Group Member: Xia Sheng cy18268, Kevin Ho pi18288

Overall Estimate of Marks:

- A for HTML
- A for CSS
- A for JS
- A for PNG
- A for SVG
- A for Server
- C for Database
- A for Dynamic pages
- 28 for Depth (out of 40)

The idea of this website is to create a home page for users' browsers. User can get all the information they need like news, tech news, weather, date and time just on one website. It also has a tool bar to direct users to popular website like YouTube, Facebook, LinkedIn etc. It gets latest news from many news agencies' RSS feed and store them in database for user consumption. Furthermore, it has a user management system with authentication for further expansion on more personalized features. User can set this website as their front page and be connected to the world straight away.

Since it is a dashboard/home page like experience, MEAN stack was chosen to be the collection of software for this project. Using Angular as front-end framework, a single page application could be easily formed, with better and suitable user experience for the website (avoid full page reloads etc.). Express provided a flexible, simple and extendable web application framework. Whole stack is based on JavaScript which full filled the requirements.

The main point of this project is to implement and experience various web technologies. Thus, apart from using the framework, the team created many elements of the website from scratch. For example, instead of using popular style libraries such as Bootstrap, the team created several custom style sheets and animations for the website for its unique look.

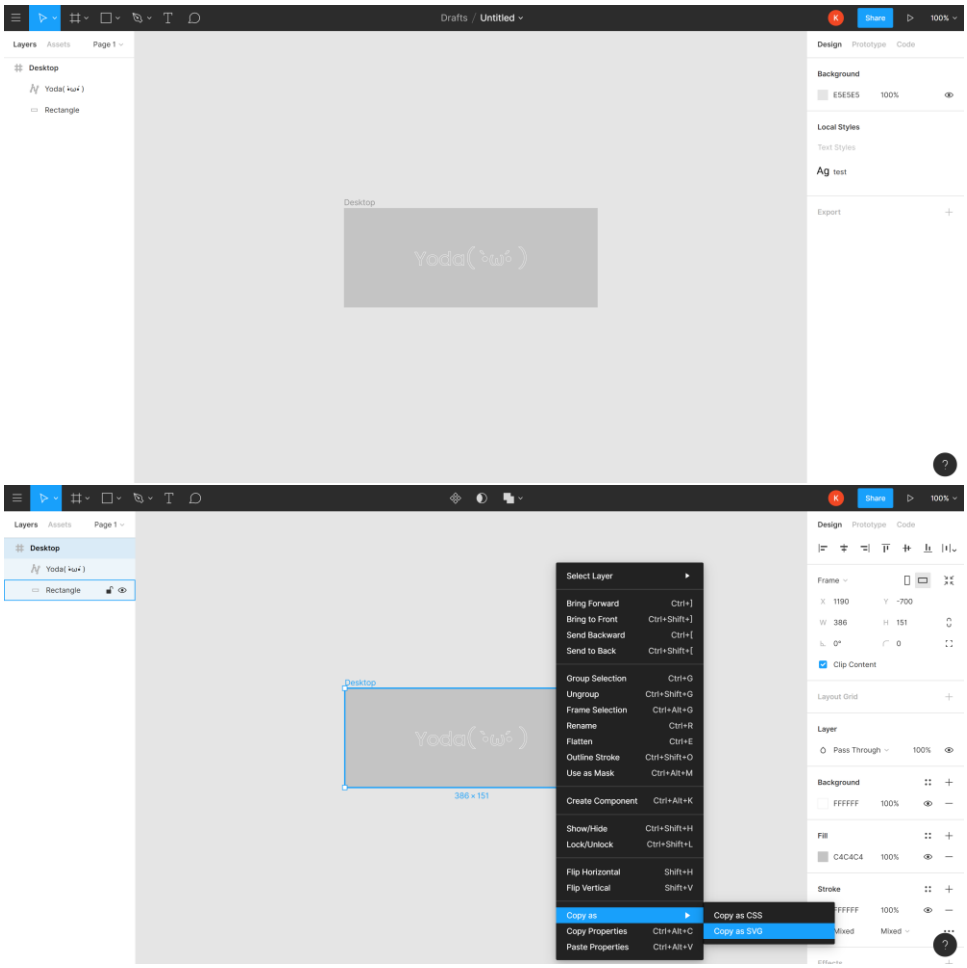
All wallpapers are downloaded from https://www.pexels.com/

HTML: All the layouts of the webpage were written from scratch. All the html were located in app and view. The app/index.html is the main page of the angular application holds the header, the about us modal and the <main ui-view></main>. The ui-view is controlled by a UI router which allows the user to easily switch between different webpages with a clear, unchanged navigator. This ui-view will direct user to news, account when user click it on the navigation bar. Each tab like account, news will have their own folder and html document in the app folder. Due to the collaboration with the server, the login page and register page are in ejs format rather than html, but the contents are still html except the alert massages. We create the modal from sratch , the team just doesn't have anything to write about us so we input lorem ipson as the paragraph. We didn't copy anything.

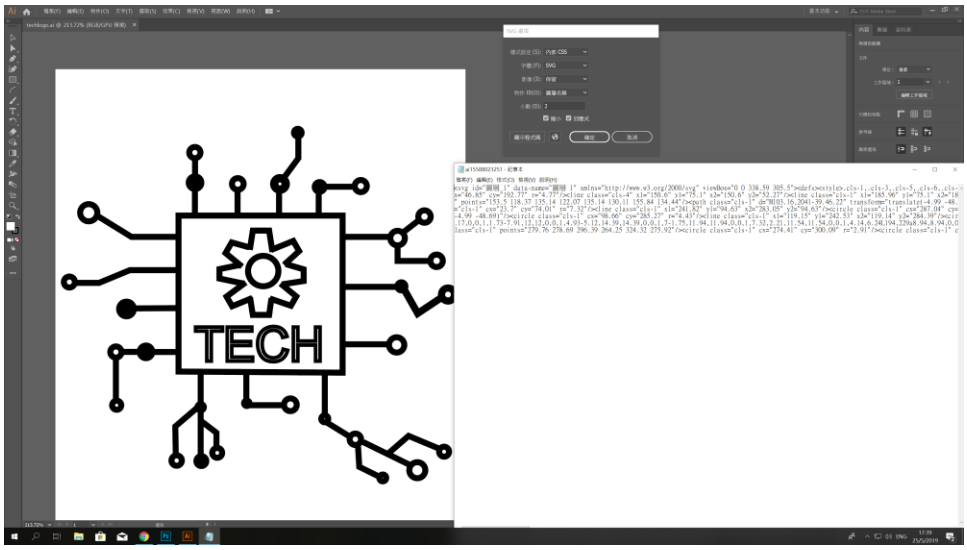
CSS: The team created all the css stylesheet. All the css stylesheets are in the app/CSS and loginCSS. In the master.css and loginCSS/testlogin.css added a navigation bar CSS animation which will enlarge the text, blur the background and fill up the box with yellow by the hover effect of different section of individual navigation button. I used a flex box approach for most of the containers to align different div block either horizontally or vertically. The home.css is mainly set up the layout and adding animation to the side tool bar in the home tab. The side tool bar firstly used logo from a website called fontawsome. Then the CSS spreadsheet add a transformation edge and color filling animation to individual icons. newsLogoSVGanimate.css was used to control the svg animation of the new and tech logo. Google font was also used to decorate some displaying text. The login screen css styling too inspiration from the https://blog.prototypr.io/how-to-create-the-snake-highlight-animation-with-anime-js-bf9c6cb66434 SVG animation tutorial.

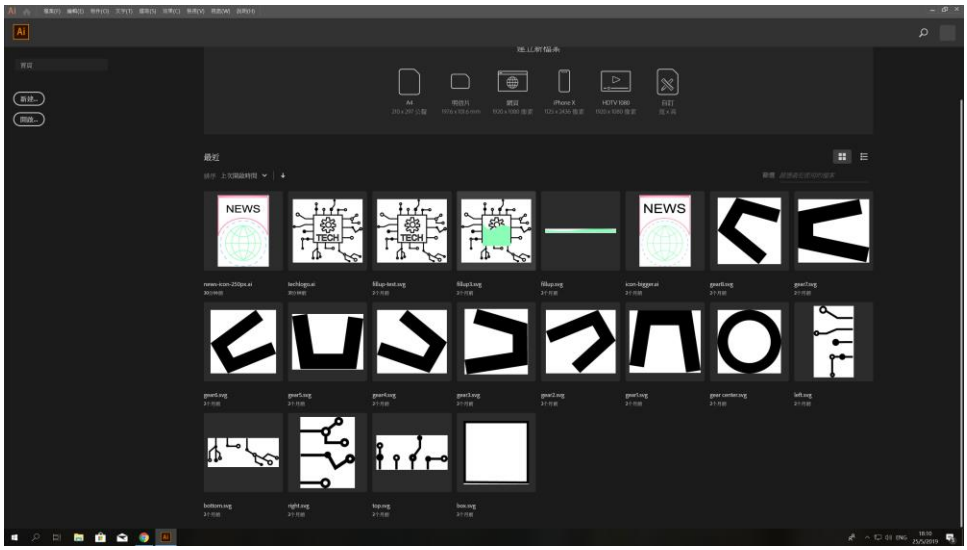
JS: All the client side javascript are located inside **app/js** and **loginjs**. All the javascript file are implemented at the end of all the html files. type_writer_effect.js created a function to implement a type writer effect (which will loop along an array of words and type out and delete them) in the welcoming massage in the home page. The team also implement clock.js to display date and time in the home page. The team used the api provided by https://enlight.nyc/projects/weather/ to implement the weather module in the home page. The team implement the modal_control.js to control the modal created by the team for the about us tab. AnimationClickControll.js was implemented to control the click trigger for the svg animation of the tech and news logo in the new tab. There were more JavaScript files created for server-side application which will be discussed later. In the loginjs folder, it has testlogin.js which was adapted from this https://blog.prototypr.io/how-to-create-the-snake-highlight-animation-with-anime-js-bf9c6cb66434 tutorial to control the “moving snake” svg animation using anime .js.

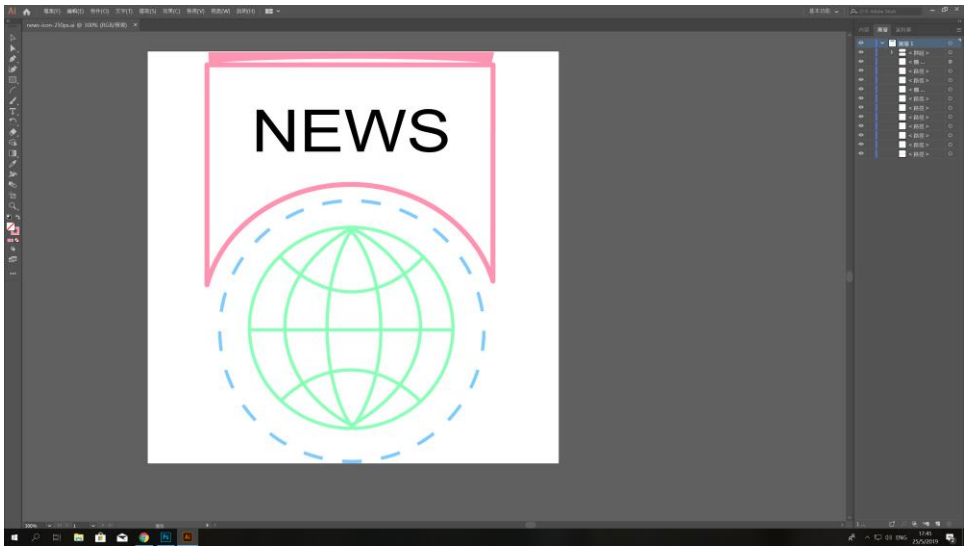
SVG: This website has three custom made SVG animations.

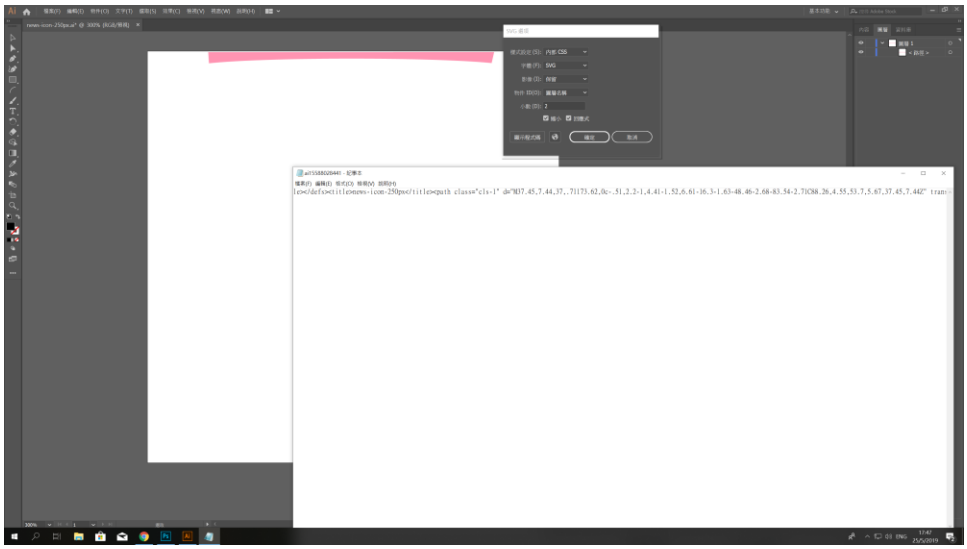


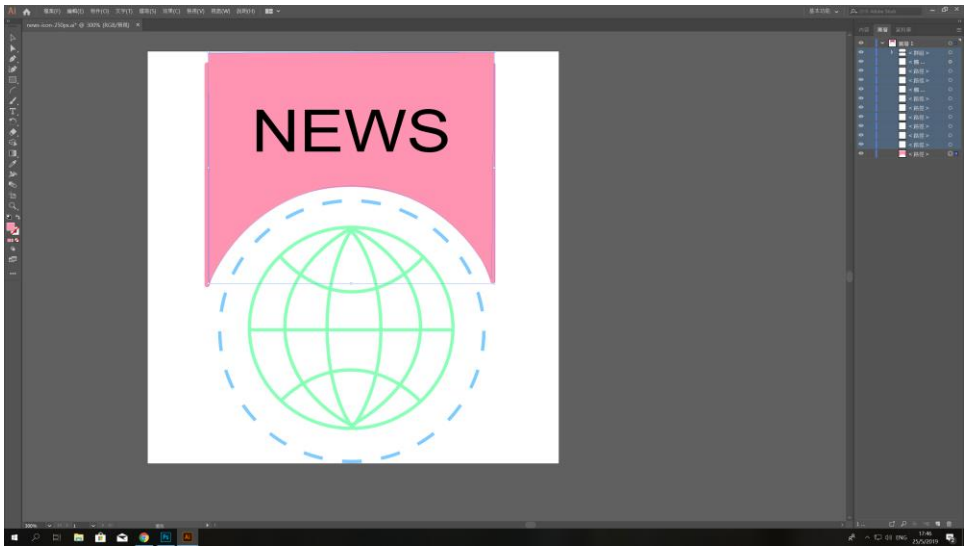
First the logo, which is an inline SVG. It was started off to be text and a software called figma was used to generate a svg outline path of the word. It's then animated by CSS. The stroke-dasharray function was first used to cover up part of the outline path. Then the stroke-dashoffset function was used to move the position of the part of cover up. This will cover up the whole stoke temporarily. Then animation function in css was used to set up the timeline and generate the animation to fill up the outline of the word. This process was repeated for individual letter in the logo.

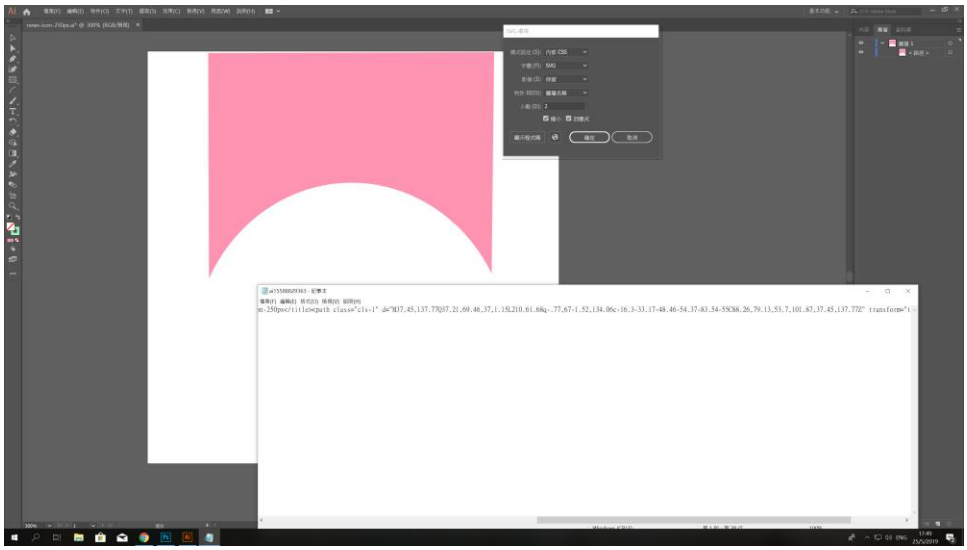

TECH


NEWS
NEWS

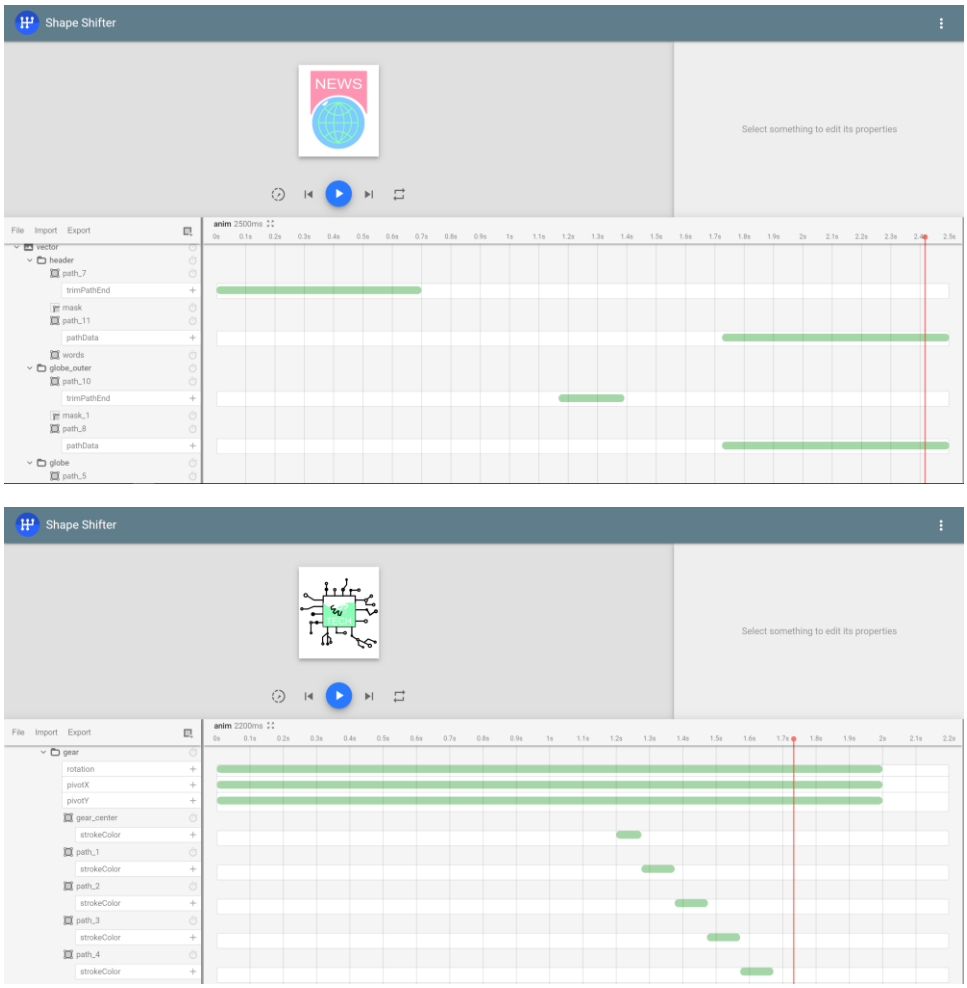

NEWS




NEWS



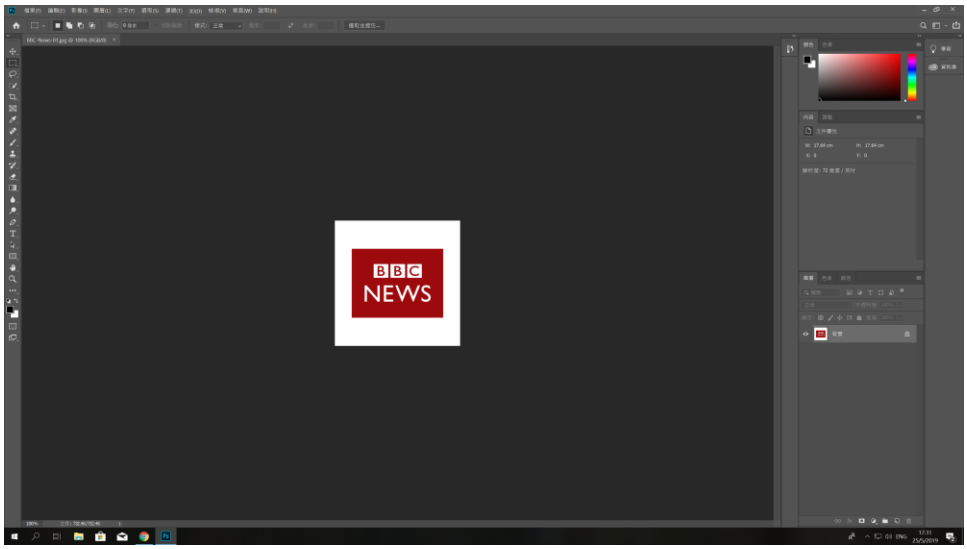
Then the news and tech logo were created by external SVG files. Both logos were originally designed and created by the team. Both logos were first drawn out in Adobe illustrator. Individual parts in the logo were exported as individual SVG files. Some shapes were dragged and resized to cover up the logo which acts as the fill up animation. The direction and location of how the fill up shapes was moved was extracted from illustrator as SVG.

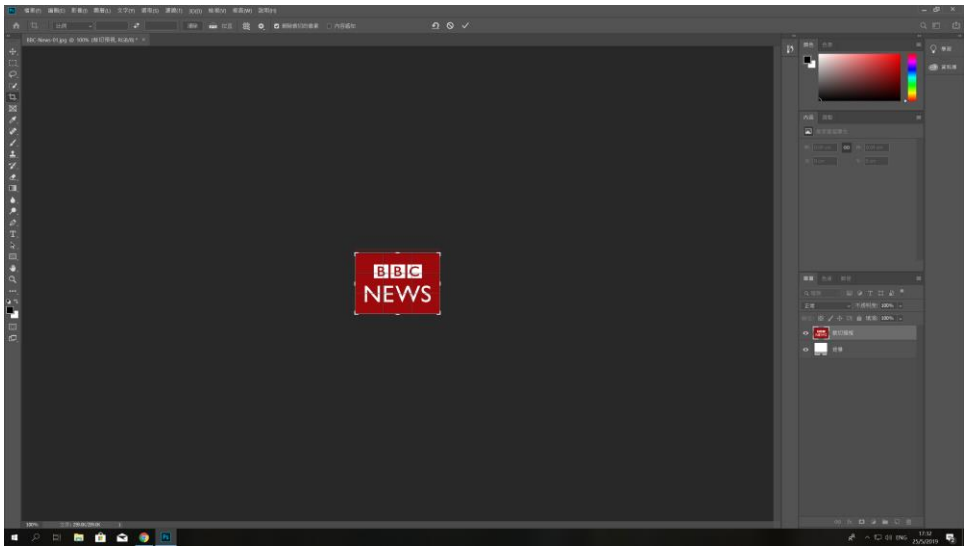


These SVG files were then imported into a software called shapeshifter. Shapshifter allowed users to set the animations type and time line for individual part of the SVG logo. All the layers and animation were exported as one SVG file. Finally, CSS was used to generate the animations of these two SVG logo and java script function was used to enable a onclick action to activate the animation.

The “moving snake” svg animation in the login screen was adapted from this https://blog.prototypr.io/how-to-create-the-snake-highlight-animation-with-anime-js-bf9c6cb66434 tutorial.

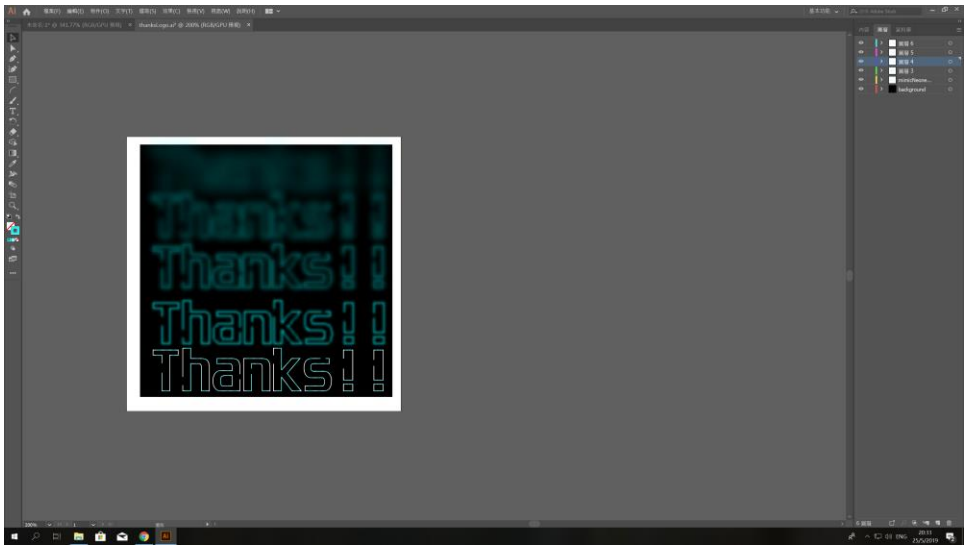
PNG:



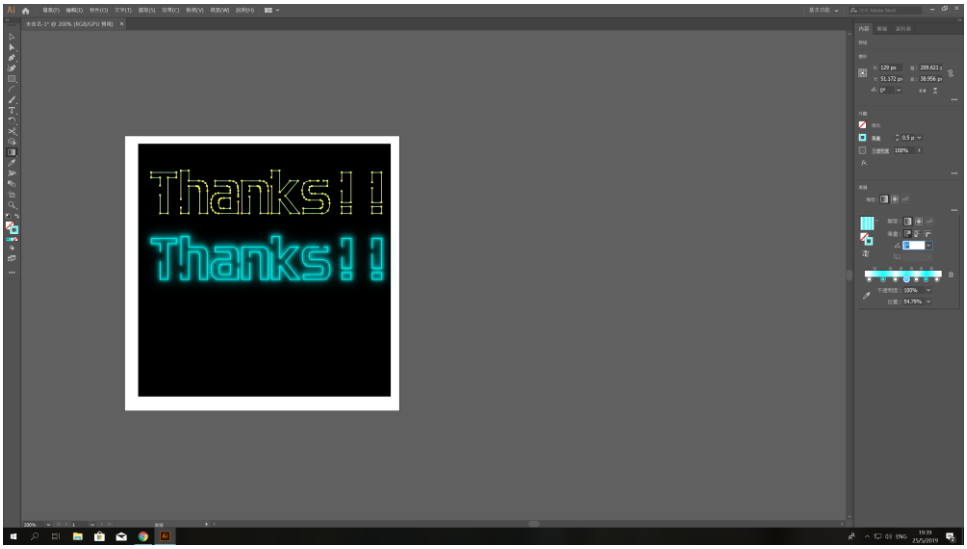


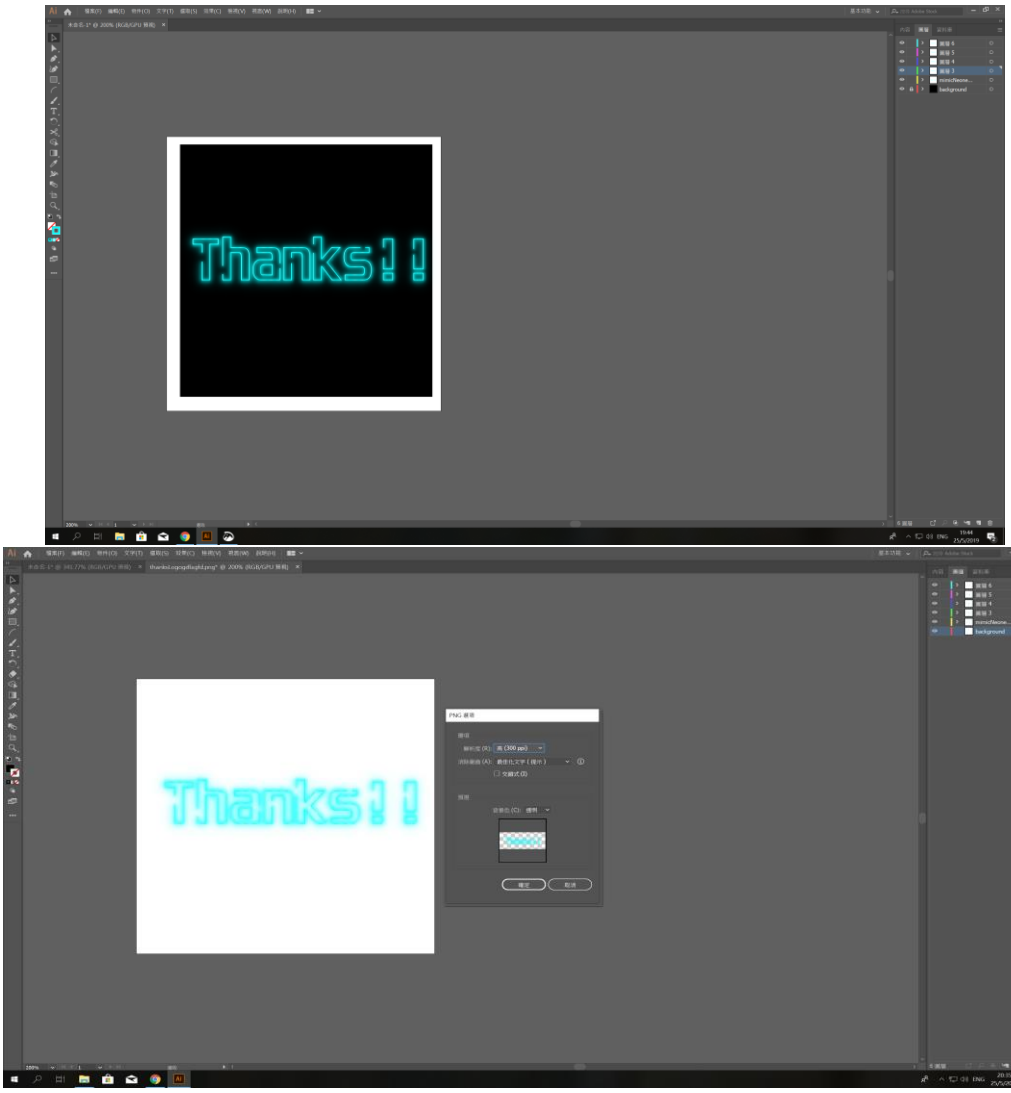
Some logos are downloaded as JPG with white background. Adobe Photoshop was used to crop out the logo and save the logos as PNG.

The team also tried to create some original PNG artwork. The first one is a neon light effect thank you sign.

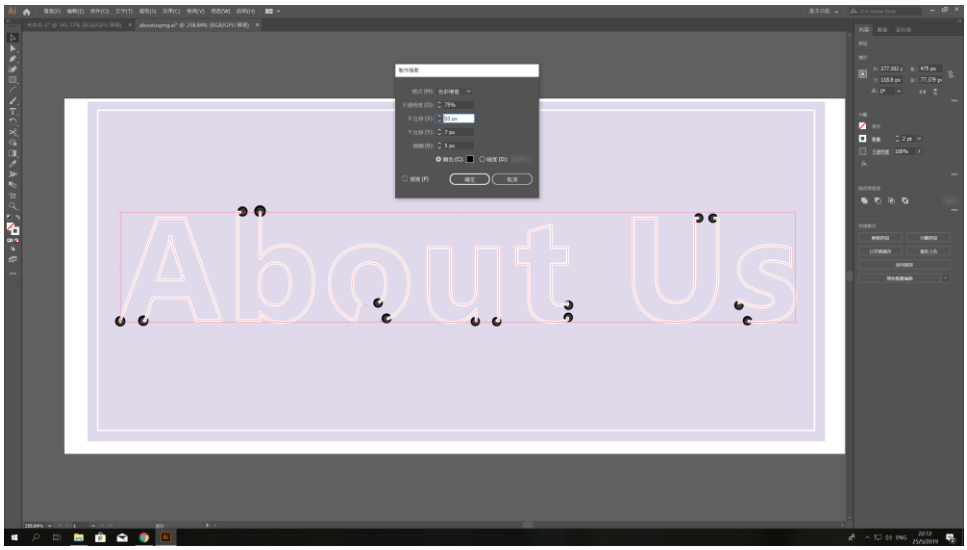


The logo is made of 5 layers of the outline of the word with different Gaussian blur (From 1px to 6px). By stacking all the layers on top of each other, a neon light effect word was created.

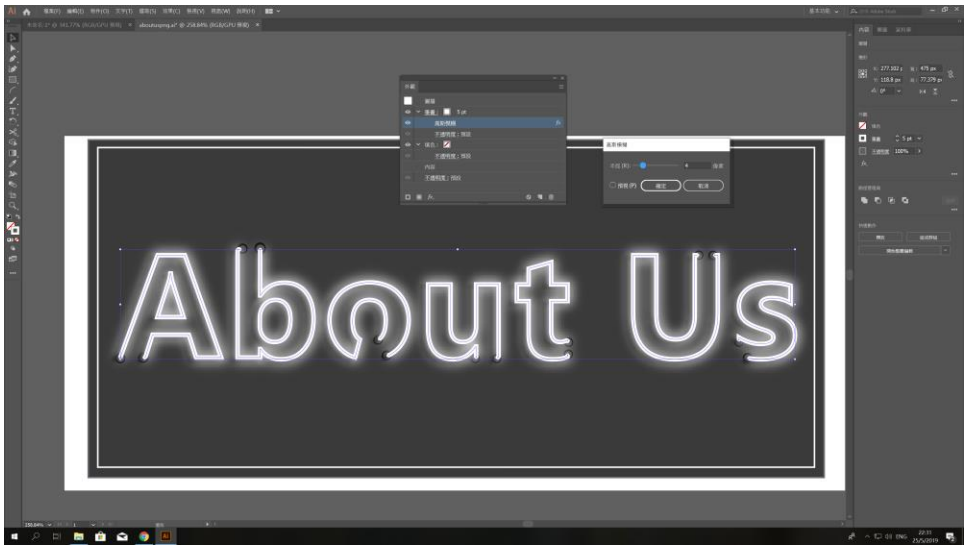




Here is the development process of the second custom PNG the team made. Just selected the right font type and turn it into stroke. Used scissors tool to cut out part of the stroke to mimic the real neon light box and fit it the connection sockets which were made my two circles laid on top of each other.



Then the overlay drop shadow was implemented to mimic the shows of a light box.



Then a copy but enlarged layer of the stroke was lay on top of the original stroke. The gaussian blur was then applied to the copied stroke to create the glowing effect.

This is the final product.

Server: The server-side of the application is supported by express.js. The express.js was structured using MVC like pattern, controllers and views were designed to support various functions that the website provide. Instead of building models, the server design used a service layer to deal with the data access.

The server was designed to provide RESTful API to guide user to certain resources. And some of these resources were protected with authorization. JSON web token was used to achieve authentication logic. The user will be automatically redirected to login page if no JSON web token (JWT) is provided. Then user could sign-in with username and password, the request form will be passed to user service layer and the password will be hashed and compared to hash stored in the database. if the information matched, a JWT will be generated and returned

to user. Then when making certain API calls (updating account etc.) the JWT will be passed for verification.

To further secure the information, the login and registration page were separated from angular application. So, all client-side angular application files will only be available for registered users.

The server also has scheduled job to update the news stored in the database. This is to reduce the workload of the server so that the news will not be update on every refresh of the webpage.

Database: Although it is not recommended in the unit. The team still choose to use MongoDB as it is an important part of the MEAN stack. The MEAN stack enables a perfect harmony around JSON: MongoDB stores data in a JSON-like format, Express and Node.js facilitate easy JSON query creation, and Angular allows the client to send and receive JSON documents. And the website functions that requires database support particularly favor the JSON format. For example, the RSS news feeds were converted to JSON for easy query and can be directly inserted into MongoDB.

There were two collections created for the website. The ‘users’, storing username, first name, last name and hashed password. The ‘news’, storing json object that converted from RSS news feeds. Mongoskin is used to connect express application and database. The Mongoskin is extremely lightweight and provided a simpler interface for CRUD operations. Sensitive data like password is encrypted using bcryptjs, only hashed passwords are stored in database.

The team understand that using the MongoDB will cause some issues on marking (not embedded and need MongoDB installation). Although we did an extensive work on database integration, we marked it as rank C for the inconvenience. Thus, to fully test the website, you may need to install the MongoDB, the default setup is used for the database.

Dynamic Webpages: The login and registration pages were templated with EJS. The footer and header, login and register component are in templates and could be dynamically rendered on demand.

To achieve certain functions (such as display latest news, update account information), the team created multiple Angular controllers, services on the client side and express controllers, services on the server-side. The controller will access functions provided by the services and request data from the server or post data to server. These get or post request will be passed to express controllers and then server-side service will be used to acquire or update data. JavaScript promises library Q was used to handle asynchronous operation and further streamline the events and callback logic and prevent callback hell. The responding data will be sent back and stored in variables in angular controller and could be inserted in html dynamically. The perfect example is the news page. All the news is retrieved from the database on server-side as in JSON and their title, descriptions could be easily parsed and inserted to html.

Problems and Challenges: The overall design of the website went through several versions during the development, some of the designed features were not included in the result for their performance issues.

During the development, the team were able to experiment many web development technologies and concepts. One clear lesson that we learn this time is choosing appropriate and updated tools. During the early development, Mongoskin is considered good for its light weight and close connection to native MongoDB driver. However, it is not updated for several years now. And its light weight also restricts us from building more complex functions.