Cliexa's Compliance Algorithm Project Report

Team Members: Nicholas Kong, Benjamin Yee, Khang Nguyen

Client: Cliexa

I. About Cliexa's Compliance Algorithm Application.

Cliexa's Compliance Algorithm is a web application that is designed to display a simplified visual output of different medical assessments based on the user's input. This web application is a medical assessment tool which allows the users to select a medical assessment to complete. After completing an assessment, the application will decide whether or not the user is qualified for treatment based on how the user's answers to the assessment. The application does not accept any user's input when answering these assessments. The users can only answer the medical assessment by clicking either the yes or no button to the question that best describes them.

II. Building and finalizing the application

1. Requirements

We met with the client to understand what application he wanted and what features he would like to see in this application. The team has written everything down that he would like to see in this application. After meeting with the client, we created a SRS (Software Requirement Specifications) document and filled out specific details about how the application would function. We then showed our SRS document to the client and got their approval to begin building the application.

2. Technologies picked

The team researched many web hosting services for the application and in the end, we decided to use Heroku. We chose Heroku because there is no license required, meaning that website development is free. The pipeline is very simple and it has an easy integration with other SaaS (Software as a Service) services and products. It supports a GUI (Graphic User Interface) which we will be using in our project. Application rollbacks are easy. Since Cliexa has said security is not an issue, we do not need to worry about the security concerns Heroku has.

3. Team members splitting the work

The team created a timeline using Google Sheets so all team members had easy access to view and modify the timeline. The timeline contained the different features and logic that needed to be implemented. We assigned different tasks on the timeline to each team member along with a deadline to facilitate the completion of each feature in a timely fashion. We also included a progress bar to show how far each team member was with the features they are assigned.

4. **Building the application**

The team each downloaded an IDE of their choice and then all installed NPM (Node Package Manager) on our computers. This allows us to run our application on our local machines to view how the application looks and functions in real time. The team used this to help build and debug the application to ensure the application was functioning correctly before pushing it to the Github repository which would deploy to Heroku.

5. **Alpha Testing (Round 1 of Testing)**

To begin our alpha testing, we had our client provide us with six people to test our application. The team provided those six testers with a link to access our application, instructions on how to use our application, and a Google survey so the testers could provide us feedback after they finished.

6. **Changes made after alpha testing**

After receiving the alpha testing feedback from our testers, we organized it into software issues and suggestions for improvement. We then divided up the improvements to each team member. While we worked to implement the improvement we also made sure to fix the bugs that were brought to our attention.

7. **Beta Testing (Round 2 of Testing)**

After fixing the broken features and bugs based on the alpha testing feedback, we updated our application on Heroku. We then provided our six testers with the same link to access our application, updated instructions on how to use our applications, and an updated Google survey so the testers could provide us feedback.

8. **Changes made after beta testing**

After receiving the beta testing feedback from our testers, we split up the feedback into individual features and assigned team members to those features. We fixed a few bugs that our

beta testers have reported. In our beta testing, we got less bugs and broken features reported than our alpha testing. We were also told that the project improved a lot during our beta testing round.

9. Finalizing the application

We updated and finalized our application based on the beta testing feedback. We created and sent our technology transfer file to our client which contains all the documents, diagrams, references, feedback from both rounds of testing, instructions on how to run and use the application, and the source code to Cliexa's Compliance Algorithm application.