WEBTECH FINAL REPORT

Rachel Kirby (rk16586) - Angus Williams (aw16797) - Bradley Horswell (bh15807)

Our website is called 'Planana'. The aim of the website is to help students find new recipes and plan their meals for the coming week, it can also provide a shopping list with all the necessary ingredients for the plan, alongside statistics on the overall nutrition. We wanted to create a website with a fairly simple aesthetic that was also playful and colourful, and was easy to navigate and interact with.

Our Grade Estimates:

HTML: A

We have developed a number of different html pages, with XHTML, using the server to redirect between each page. The pages include a variety of features such as tables, embedded images, and navigation bars. We have not used frameworks for our HTML pages but written them from scratch.

CSS: A

We have a number of stylesheets, including a main and nav which are connected to the majority of the HTML pages, and then some stylesheets which are specialised for particular pages. We have removed style tags and attributes from inside our HTML pages, instead restricting all styling to be within the linked stylesheets. We feel confident using our variety of CSS attributes, including animation features used for the welcome, login, and saved plans pages, as well as animated buttons across the site. We have used CSS to create continuous font styles and sizes, colours, and layouts. We also used flexbox to allow for dynamic resizing based on window size and the number of elements loaded from the recipe API.

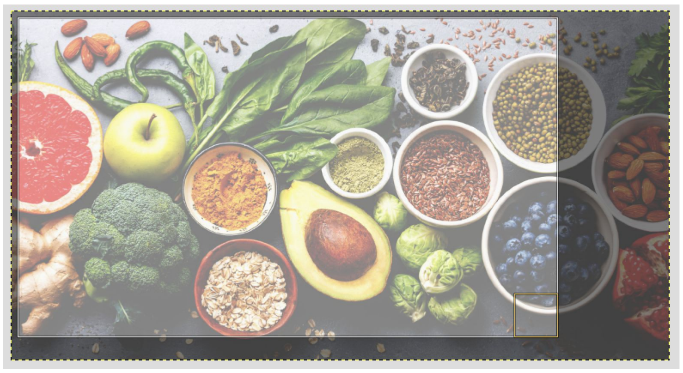
JS: A

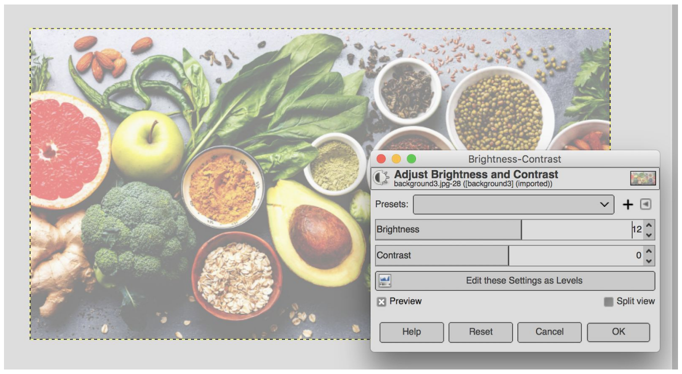
We have used client-side JavaScript pages to add interactivity and further animations to our pages. Some of our highlights include the drag and drop feature in the 'create weekly planner' page and the search meals function is 'search meals'. We have written numerous script pages from scratch to create these. In order to get recipes, we use the Yummly API (https://developer.yummly.com/), which aggregates websites. This was a pain point for some time, as, due to the asynchronous nature of javascript, we couldn't access the data for the recipes outside of each API call, which we managed to overcome.

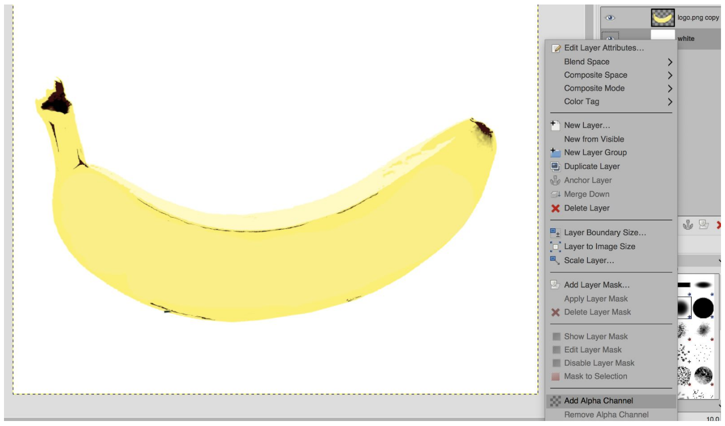
Our program also allows users to export a meal plan to PDF, which we did by using the jsPDF (https://parall.ax/products/jspdf) library, which proved a challenge to use in conjunction with the Yummly API calls, which, again, we eventually overcame.

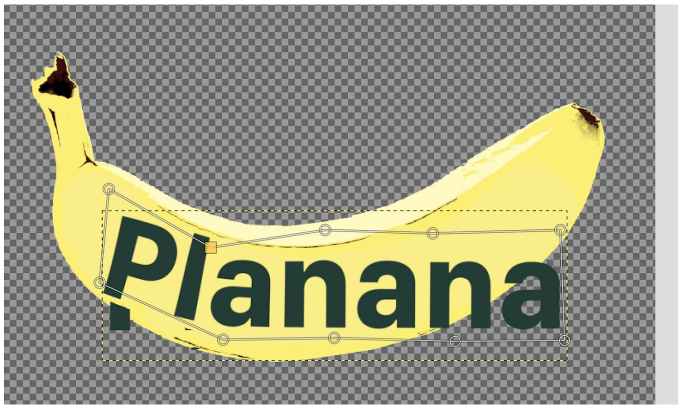
PNG: A

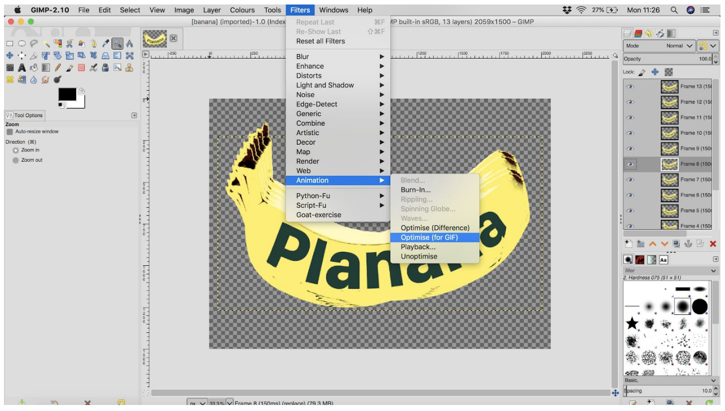
We used GIMP to create our 'Planana' logo, which can be seen on the welcome page and in the corner of the navigation bar. Creating this logo involved using a google image as a template for drawing the banana outline and colours. We then added text, using deform tools to fit this text to the shape of the banana. Then duplicated it over different layers which were rotated in order to create a gif of the logo moving. Additionally, we used GIMP to manipulate various background images used on our page, both using filters and the colour tools to tweak the contrast, brightness, exposure, etc. of images to make them ideal for background use.



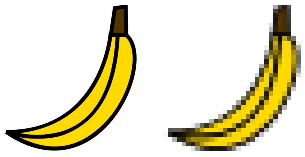

Brightness-Contrast
Adjust Brightness and Contrast
background3.jpg-28 ([background3] (imported))
Brightness 12
Contrast 0





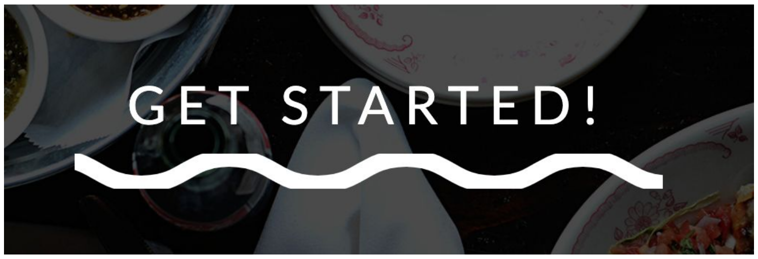


We also created a 32x32 pixel "favicon", of a banana, using bezier spline curves in inkscape, for use in the website's tab bar (see hi and low res versions below):

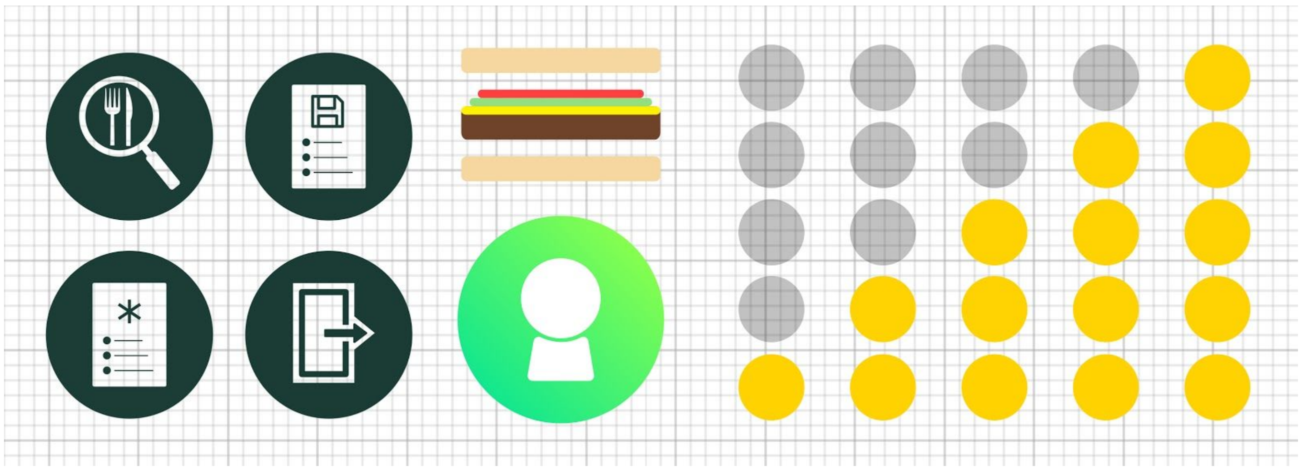


SVG: A

We created multiple original SVG icons using Vector Drawing Tool "Graphic" (https://www.graphic.com/) for use throughout the site, and used SVG animation to animate a "squiggle" underline for various text elements, visible below:



Our navbar links each have an icon, we created original graphics using paths, shapes, and bezier splines, and then used boolean operations with a circle background to create cutouts. For our login page, we created an icon using a color gradient background, and a human figure using bezier splines. We also created a "hamburger" menu icon in the form of a hamburger. For each recipe that is displayed, we assign a custom rating graphic, made up of circles, some with partial opacity, based on the rating of the recipe out of 5. Below are all of the SVGs we created.



Server: B
Our server uses the Express framework to implement a variety of routing and redirect functionality for serving up various web page files the user accesses. It also interfaces with the SQLite database and provides handling of various GET and POST requests that the client makes to update dynamically. The server implements various forms of both URL and data validation, such as preventing users accessing anything but the landing and welcome pages until they are logged in, 404 error generation for undefined page access and validation of signup and login credentials. Cookies are used to store dynamic session information associated with user login. This is mainly used to prevent users accessing any restricted pages that are only accessible once logged in as well as remembering the user is logged in so that they do not have to go through the landing pages again. We did attempt to implement a https server version however this connection is currently deemed insecure by the web browser due to using a self-signed certificate and so it is recommended to load the http version instead.

Database: A
We have gained a significant amount of experience with SQL including inserting and extracting our data from a variety of different tables with appropriate callbacks afterwards. We have tried to organise our database in a way so that data is not repeated unnecessarily, by utilising various unique ids that act as foreign keys to link tables together. For example the plans table only stores the unique meal ids associated with each plan, which can then be linked to the meals table which contains the additional meal information such as image URLS. We then further utilise these to ensure data is captured in single SQL statements using various WHERE and JOIN clauses. This is beneficial as the logic for data retrieval is completely contained with the database functionality and the server does not have to do any extra logic, matching fields together or filtering. Finally in terms of security we use prepare statements to prevent SQL injection as well store passwords via hashing with a salt so that they are secure and unreadable.

Dynamic pages: B
We delivered dynamic pages on our website by requesting data from the server and inserting this in to our existing pages. This is implemented on both the 'create week plan' page and 'saved plans' page, which make requests to the server to find the recipes and plans the user saved and insert data dynamically into the page. Additionally client side scripts call the Yummly API to dynamically generate meals for the user to save in the "find recipes" page. These pages are therefore individual to each user depending on they're present and past saves of various meals and plans. Additionally the drag and drop feature allows the user to dynamically alter the meal plan grid for the pages.

Website Walkthrough

On starting up our website on a local host *(node server.js)* you will see that you can visit two pages. One is our secure page *(http://localhost:4000/)*, as we have been working on a local host we have written our own certificate rather than getting an official certificate. Therefore you need to use advanced settings in order to access the page, and a number of the websites features are blocked. We therefore have kept an non-secure option available *(http://localhost:3000/)* which can better demonstrate the websites features.

The website has been designed for desktop-first use, and we've tested it over a number of browsers, aiming to use the correct forms of sizing (em, %, or pt, dependant on object being sized) in order to keep the website efficient despite differing screen sizes and quality.

On entering our page you will be taken to the **"Landing"** page. On scrolling downwards you can find a short 'about' statement and the 'Get Started!' button that will take you to the **"Welcome"** page. On this page, users will be encouraged to either create a new account, or login with a past account.

Once you have logged in you will be able to see the main pages of the website - "Search Recipes", "Create a Plan", and "Your Saved Meals" - which are accessible via the navigation bar. For a new account, the 'Create Plan' and 'Saved Meals' pages will print some inner HTML confirming the account has not yet saved any meals or plans.

On the **"Search Recipes"** page, users are able to search any meal they are interested in. A large variety of options will be returned, and the user can either find the recipe page by clicking on the recipe title, or they can save the meal to their account by clicking the 'Save' button. They can also see the rating of each meal, visually demonstrated by the yellow circle bar on the recipe card.

After saving a variety of meals, the user can move on to the **'Create a Plan'** page, which will now show any of the meals that the user saved. The user is able to drag and drop the meals into the table, creating their plan for the week. If the user wants to remove a meal, they can do so by clicking the box the meal is in, for which an alert will pop-up confirming the removal of the meal. The user can also change the placement of any meal by simply dragging it to a new space. If a box is already full the user will not be able to move a new meal there, and an alert will pop-up explaining so.

Once they are finished creating the weekly plan, the user can save the plan via the 'Save' button, another alert will appear urging the user to choose a name for the plan. They will be unable to save the plan until they have done so.

The plan will be stored in the database for their account and they will be able to view it by visiting the **‘Your Saved Plans’** page. All different meal plans will be found here, clicking the plan name will show an overlay of the meal plan they created, and will offer an “export” button. Clicking the ‘export’ button will download an ingredients list for each meal in their plan to the users computer. It also shows the nutritional value of the website.

The user is able to logout at any time using the ‘Log Out’ option in the navigation bar, this will take them back to the landing page.