WAD FIRST ASSESSMENT REPORT

Konstantin Milchev

51663558

The Aaron Rodgers Project

1. VISION

The purpose of this document is to collect basic information about Green Bay Packers quarterback: Aaron Rodgers. Furthermore, this wiki has implemented a questionnaire feature so users would be able to provide feedback based on their experience from the website. Finally, this wiki creates a friendly environment where Rodgers fans would be able to view additional information regarding him.

2. FEATURES AND FUNCTIONALITIES

It does so by providing user-easy functionality and a user-friendly design. The main page at the moment of entering is the Home page. It contains buttons which can lead the user to all parts of the website. As mentioned, the project starts with a Home Page which contains three text fields. The top field is specific for the Homepage and displays brief information about the main attraction of the website (Aaron Rodgers). The bottom – left field displays the number of characters and words in the bottom – right field. The latter is a textual file, which can be edited in the "Edit" page. (*Screenshot2)*

An About page. Provides a short summary of the creator of the webpage. (*Screenshot1)*

A Register page, where a future user is able to register on the website by selecting a username and a password. After the creation of an account is complete the user can use the log in button in order for them to be redirected to the log in page, where they can enter credentials and enter the website. *(Screenshot3)*

An Edit page. Only one account has authorization to view the contents of the page. The credentials for the user are *username:* Admin *password:* Password. The Admin user has an additional button which allows him to view a list of all the users registered on the website.

On the Edit page there are three buttons underneath the textual field. The Reset button resets the page to the initial statement. If the text has been altered the reset button will return the initial value. The Backup button saves the text onto a textual file called backup. Thus, if there is an issue with the text it can be restored via the Restore button, which brings back the backup text.

One important functionality is the information being saved in the active_record file. When the user logs in and if possible edits the page, the file will keep those changes along with the username, time of action and the updated text content.

Additional features:

The About page contains an implementation of a google maps API.

A Questionnaire page. Contains several questions aimed at exploring the user's impression from the webpage. The latter functionality is designed in order for the owner of the website to receive valuable feedback and strive to satisfy its users. After the questionnaire has finished the user presses a button and sends the contents of the questionnaire via email to a previously specified address. *(Screenshot4)* I was able to adopt this code into my own after finding it online (http://www.wikihow.com/Create-a-Questionnaire-in-HTML)

A Video page. This page contains two video clips from YouTube featuring Aaron Rodgers. *(Screenshot5)*

Every page has a website link at the very top, which redirects the user to the Green Bay pakcers website.

Everything apart from the standard functionality and the questionnaire was added and from "w3schools".

3. Difficulties report

Throughout the development phase there have been several difficult periods. Firstly, I had issues with the *active record* gem. Initially it did not create a database when the wiki started. After some research I and found out that some syntax needs to be changed and an operation needs to be provided to the wiki.rb. I could have used it but decided to go with data mapper as I had already started practicing with it.

Furthermore, I had issues with the main container and body . I removed main container because I believed that it restricted my code with the initial code within it. We had also initially set a *hight* value as part of the body code, which was not working out as I had my text field where the buttons were and the user would not have had a pleasant experience.

4. Running the application

Windows users*
If you would like to run the application you would need to download one of the versions of ruby available on railsinstaller.org
After this is complete you would need do download several gems by running the command: *gem install gem-name-here*

- sinatra
- data_mapper
- dm-sqlite-adapter
- sqlite
- haml
- rspec

Next, you would need to press the windows button and write down "cmd" in order to find the terminal through which a command will be entered and the application ran.

When cmd is opened it will display a directory, place the project file inside that directory. Afterwards, write down the command *cd project* This should open the folder project inside the original directory.

After the cd project command write down *ruby wiki.rb* This will run the code inside the wiki file.

Finally, open a chrome web browser and type in the searchbar: *localhost:4567*

That should do it. Your code should be working and displaying pages as the ones in the screenshots.

If you wish to have access to all parts of the website you would need to use the username and password previously specified.

Appendix:

The screenshots are located in the main application folder/screenshots