ease and changes depending on whether a user is logged in or not. James would handle and post the release notes for the project.

- Aiden Zavala: Aiden handled creation and styling of the homepage front and backend. The homepage will allow users to see other users’ worlds and be directed to their world. The homepage presents color changing “world nodes”, created using JavaScript, which display a user’s name and their file name, using the already existing SQL Databases. Aiden also created the User Acceptance Test Cases, covering features like login and register. Aiden provided James with the original draft wireframe with what he then used to create his mockups.
- Gregory Del Bene: Gregory handled the filesystem backend implementation, file system and user base API calls, and front end for the “world editor” which consists of a basic text editor with file operations. Gregory also handled the backend for file retrieval in order to view user worlds. A message, query string, and cookie system were implemented to assist with passing information with ease throughout backend and for usage in frontend. While logged in, users are able to rename, save, delete, upload, and create new files for their world. To view any webpage–including the user’s own–the webpage is rendered within an iframe.
- William Mowbray: Sadly due to health and hardware difficulties I was not able to contribute as much as I would have liked. Despite being the first person to start adding to the repository I have not contributed a lot, I ended up making the basic scaffolding for our projects, simple things like improving aesthetics in various places I created the basics

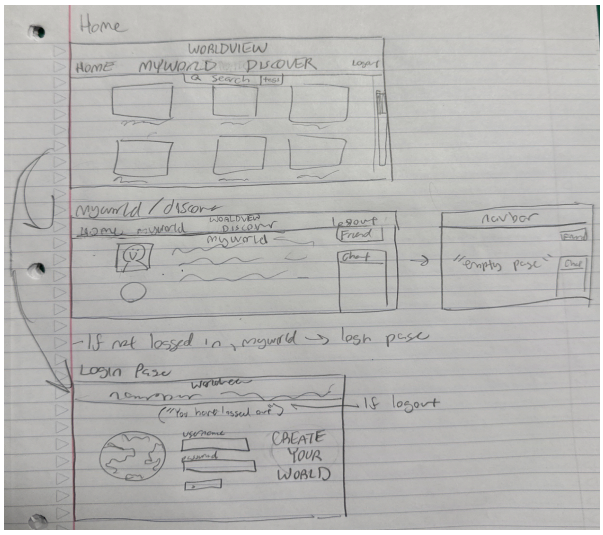
of all the foundations from the old labs like the register page (also added a disclaimer to said register page). Ultimately I primarily acted as support, while I did not directly submit any of the labs for this project I helped walk the person who submitted each of them through it acting as a second monitor brain and a go-between to ask questions. On top of that, I worked on the Discover page so people could see the code for specific files and what they do.

- Jake Zellmer: For the first half Jake spent the project working on integrating and exploring alternative third party API such as google drive to assist in the implementation of the file storage and maintenance system. This was super complex for me and very confusing but I learned a ton from it. After this Jake designed the Tag system. For this many different database methods were tested, eventually finalizing on one that integrates tags with each file. The tag system had to have the same information across multiple pages and users, and you had to be able to access it from anywhere, meaning it was implemented to be accessible even without a user logged into the session.

Use Case Diagram:

https://www.canva.com/design/DAF_bNmVHB0/5ncuL9CRyBVWu7i1wKSsFQ/edit?utm_content=DAF_bNmVHB0&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Wireframes:



Wireframe 1 designed from the graphic above

Wireframe/Mockup 2

Final Wireframe/Mockup

Test results:

Test Case 1: Create an account, login, and visit a random world. After looking at another user's world, go back to home page, and log out.

Results: The user was at the root page (which is the home page) and looked around. They utilized the navbar that had the register button and registered an account. They then logged in, which redirected them back to the home page with other user's worlds. They visited one of the worlds and went back to the home page. After that, they took a few seconds to find the log out, but saw that the navbar had changed. They clicked on the arrow button that pops up the log out button and finished the test case. The user did not deviate from expected actions, so no further

changes to anything was noticed during this test case. They also had no complaints when finishing the instructions.

Test Case 2: Create an account, login, and create a world. Go to the world editor and write a heading text saying, “Hello, World!” See that the world was created and sign out.

Results: The user was at root page and looked around. Saw the register page within the navbar and created an account successfully. The user also had no trouble of finding the “World Editor” page which was located on the navbar. The test proctor told the user where and what to write in the file editor as they were not that experienced in programming. After that, they had slight trouble trying to figure out how to look at their world. They were then told to use the white navbar underneath the green navbar and was able to see the “Hello, World!” text on their page. The user then took a few seconds to find the sign out button again, but was able to sign out to finish the test case. It seems that the users take a while to find the sign out button. If the next test case users have the same problem, we will make sure to change it.

Test Case 3: Create an account, login, and message user. Find where the messages page is, have a conversion with test_user1 (the test proctor), and sign out.

Results: The user had no trouble creating an account and logging in. The user also did not have any trouble locating the messages pages to find a use to message. They chose test_user1 as the user and messaged them. The test proctor and user were able to communicate with each other. The user took a few seconds as well to find the sign out button, but eventually finished the test case. We will definitely change the dropdown to something else as we only have “sign out” as the only drop down item. The user did not deviate from expected actions. They also had no complaints when finishing the instructions.

Test Case 4: Create an account, login, friend and unfriend a user, and log out.

Results: The user had no trouble creating and logging into their account. The user also did not have any trouble finding the friends page. Although they had to find a user to add, which was not described within the instructions. The test user looked around and found usernames at the home page. The test user then added the user that had the most interesting world to them. After that, they unfriended them and signed out to finish the test case. The test user asked how the user was able to add them back so fast. As we did not implement a request system for friends, the user would just be instantly added to the friends list. In a sense, it was more like a follow list more over a friends list. In the future, we will have a functioning request system for the friends list.

Deployment:

http://worldview.eastus.cloudapp.azure.com:3000/

Source Code:

https://github.com/SpurSlicer/World-View/tree/main/ProjectSourceCode

ReadMe:

https://github.com/SpurSlicer/World-View/blob/main/ReadMe.md