Web Technologies: Final

Charana Nandasena (an15739) and Joshua Van Leeuwen (jv15626, 23305) University of Bristol

May 21, 2018

1 Prerequisites

- We are using secure HTTP through use of a self-signed certificate and as such, our personal certificate authority must be added to you chosen browser. This root CA can be found here '*site/secrets/rootCA.pem*'.
- We are utilising docker containers in our website to isolate nodejs jobs, and although not required, it is recommended that this is installed and running. If installed, ensure docker is running by checking the systemd unit it running '*systemctl start docker*' on Linux or that the '*Docker.app*' process is running on MacOS. Failing this, the server will fall back to utilising the local userspace however, this is not the intended execution.
- Once '*server.js*' is running, the server can be reached at '*https://localhost:8080*'

2 Introduction

This website is designed to be an educational tool for users to learn to program Javascript. This is primarily done by users completing a set of Javascript programming tasks where their progress is tracked. A supporting forum is included whereby users are able to create forum 'threads' where they are able to ask questions or have general discussions with the community. Some games are also included to demonstrate the possibilities of what the user will be able to create once the user has completed the website tasks and furthered their ability to code using Javascript.

Although the user is able to browse the website, much of the functionality requires the user to have signed up and have an active session. This grants the user the ability to post and reply on the forum as well as attempt the challenges.

3 HTML and CSS - A

- EJS for dyanamic webpages, html templating and generation
- webpage XHTML delivery and Ran HTML through XHTML validator jar.
- Examples

  Figure 5a shows use of css animation keyframes including basic 2D translation and 3D rotations, box shadows for popin/popout affect, absolute positioning with z-index manipulation to manipulate layers etc.

  Figure 6b shows use of background color linear gradients, css filters (specifically blur), a popup using html and javascript show/hide handlers etc.

4 Javascript

JOSH - Basic javascript work. Some API calls, calling docker, reading writing files, a few simple games. CHARANA -

5 PNG - A

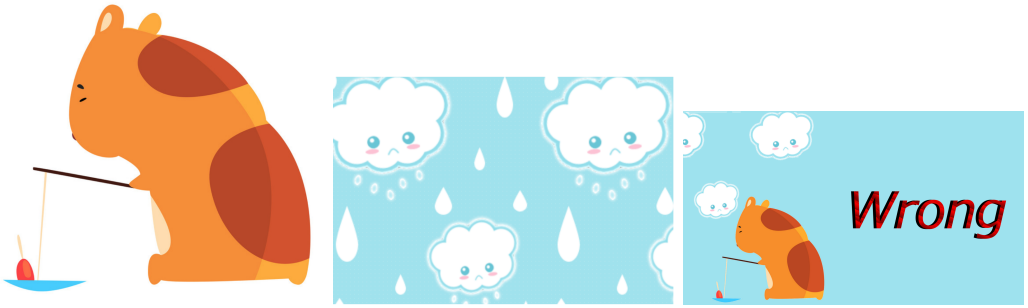
Two images were made in GIMP as seen in 7b and **??**. 1b and 2b were used as the backgrounds respectively and 1a and 2a were used as the foregrounds respectively.

The associated images for each graphic were first resized, manipulated (colors edited, portions removed/moved etc.) and background thresholded from foreground before being replaced by an alpha channel.

The cleaned imaged were then imported as individual layers.

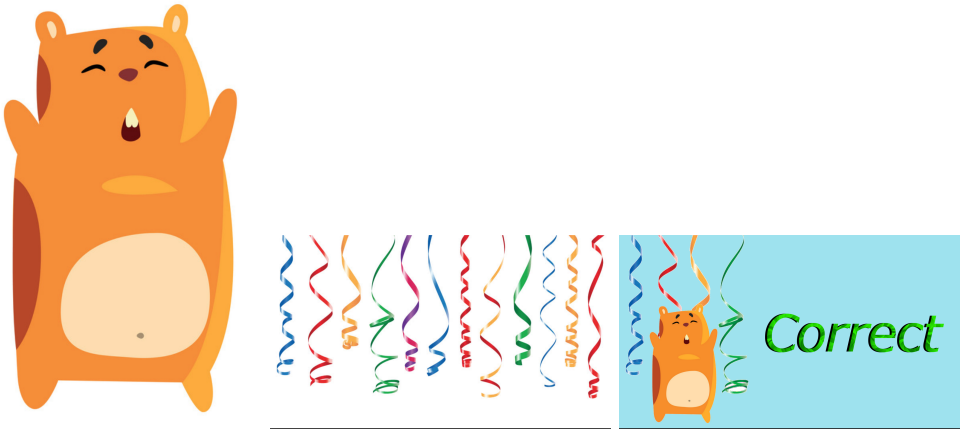
A new text layer was created to write the "success" or "wrong", a blur filter subsequently then applied to a duplicate layer obtain the 3D text affect. A gradient color was created and applied to the original text layer (only coloring the front facing side of the text).

The layers were then combined and exported as a PNG. This was done for each image.



(a) Hamster Failed (b) Sad Background (c) Wrong Graphic

Figure 1



(a) Hamster Success (b) Success Background (c) Correct Graphic

Figure 2

6 SVG - A

5b is our site logo made in Inkscape. In creating each component of the logo circles, rectangles and polygons were first draw, these shapes were then converted into paths using the "stroke to path" tool. In creating more complex shapes the "bezier curve" tool was used. The "edit paths by node" was used to manipulate the curve trajectories, carefully inserting/removing nodes along the outline to obtain the desired shape. Several individual components were then grouped together and flipped horizontally to obtain a symmetrical look. A linear gradient was finally applied along the vertical axis.

7 Server - A

The server has been expanded upon from the original server.js file given.

- Use of URL validation and redirection.
- Sign up and reply/post field validation.
- Handling various user data including text, code and images.
- Generation and validation of captcha to prevent malicious botting of sign up requests as well as user submitted forum posts and replies.
- Extensive use of our own written modules with both private and public exposed functions providing APIs to applications of the server.
- Use of a standard page delivery flow in conjunction with our modules to provide server with ease of extensibility. This is achieved through use of so called 'pre functions' and main data delivery functions that allow dynamic data delivery to the EJS HTML template library resulting in dynamic pages based on the state of the server and database.
- Providing HTTPS connection to ensure a secure connection. This is using a self-signed certificate which is enabled as described in the perquisites.
- Use of internally generated and managed cookies to provide user sessions.
- Passwords stored using a salted hash preventing storing passwords as plain text in the database.
- In order to isolate code given by the client, the use of docker containers have been used to execute the code, preventing malicious code being run in the server namespace. The code will be run in a docker container whereby it's stdout/err will be captured and saved to file. This file is copied out of the container and is checked against the relevant solution file. As described in the perquisites, the use of the local userspace will be used instead, in the event of the docker image failing to build, preventing this feature.

8 Database - A

The SQLite relational database has been used as our database.

- A number of tables are used to store user data, user voting data, user challenge progress data, challenge data as well as forum data.
- Each group of associated tables are accessed by through use of separate modules, acting as APIs, all in turn interacting with the main database_api module that manipulates data in the database.
- Tables are created relational so to provide extensibility to data such as the forum replies table relating the forum post table.
- Passwords are stored as a salted hash along with the salt in the users tables to prevent storing as plain text.
- Modular tables have been used in an effort to reduce read and write access of the database.

9 Dynamic Pages - A

10 Depth - X

A Images

Figure 3: Header

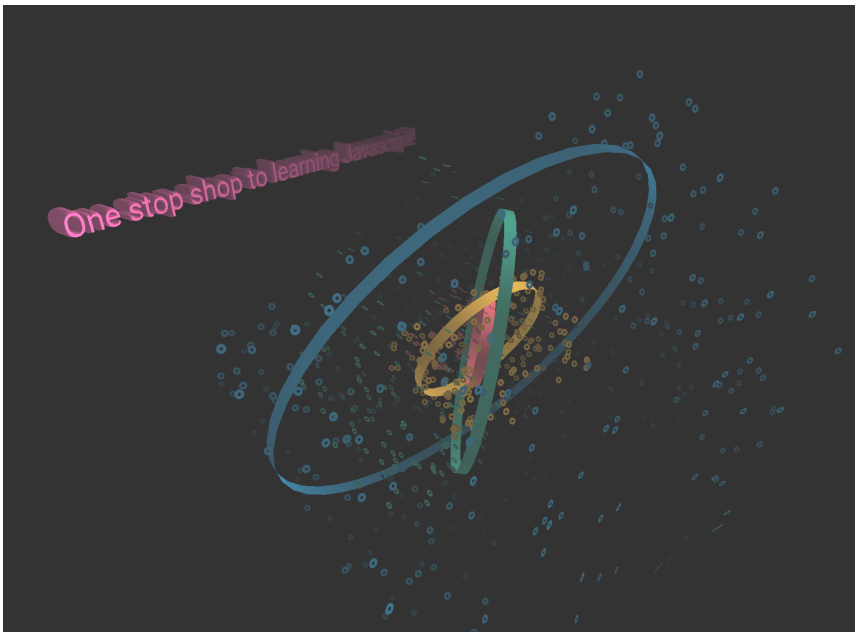
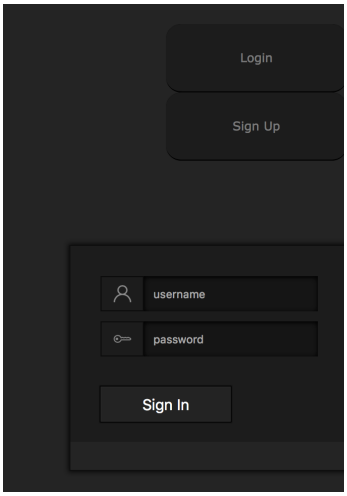
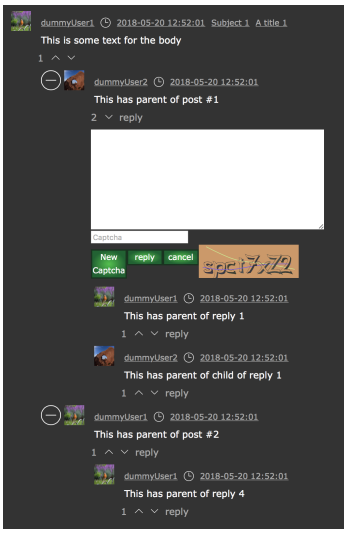


Figure 4: 3D Background



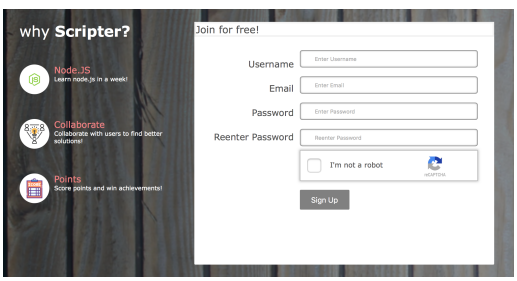
(a) Settings Popup and Login

(b) SVG Logo

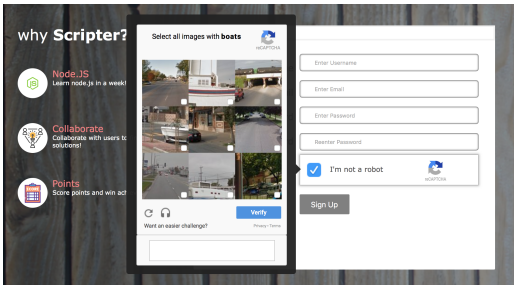


(c) Forum Post

Figure 5

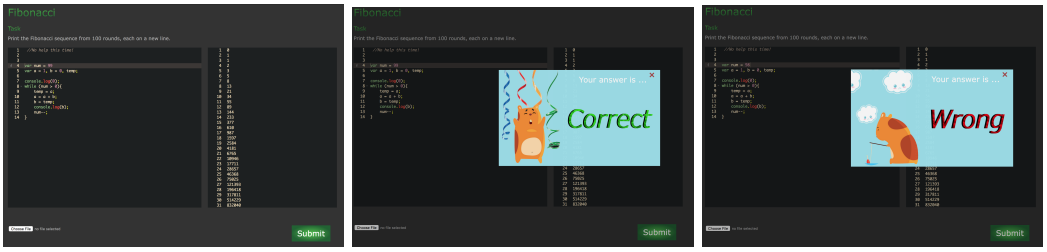


(a) Sign-Up Default



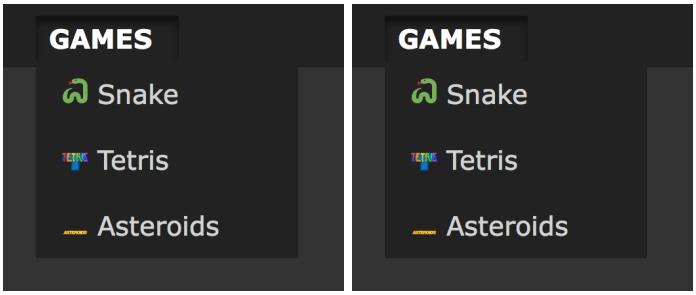
(b) Sign-Up RE-CAPTCHA Popup

Figure 6



(a) Editor Default (b) Editor Success (c) Editor Failure

Figure 7



(a) Games Dropdown (b) Games Dropdown

Figure 8