Requirements Specification Document

1.1 Introduction

Squibble is an online website based in React, Firebase, and Node.js that acts as a canvas to collectively host media and drawings from users all over the internet. It authenticates user input, and manages animated and static raster and vector graphics, ordering their visibility by the order they were posted in through the use of the Firebase database. Squibble can manage various sizes and colors, text fonts and sizes, as well as images, and gifs. The website shall also have basic moderation through the use of Firebase authentication. After a fixed period of time elapses, the board is reset for all users, and a copy of the final board is saved to a publicly accessible archive where it can be viewed.

1.2 CSCI Component Breakdown

The online whiteboard application, Squibble, is composed of the following CSCs divided into six different subsystems and subcategories. These categories can be noted as both the User Interaction category and the Website Functionality and Backend category. The interactable category shall include the whiteboard display [WBD], media object, and the post-it object concepts. The website functionality section shall be work involving the header, timer, and archive features of Squibble.

1.3.1

The following requirements are levied on the User Interaction subcategory of the Squibble project.

1.3.1.1 The subcategory shall be displayed in a web page.

1.3.1.2 The page shall be hosted with React and coded with Node.js.

1.3.1.3 The page shall have a whiteboard that will have a fixed length and height and shall be navigable with dragging of the mouse, mousewheel, and arrow keys.

1.3.1.4 The page should have the ability to change page colors including a possible dark mode.

1.3.1.5 The page will work with Firebase authentication to log in Google users.

1.3.1.6 The page, once logged into, shall display a button to access a side menu to allow for media to be added.

1.3.1.7 The page, once logged into, shall display access to the subpage for past website archives.

1.3.1.8 The submenu should consist of four different options including text, images / gifs, drawing, and post-it.

1.3.1.9 The text menu shall allow text to be selected from a variety of fonts and colors.

1.3.1.10 The text menu shall allow text from a dialogue box to be dragged onto the board.

1.3.1.11 The image menu shall allow for a user to upload images to be pasted.

1.3.1.12 The image menu should allow a user to search for gifs through an api such as Giphy or Tenor.

1.3.1.13 The image menu shall allow an image or gif to be dragged and placed on the board.

1.3.1.14 The drawing menu shall open a small sketchpad for the user to draw on when clicked.

1.3.1.15 The drawing submenu should have the ability to select multiple colors, sizes, and pen types.

1.3.1.16 The drawing menu shall have a completion button to save a work for use when done.

1.3.1.17 The submenu shall allow for media created or searched to be dragged and dropped anywhere on the board.

1.3.1.18 The submenu shall allow a user a separate option to create a post-it note opening a small menu.

1.3.1.19 The post-it note submenu should allow the user to change the color of the note.

1.3.1.20 The post-it note submenu should allow for any type of media previously mentioned to be added to the creation.

13.1.21 The post-it note, when done, shall be able to be dragged onto the board by a user.

1.3.1.22 The post-it note shall have the ability to layer on top of any other board post covering them and making a hierarchy.

1.3.1.23 The website media hierarchy system shall be handled by Firebase database and moderation.

1.3.1.24 The website shall have the ability to handle basic moderation through human use

1.3.1.25 The website should have the ability to moderate with censorship features such as blurring and pixelating images or text

1.3.2

The following requirements are levied on the Website Functionality and Backend subcategory of the Squibble project.

1.3.2.1 The page shall have a header above the whiteboard.

1.3.2.2 The header shall host the site logo, timer, sign in button, sign out button, media submenu, and archive page.

1.3.2.3 The header shall only display the sign out button and media submenu when a user is signed in.

1.3.2.4 The header should give brief instructions on submenus and where everything is.

1.3.2.5 The logo shall reload the website for a user when clicked on.

1.3.2.6 The timer of the website shall be in the middle of the page.

1.3.2.7 The timer of the website shall start at three days and tick down over time.

1.3.2.7 The timer should have some sort of setting to toggle its display.

1.3.2.8 The timer, once it hits zero, shall reset back to three days.

1.3.2.9 The page shall be reset and cleared after the timer hits zero.

1.3.2.10 The page shall screenshot itself with the use of Puppeteer through Node.js when before reset.

1.3.2.11 The whiteboard shall clear off all media during a reset.

1.3.2.12 The screenshot shall be saved to the website in a separate page titled “archive”.

1.3.2.13 The header shall have a button to access the archive.

1.3.2.14 The archive button, when clicked,shall link to another web page.

1.3.2.15 The archive webpage shall contain photos of screenshots of the website.

1.3.2.16 The screenshots shall be ordered from most recent to least recent by scrolling down the page.

1.3.2.17 The screenshots should have the timespan above them of when the website was active.

1.3.2.18 The archive should be updated every three days moments after the whiteboard reset.

1.3.2.19 The subpage shall have a back button on the top right to return to the main page.

1.4 Performance Requirements

Squibble shall not require any special computing hardware to operate.

1.5 Project Environment Requirements

1.5.1 Squibble shall be able to execute using any standard web browser.

1.5.2 Squibble shall be able to be viewed with any media account.

1.5.3 Squibble shall be able to post through Firebase with a Google account.

1.5.4 Squibble should be able to work on both desktop and mobile devices.

1.6 Development Environment Requirements

1.6.1 Squibble should be able to be worked on in any code editor.

1.6.2 Squibble shall require Node.js and therefore Javascript for backend and frontend code.

1.6.3 Squibble shall require multiple API's that work with Node.js possibly including Tenor and Puppeteer.

1.6.4 Squibble shall require React support to allow for hosting and frontend design.

1.6.5 Squibble shall require the use of Google Firebase for both authentication and databases.

1.6.6 Firebase shall be used to allow for sign in and sign out into the site and moderation.

1.6.7 Firebase shall be used for databases to allow for archiving, hosting, and deleting images on the site.

1.7 Execution Environment Requirements

Due to the nature of the project, Squibble will not need to be executed from a local device aside from hosting with React and deployment with Firebase.