Marcel Metzger Student-ID: 4000497

TB2 Report

Development Process

I began this project by migrating all elements of my Tkinter-based application to the Streamlit framework. I started with the homepage, followed by the Number Game and the Word Game. During this process, I realized the need for a navigation system, which led to the implementation of a sidebar where users can select different pages.

Once the migration was complete, I considered the core functionalities the application should include, as well as its overall design and user experience. Consequently, I developed a login and registration system, enabling users to save their scores and track their high scores. Since we had already covered this topic in our seminar classes, I utilized the foundational pages from those sessions and adapted them to fit my application. A MongoDB database was used to store all user-related data. To enhance the user experience, I ensured that after logging in or creating an account, the login and registration pages would disappear from the sidebar, creating a cleaner interface and preventing potential technical issues.

The next step involved implementing a statistics page where users could view their high scores. Additionally, I developed a function that detects when a user achieves a new high score and updates the MongoDB database accordingly. To present the high scores, I displayed the user's MongoDB document in JSON format, omitting sensitive information such as passwords, usernames, and IDs. For this aspect, as well as for database-related functionalities, I sought assistance from ChatGPT, as I was unfamiliar with the required commands.

Following the completion of the statistics page, I proceeded to integrate a new game I had envisioned: *Simon Says*. I wanted to incorporate this game because it focuses on memory retention, requiring users to recall and repeat a sequence—offering a different cognitive challenge compared to the other games. After exploring different approaches to implementing *Simon Says* in Streamlit, I developed a version where users are presented with a sequence of colored buttons and must replicate the order correctly.

With the completion of this game, the core functionality of the *Memory Benchmark* application was nearly finalized. After ensuring that all features were working correctly, I focused on improving the app's design. To achieve this, I used ChatGPT to generate HTML-based markup elements, as I had no prior experience with HTML. Finally, in the last stages of the project, I reviewed the code, cleaned it up, and added comments for clarity.

Challenges and Limitations

Throughout the development process, I encountered several challenges. The first major difficulty was understanding and utilizing st.session_state. Initially, I struggled with this concept, as I had already found it challenging during the seminar classes. To overcome this, I dedicated time to thoroughly exploring its functionality. Eventually, I was able to develop a solid understanding of how st.session_state works and how to implement it effectively within my application.

Another significant challenge arose while developing the *Number Memory* game. For reasons I could not entirely pinpoint, this particular game was the most difficult and frustrating to work on compared to the other pages. Numerous issues emerged throughout the process, often requiring substantial time to troubleshoot and resolve. At times, it felt as though every attempted fix led to new complications. Although I managed to address most of these issues in the end, I am still not entirely satisfied with the final implementation.

Reflecting on the limitations of this project, I believe the primary constraint was my own coding abilities. For instance, I was unable to make the "Submit" button disappear after the user entered their answer in the *Number Memory* game, nor could I hide the "Start" button once the game began. Additionally, I was unable to clear the input field between levels, resulting in the user's previous input remaining visible. If solutions exist for these issues, then my programming skills were the limiting factor. However, if no viable solutions are available within Streamlit, then the framework itself posed a constraint.

These shortcomings make the application vulnerable. Due to the unresolved issues, the app remains susceptible to potential disruptions. For instance, repeatedly pressing certain buttons can cause the games to malfunction, negatively impacting the user experience.

Feedback from Testers

After completing the app, I shared the link with several people I know. The feedback was mostly positive, with many praising the design, the enjoyable gameplay, and the overall functionality of the app - despite the presence of buttons that do not disappear and serve no real purpose. A small number of reviewers pointed out the previously mentioned issue of games breaking due to button spamming. While I was aware of this problem, I was unable to find a solution. In conclusion, the majority of the feedback was positive.