Clothing Site (Chaos Clothing) - Planning and Design Doc

What is it about?

The project in question is an online shop for clothing where people can browse and shop for different items. The shop will function like a normal clothing store so users will be able to look at different items, filter them based on their size, color, brands etc…

Who is it for?

This is accessible for anyone who wants to view and browse an online collection of clothing, look at inspiration for their outfits, even try and shop for pieces of clothing.

Relevant Implications:

Website:

Aesthetics: Aesthetics are how the website looks to the user. The color palette, the UI and everything to do with appearance.The website needs to look good and have a good looking user interface. The colors need to be simplistic in order to make the pieces of clothing stand out and not be distracting for users. This was taken into consideration by addressing it as the color palette of the website is plain black and white which causes the colorful items to show on this background compared to if the background was colored.The layout of each page is done specifically so that the important information is there while maintaining a simplistic design

Functionality: Functionality is how the website is supposed to work. This is important because if our final product doesn't work properly then ultimately, the project is unsuccessful. For our project to work we need to make sure the pages display all the data that we want it to display, for all the links to go to said pages and for all our queries to work. I addressed this by testing my queries and making sure they returned the data that I wanted. And making sure that said data was displayed correctly. Another way I made sure to address this is by making sure my links worked and that if there are non existent pages that an error message would show

End user Consideration and Social Implications: Things that the end user would like to know and be informed about when coming online onto my website.

Clothing Site (Chaos Clothing) - Planning and Design Doc

As this is a clothing store they need to know the colors of the items, the price and what the items look like. We also need to keep in mind that some people might be disabled in different ways so we need to make sure we add things that would make viewing the website easier for them. I addressed this by adding colors in words on the pages in order to make it easier for people with color blindness to shop.Another way the website is user friendly is that there are pages which help the user get back to the home page if they end up going astray.

Database

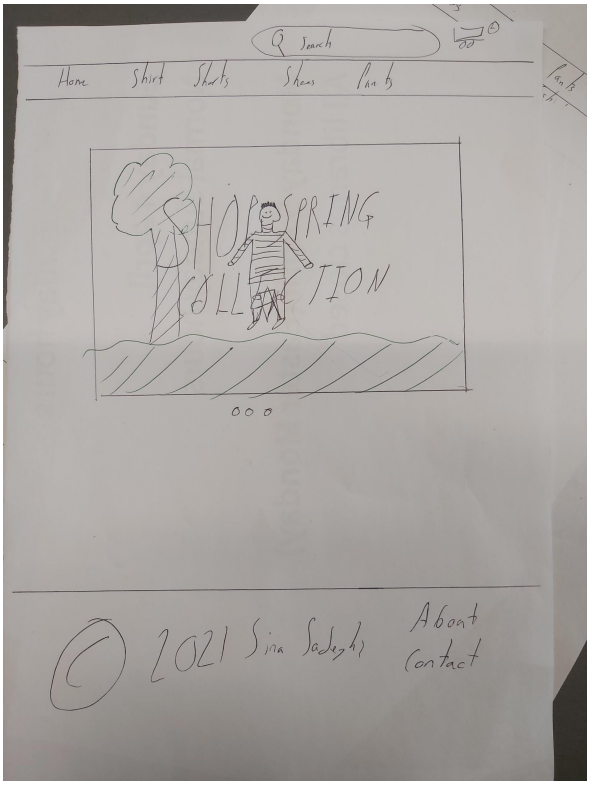
Intellectual property: This means that any information used in my database or website is information/photos that I am legally allowed to use under the trademark and copyright acts. I made sure to do this in my database by writing my own descriptions for each item of clothing. By doing this I addressed this relevant implication by making sure that my descriptions were original and mine. This ensures that my website has not stolen any description or information from any other sources. I also removed the brands table from my original plan due to the decision to use licensable images.

Sustainability and Future proofing: This means that the program and database will live on for future use. I addressed this relevant implication by making sure that if a clothing item needed to be added it would just require a new entry in the database along with assignment to colors. This makes the project future proof as it is easy to add more and relevant information. The data in this website also doesn't need to be updated constantly as the data never changes due to these items being set items and not changing themselves. This further adds more to the future proofing.

Clothing Site (Chaos Clothing) - Planning and Design Doc

What would you expect to see?

The font planned to be used throughout the project is Verdana.
The heading will display the logo of the site to the left (size 20) and have a search bar to the right, with a shopping logo where you can check your cart from any page.
Below that will be a navigation bar where you can select certain categories of clothing, this will be black with bold white text (size 15). The nav bar will be based on categories of clothing (sweater, pants, shirts, etc…)) and also have the home page.
The footer. will be white background with bold black text (size 15). It will also have an about page and a contact page where people can read about the site and contact us if needed.
The Contents of the page's will be just plain black text with a white background, if any subtle details are to be added (like price under name) it will be in a grey font.



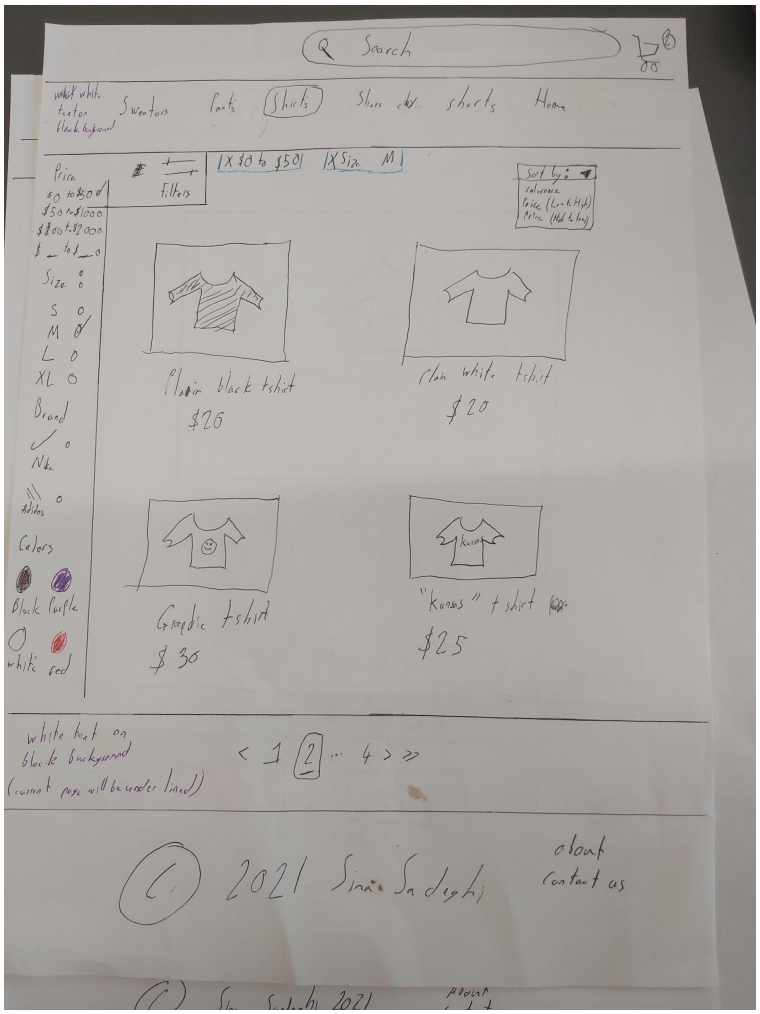
What pages would you want?

1) **Home Page**

There will be a Home page where you can scroll through new arrivals, sales, etc. in this page there will be

Clothing Site (Chaos Clothing) - Planning and Design Doc

2) Shirts Page (Page for any category on the navigation bar except home)

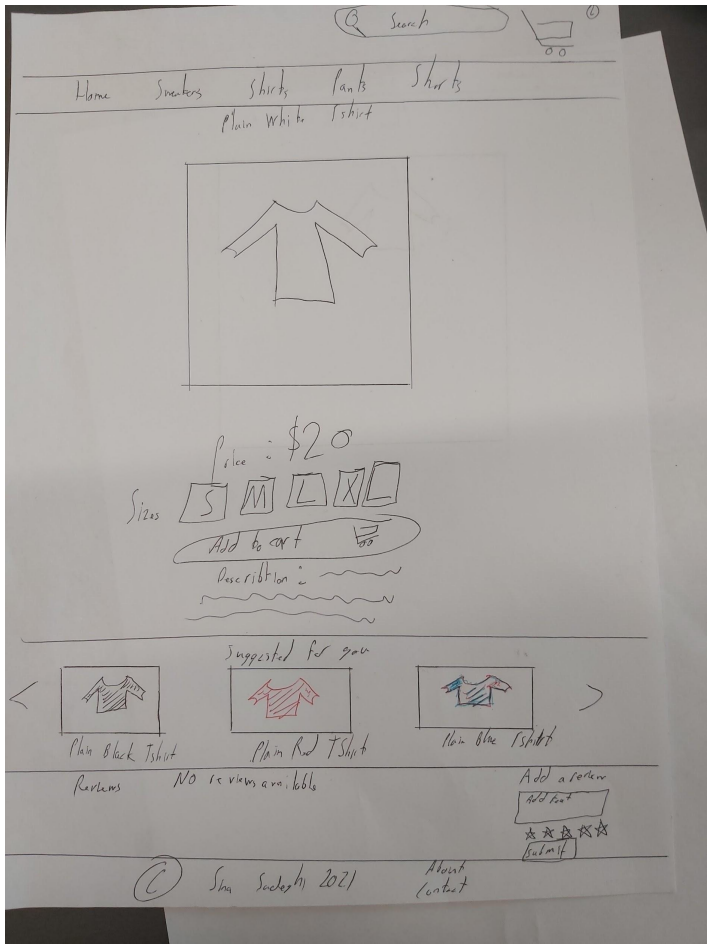


Once you click on the clothing type that you want through the nav bar (eg, shirts) it will take you to “/all_shirts” where it will display all the shirts “def (all_shirts)” There will be a filter menu where you can filter the results based on criteria like size, color, brand and price. (Please refer to the design below). This process will continue for every category, and when filtered, the filters will pop up on top of the page, and can be deleted at any possible time by clicking the “x” next to the filter.

This could be made possible with “/shirt/<price = int>”. There will be an option to sort based on price and relevance (id). Each page will have 4-6 pieces of clothing displayed with their name and price in a grey and smaller font underneath.

Clothing Site (Chaos Clothing) - Planning and Design Doc

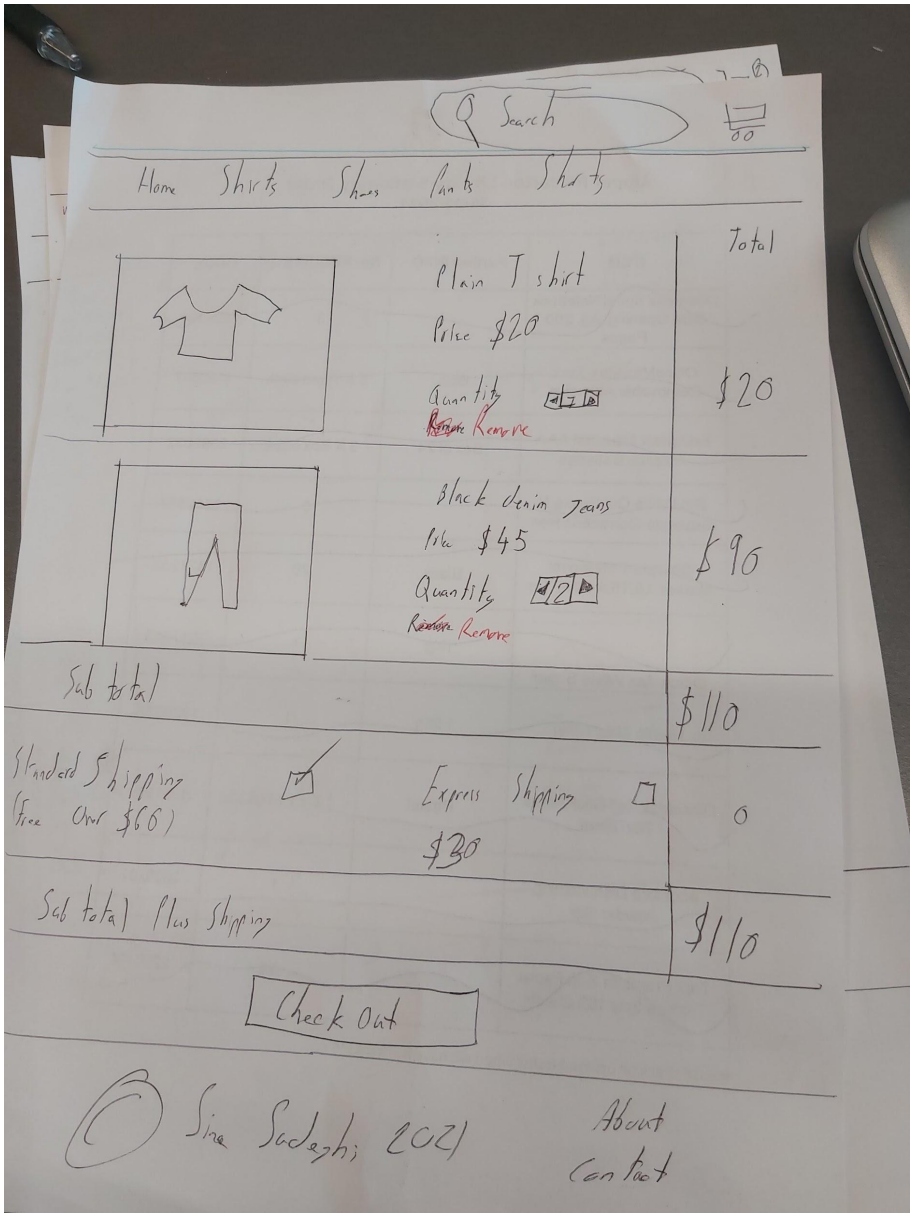
3) A page for any 1 piece of clothing



When you click on an item of clothing it will take you to the page of said item (it will take you to the id of that shirt “/shirt/<int:id>”) and will allow you to select the size, see descriptions, prices, even suggestions based on the item. From there you can add said item to your cart. This will happen for every piece of clothing in my database.

Clothing Site (Chaos Clothing) - Planning and Design Doc

4) A checkout page which displays a total with shipping.

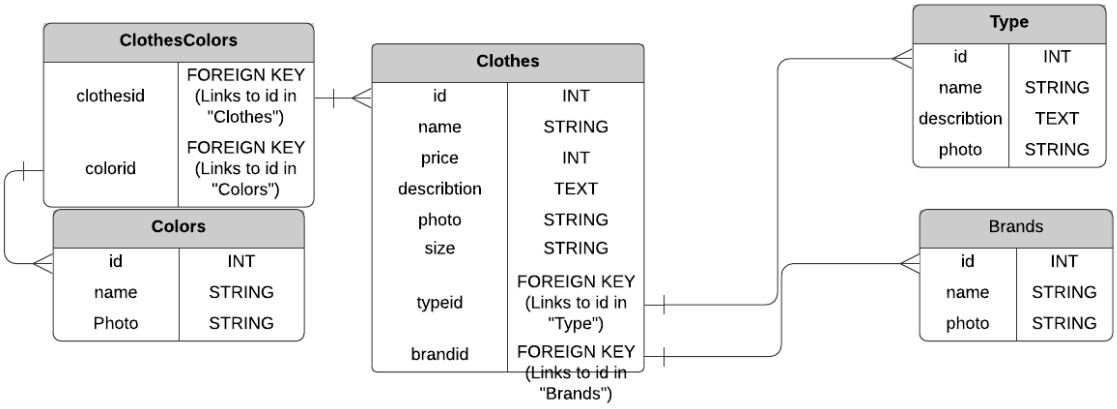


In this page all items that are in the cart will be displayed and you can increase the quantity, remove it, and choose the type of shipping etc… everything a normal shop can do without actually checking your items out and entering your details. (This isn't a massive priority though)

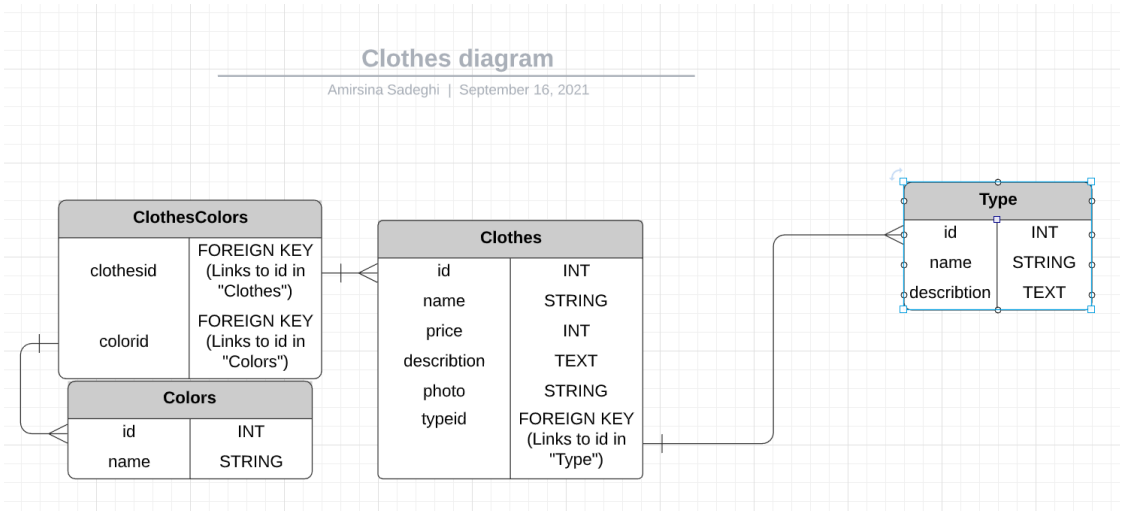
Below is a diagram of the database planned to be used in the website. In this database clothes are

Clothing Site (Chaos Clothing) - Planning and Design Doc

categorized by different types, brands and colors. This will be used to filter the site and results..



During development I realized that having a brand table isn't as necessary as most of the clothing used is unbranded for licensing purposes. So I deleted the brands table and its connection to brandid. Also during development I decided to delete the size column as I decided to make this site less of a shop and more into a viewing website. I also removed the photo columns from the "Colors" and "type" table as I realized they served no purpose. The updated plan can be seen below:

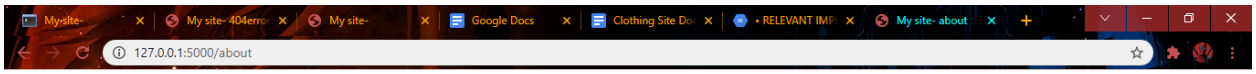


Clothing Site (Chaos Clothing) - Planning and Design Doc

Iterative Development and improvement(Breaking the project up into manageable chunks):

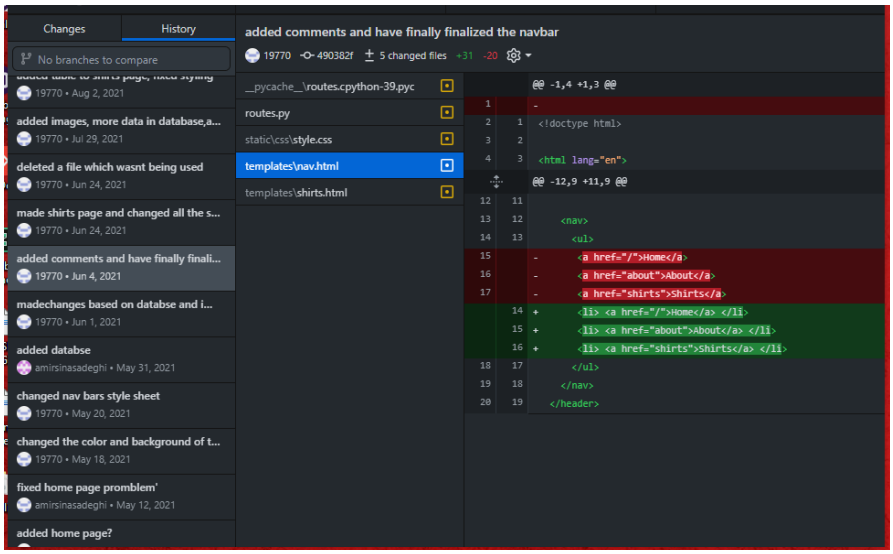
1) Add a nav bar which links to the different pages:

To do this we need to create a navbar which links to different pages. I did this by having the words link to the pages by using hyperlinks. Then I styled it to make it match the black and white color theme. I then added a hover effect to make so that the user knew where they were clicking. The final product can be seen below.



Chaos Clothing

Home Shirts Pants Sweaters Contact

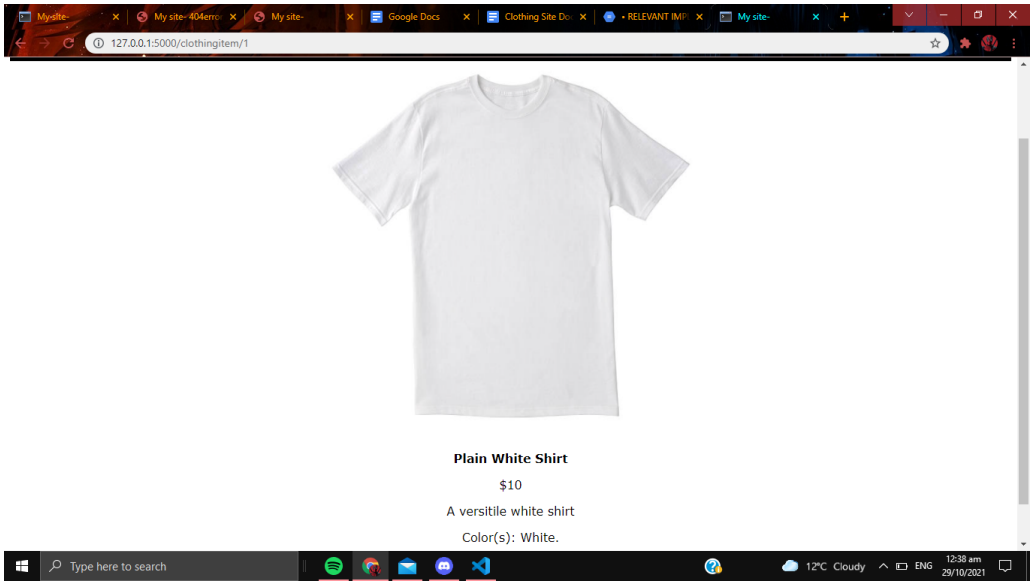


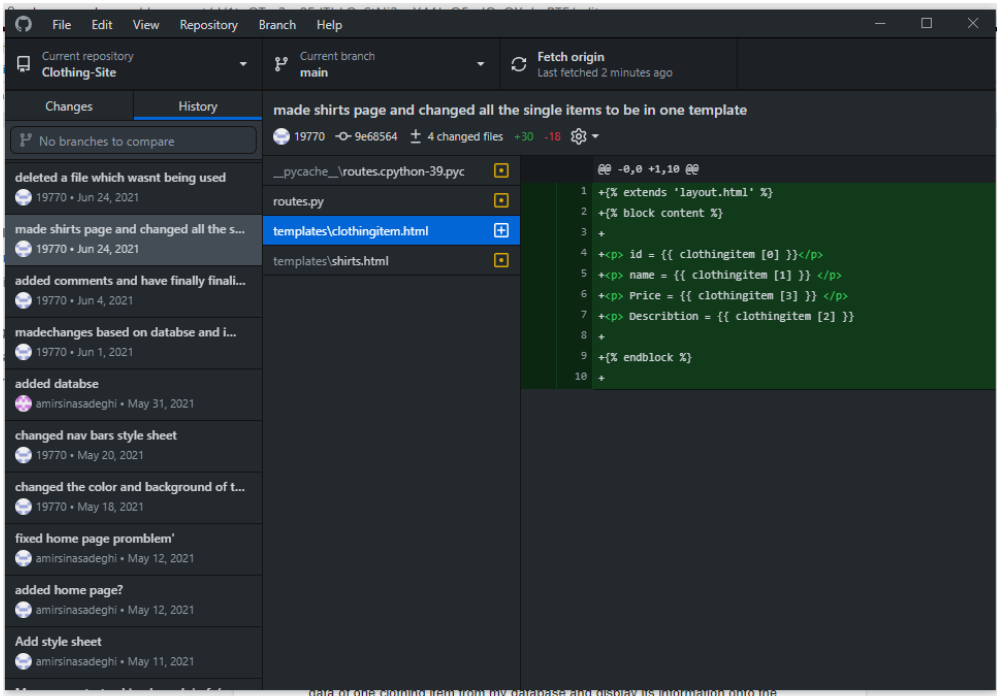
2) To have a page display 1 singular item of clothing:

This is the most important step as the whole point of the website is to display different clothing pieces and if this page doesn't work or doesn't look appealing, this ultimately defeats the purpose of the website. To do this I needed to pull the data of one clothing item from my database and display its information onto the page. I would grab all the info from the id of that clothing item then choose to display the data in different rows as this was the neatest way to do it. I then

Clothing Site (Chaos Clothing) - Planning and Design Doc

added CSS to make all the words and photos centered and make the name bold. The finished product is seen below:



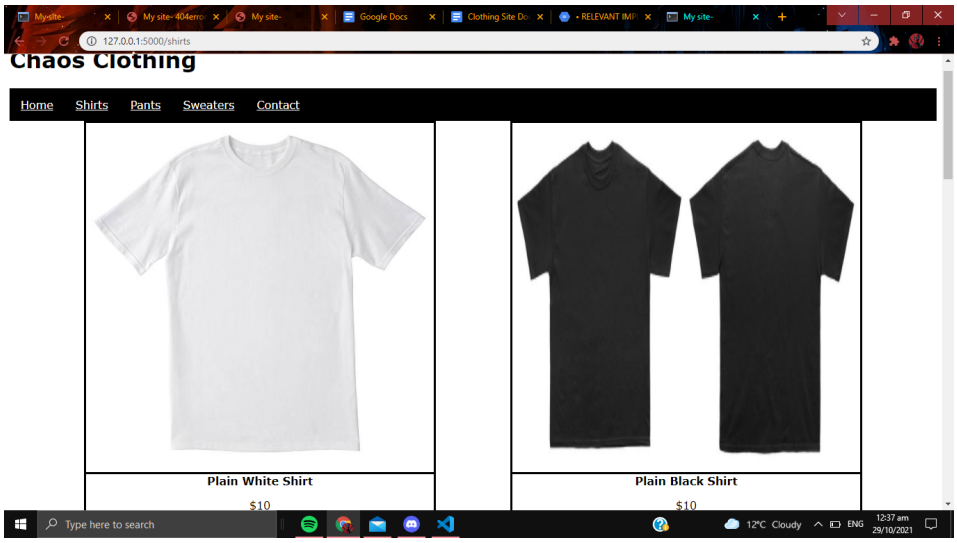


3) To display all the the types of clothing :

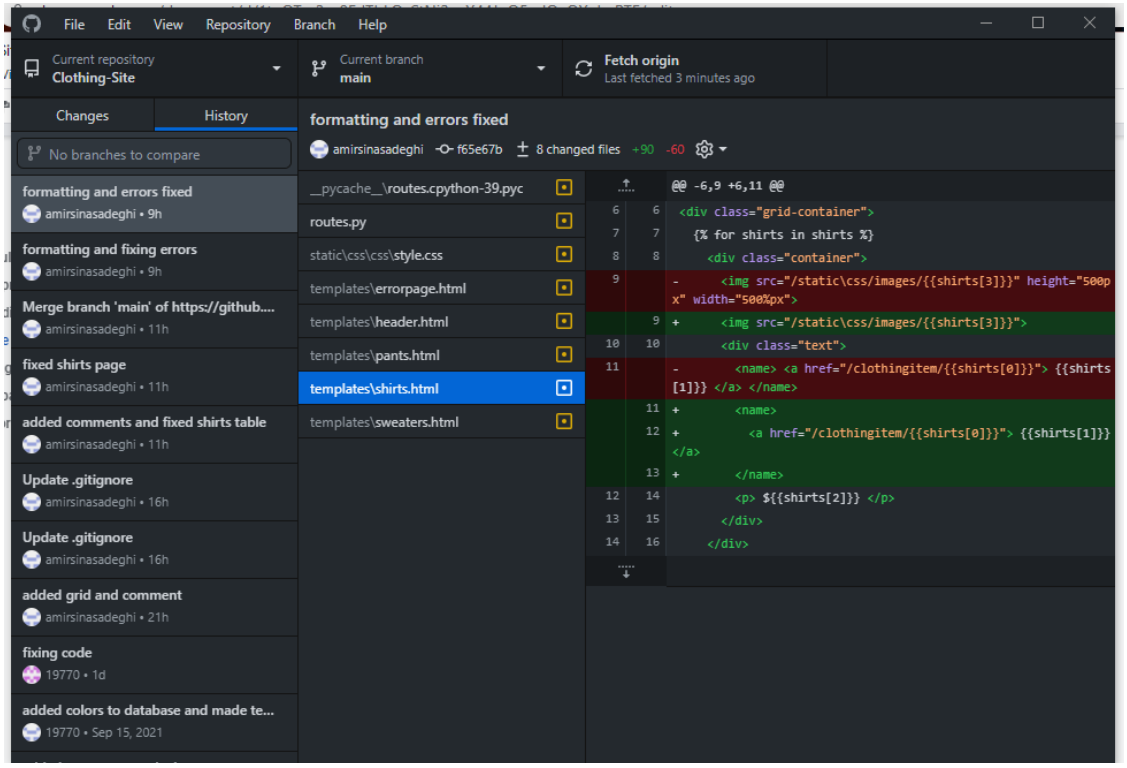
In order to do this I made a foregin key in the clothing table labelled “type id” then I made a type table. I then assigned the type id of each item of clothing to the respective item. In order to display all the clothing I made, I pulled all the data from the clothes table that had a specific typeid (for example shirts = typeid 1) then displayed the information. As I only wanted the minimal important

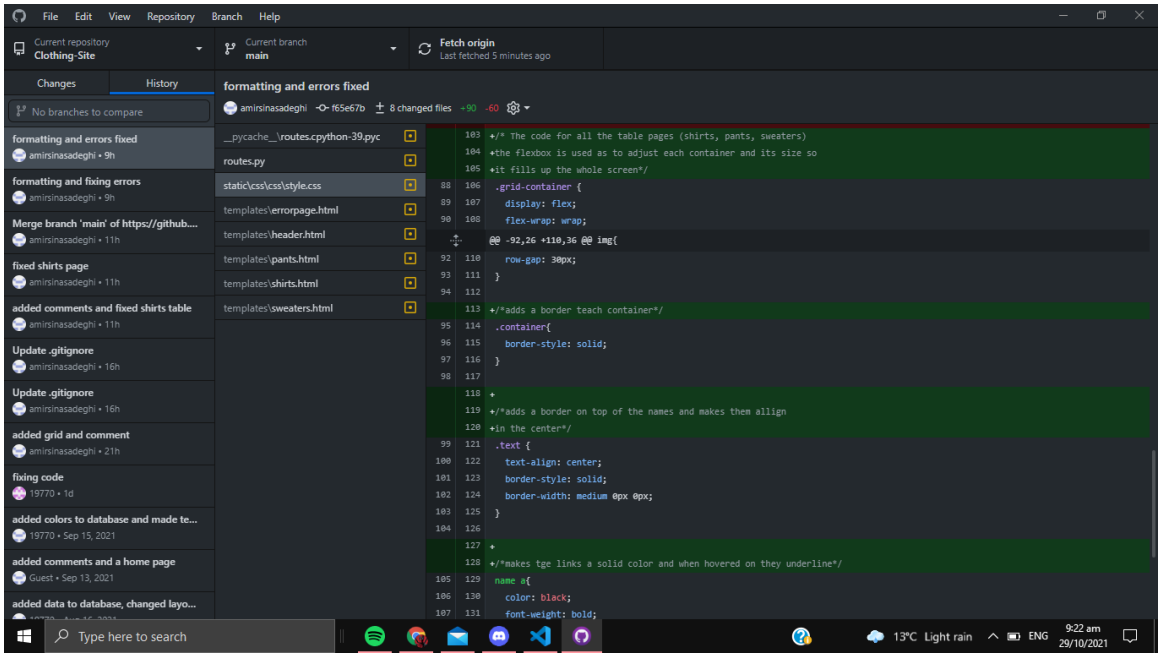
Clothing Site (Chaos Clothing) - Planning and Design Doc

information I decided to display the name, price and photo of each piece. I then made the name of the item link to the item page i created above. I improved this by using CSS flex boxes and containers to improve the overall aesthetic of the website. At first I had all of these displayed on a table but due to the outdated method I decided to use flex boxes. This can be shown in my CSS. After making and finalizing this for one of my pages I just needed to copy and paste this template to my other Item pages.



Clothing Site (Chaos Clothing) - Planning and Design Doc

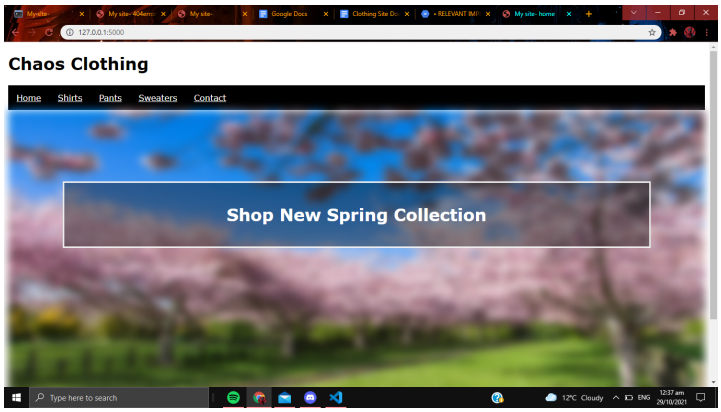


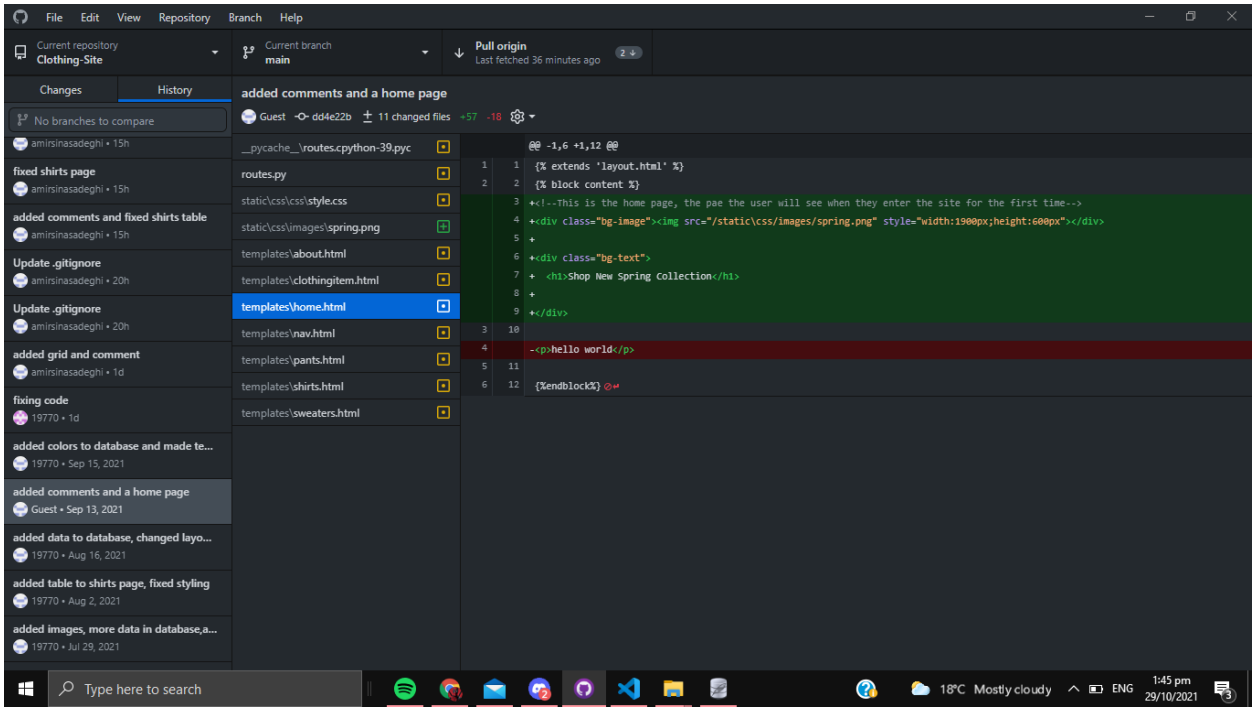


Clothing Site (Chaos Clothing) - Planning and Design Doc

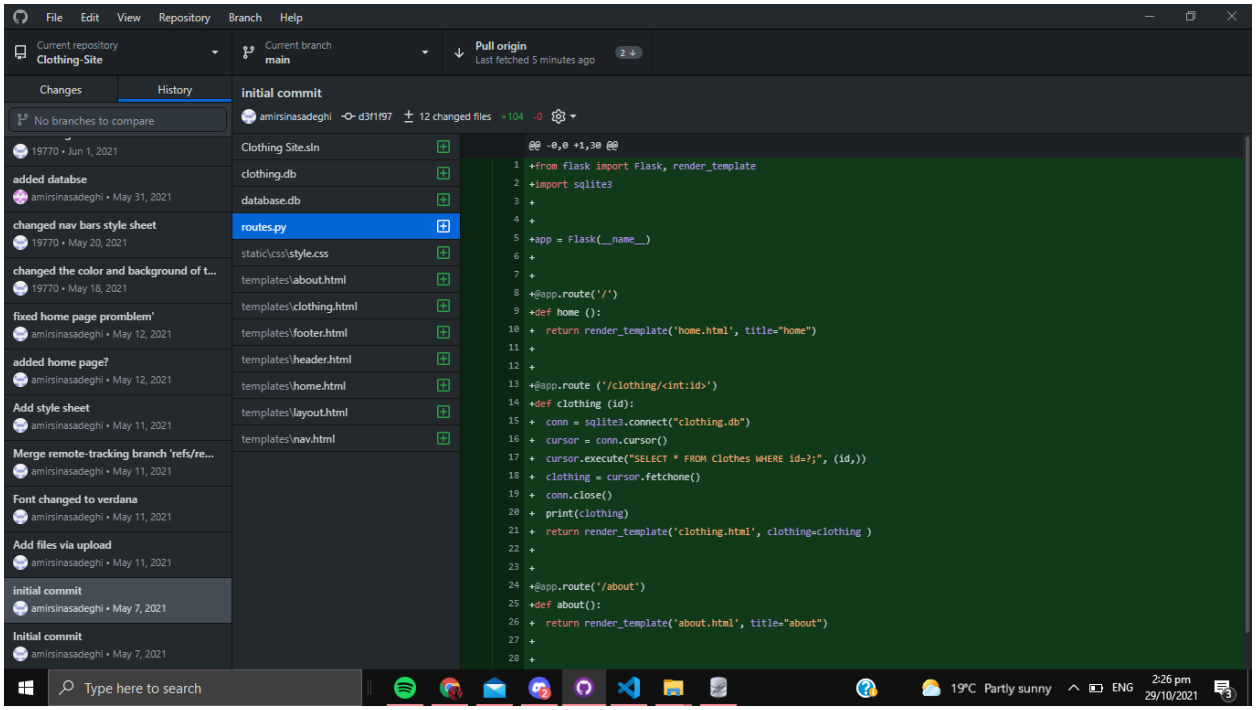
4) Make a homepage:

I wanted a homepage that attracts attention so I made a blurred image and added some text on top of it. This mimics other sites and what they do during season changes. Since I made this in October it was suitable to make the blurred image a spring themed image. Then using CSS I made the image fit the entire screen. The next step was creating a text box and adding the text to it, making the box overlap the image itself. The final product can be seen below:



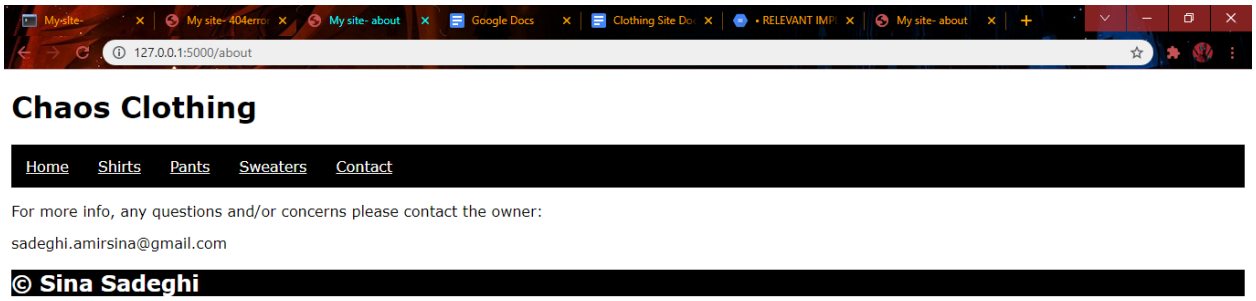


Clothing Site (Chaos Clothing) - Planning and Design Doc

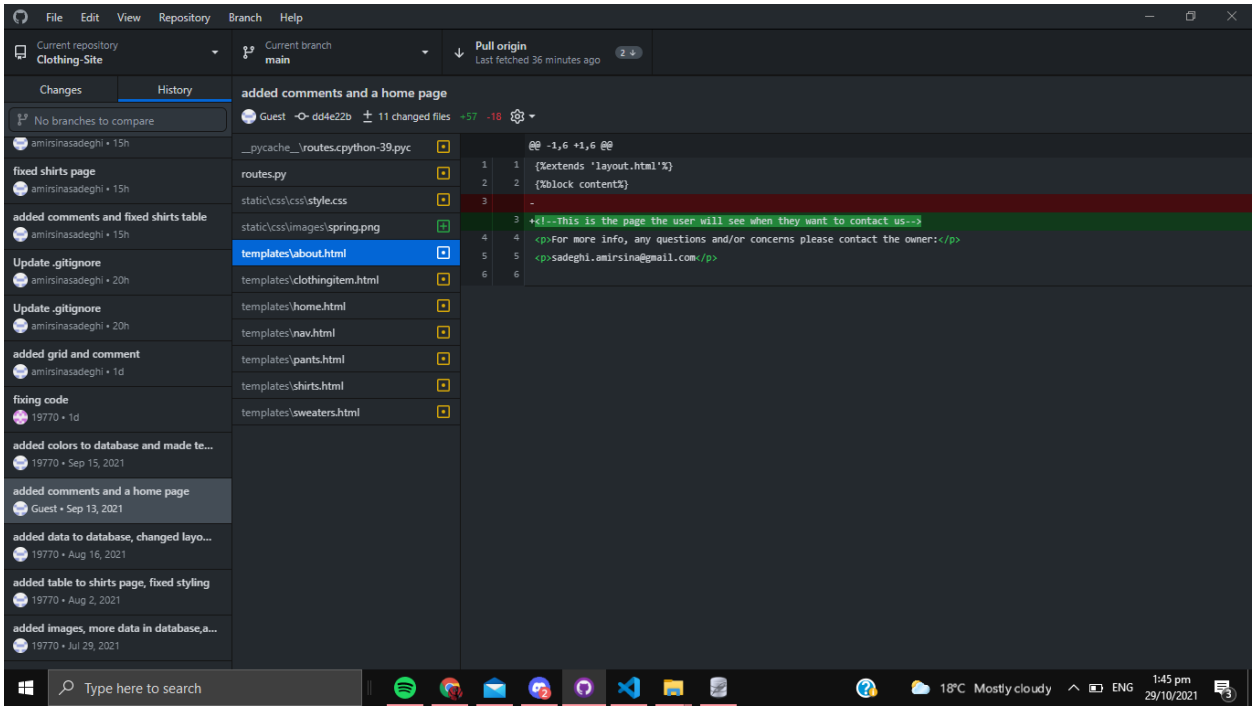


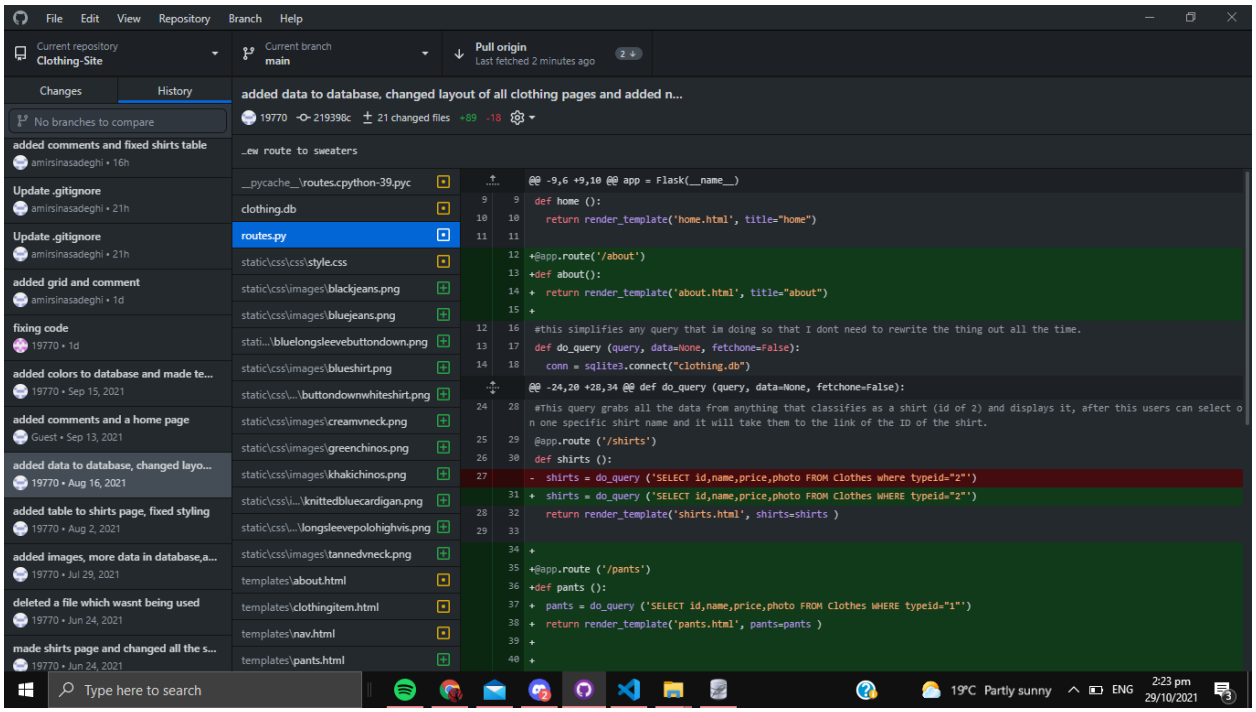
5) Make an about page:

This page just consisted of the app route and 2 lines of written text that gave out my contact details if needed.



Clothing Site (Chaos Clothing) - Planning and Design Doc



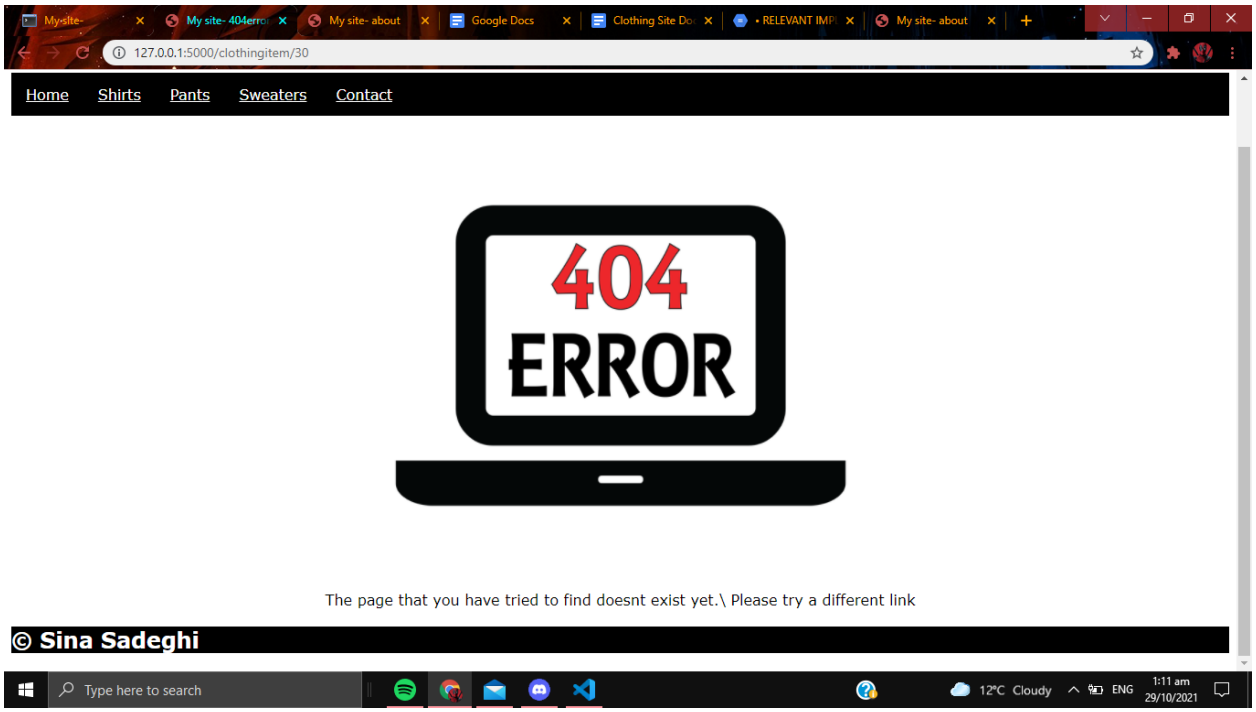


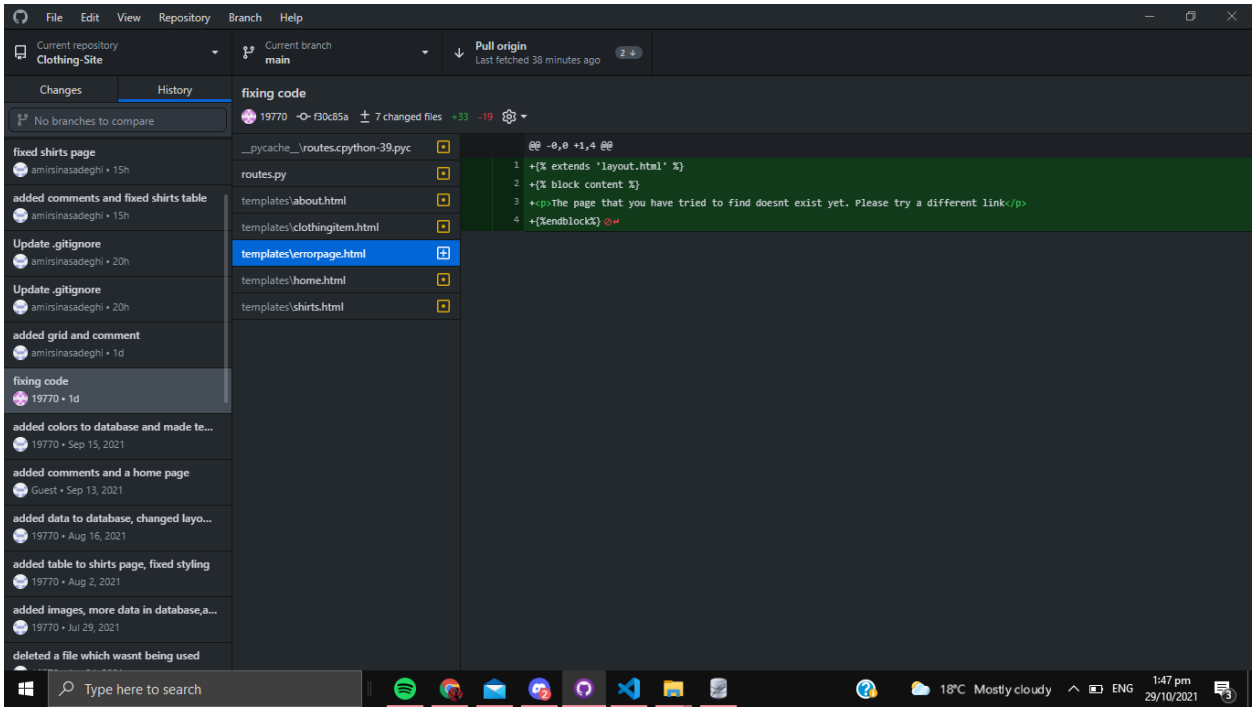
6) Make a 404 error page

This is important because this causes users to be redirected to a different page which helps them realize that the url they have tried to access does not exist. To do this i made an app route with a error handler then connected it to my routes and added it to my clothing item page (assuming the id is bigger than 15) due to

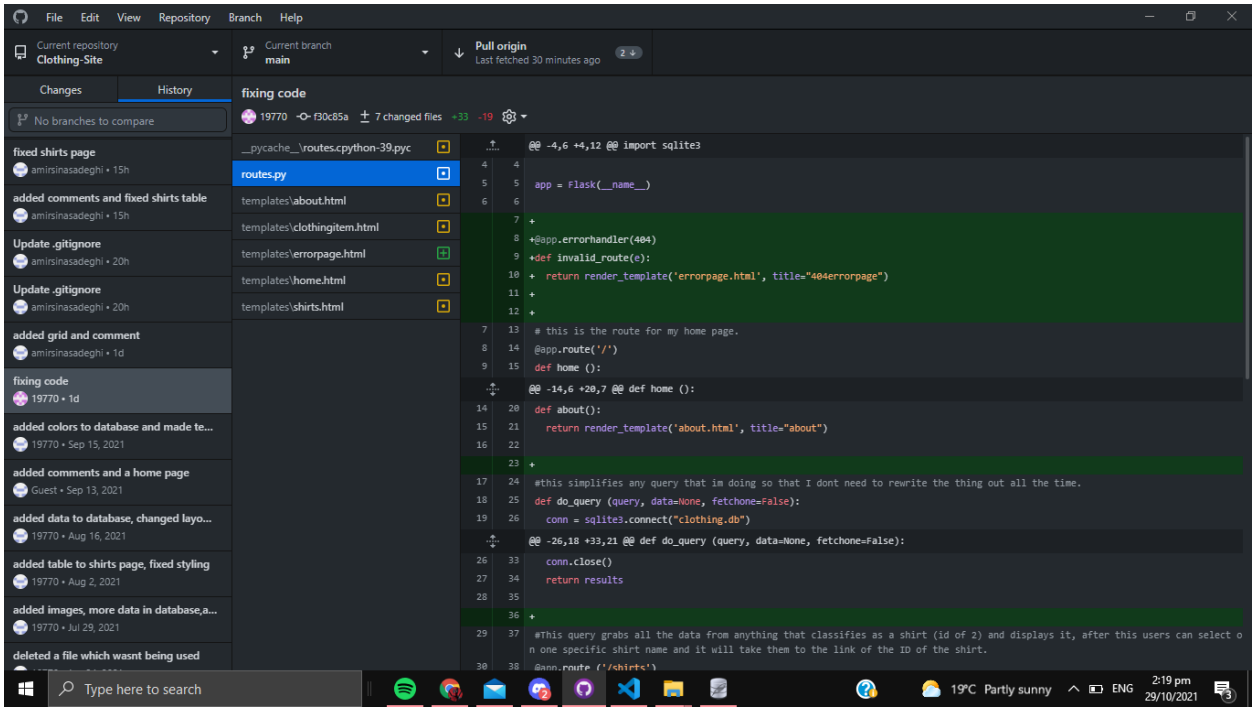
Clothing Site (Chaos Clothing) - Planning and Design Doc

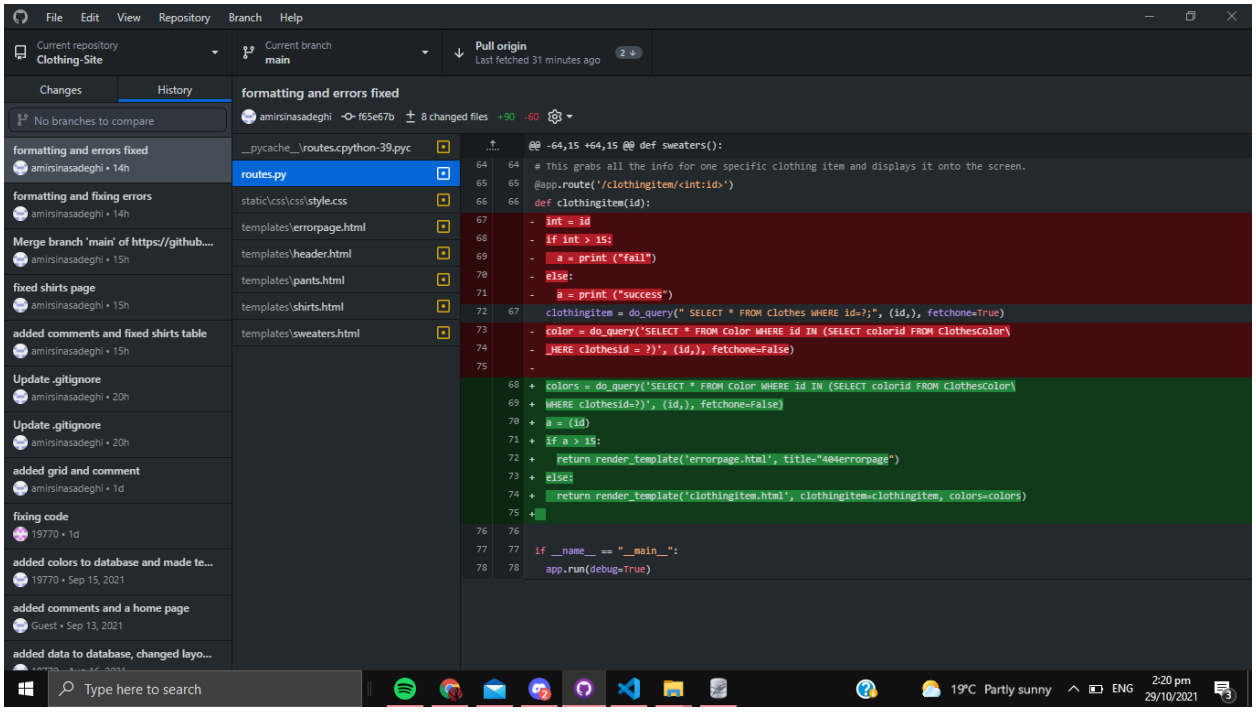
the fact that if users enter a id above 15 they'd get a empty page, but the use of this would cause the error message to appear. To do this I did 3 different things. In my first chunk I made the app route which would handle any 404 errors. Then I made a template which is the content which is displayed in the error page. For my final development stage I made an if statement in my clothing item which said if the id is more than 15 it would run the error page.





Clothing Site (Chaos Clothing) - Planning and Design Doc





Code Snippets:

Clothing Site (Chaos Clothing) - Planning and Design Doc

1) General query which make writing other queries simple

```python

#this simplifies any query that im doing so that I dont need to rewrite the thing out all the time.
def do_query (query, data=None, fetchone=False):
    conn = sqlite3.connect("clothing.db")
    cur = conn.cursor()
    if data is None:
        cur.execute(query)
    else:
        cur.execute(query, data)
    results = cur.fetchone() if fetchone else cur.fetchall()
    conn.close()
    return results


```

2) App routes for the error, home and about pages

```python

@app.errorhandler(404)
def invalid_route(e):
    return "Invalid route."


# this is the route for my home page.
@app.route('/')
def home ():
    return render_template('home.html', title="home")

#this takes you to a page which allows the user to contact the creator
@app.route('/about')
def about():
    return render_template('about.html', title="about")

```

3) App route and query for one single clothing item, its information and colors

routes.py | about.html | clothingitem.html | clothingitem.html (Working Tree)

routes.py > ...

```python
# This grabs all the info for one specific clothing item and displays it onto the screen.
@app.route('/clothingitem/<int:id>')
def clothingitem(id):
    clothingitem = do_query(" SELECT * FROM Clothes WHERE id=?;", (id,), fetchone=True)
    colors = do_query('SELECT * FROM Color WHERE id IN (SELECT colorid FROM ClothesColor\
    WHERE clothesid=?)', (id,), fetchone=False)
    a = (id)
    if a > 15:
        return render_template('errorpage.html', title="404errorpage")
    else:
        return render_template('clothingitem.html', clothingitem=clothingitem, colors=colors)


if __name__ == "__main__":
    app.run(debug=True)

```

4) The query and app routes for all my general pages (shirts, sweater and etc)

Clothing Site (Chaos Clothing) - Planning and Design Doc

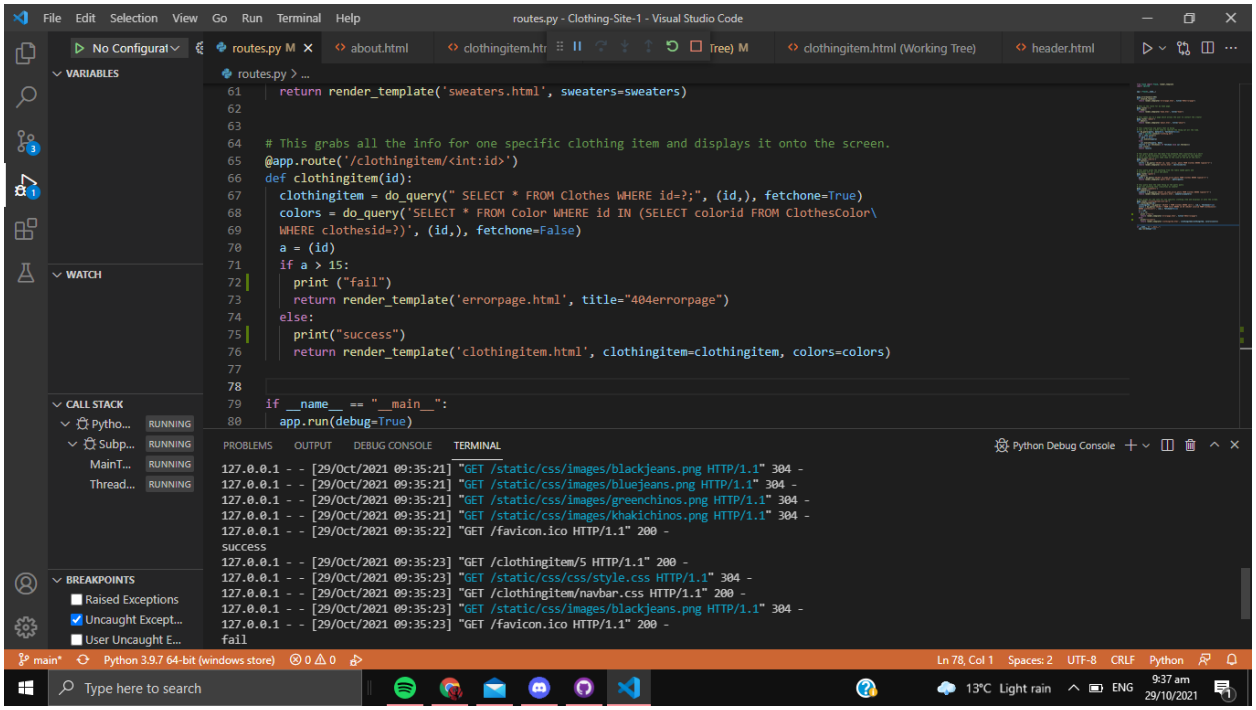
```
rminal  Help                                   routes.py - Clothing-Site - Visual Studio Code
routes.py ×
routes.py
35
36
37  #This query grabs all the data from anything that classifies as a shirt (id of 2) and displays it, after this users can select on one speci
38  @app.route ('/shirts')
39  def shirts ():
40    shirts = do_query ('SELECT id,name,price,photo FROM Clothes WHERE typeid="2"')
41    return render_template('shirts.html', shirts=shirts )
42
43
44  #this query grabs the anything from the table named pants and displays its id, price and photo
45  @app.route ('/pants')
46  def pants ():
47    pants = do_query ('SELECT id,name,price,photo FROM Clothes WHERE typeid="1"')
48    return render_template('pants.html', pants=pants )
49
50
51  #this query does the same thing as the pants query but for anything that classifies as sweater
52  @app.route ('/sweaters')
53  def sweaters ():
54    sweaters = do_query ('SELECT id,name,price,photo FROM Clothes WHERE typeid="3"')
55    return render_template('sweaters.html', sweaters=sweaters )
56
```

5) The home pages content

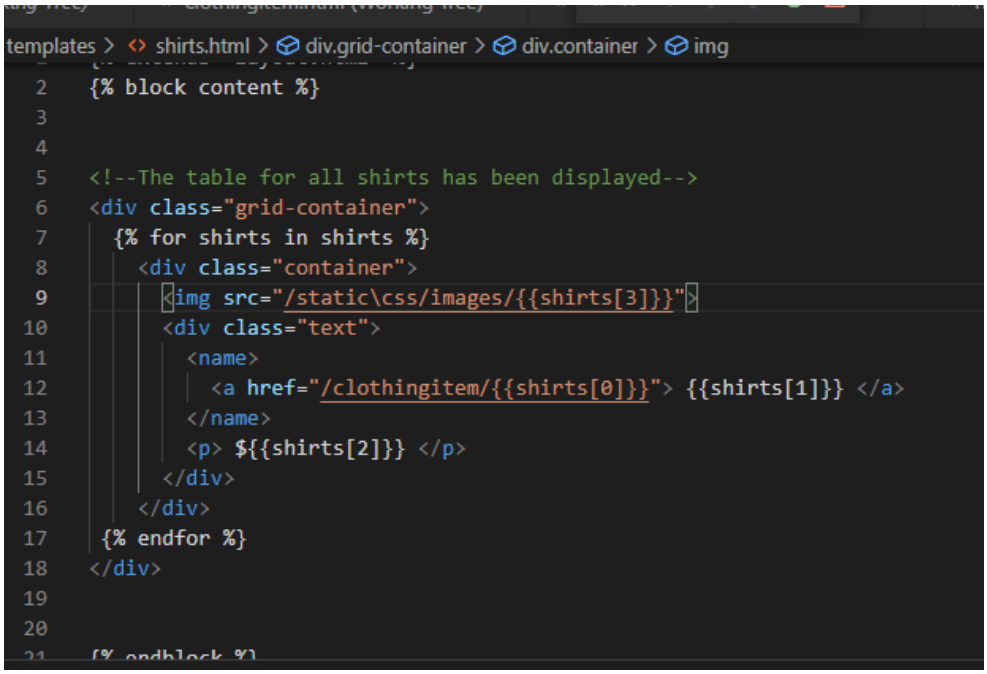
```
Terminal  Help                                   home.html - Clothing-Site - Visual Studio Code
home.html ×
templates > home.html > ...
1   {% extends 'layout.html' %}
2   {% block content %}
3
4
5   <!--This is the home page, the pae the user will see when they enter the site for the first time-->
6   <div class="bg-image"><img src="/static\css/images/spring.png" style="width:1900px;height:600px"></div>
7
8
9   <div class="bg-text">
10    <h1>Shop New Spring Collection</h1>
11
12  </div>
13
14
15  {%endblock%}
```

6) The content of a singular clothing page

Clothing Site (Chaos Clothing) - Planning and Design Doc

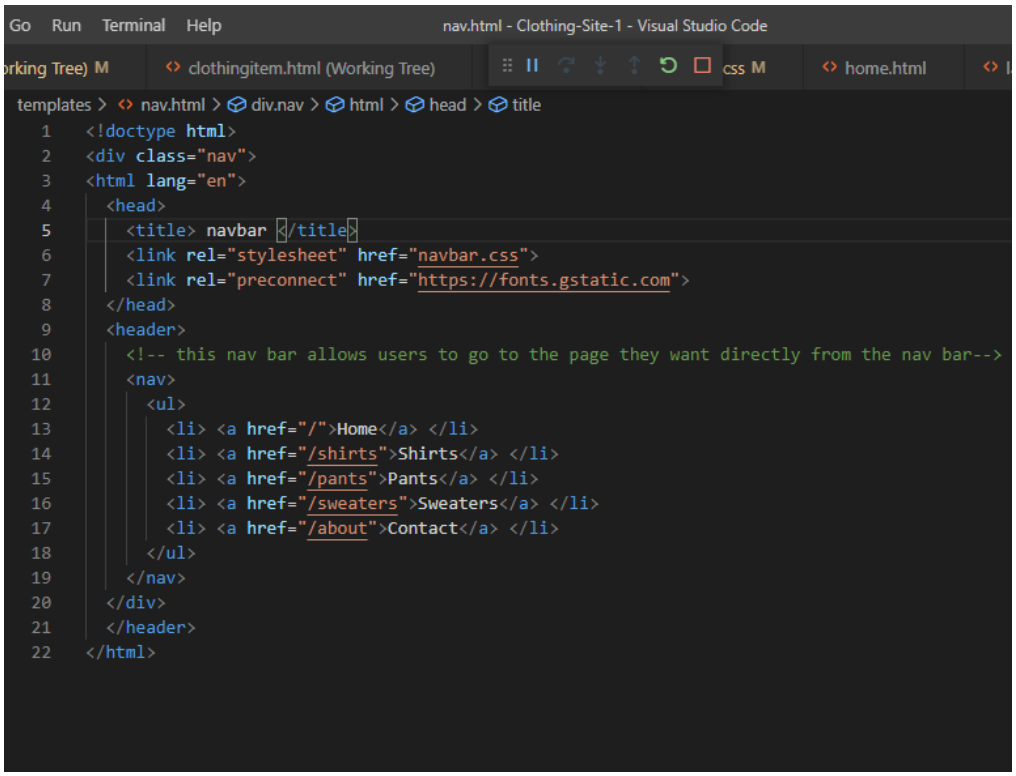


7) **The content of the general items (This is for shirts only but the template is used on the other page as well)**



Clothing Site (Chaos Clothing) - Planning and Design Doc

8) The navbar



Clothing Site (Chaos Clothing) - Planning and Design Doc

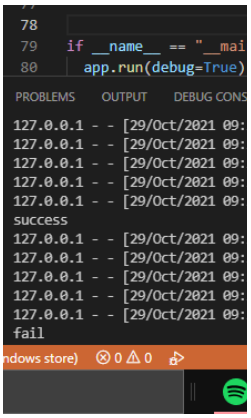
Testing table:

Query that I'm testing	How am i testing it	Expected results	Results	Notes
The "do query" that would cause less hardcoding and simplify my queries (code snippet 1)	By making it load all the pages where do queries are used: the single clothing item page and the all items page"	That when I click onto the each respective pages the page would only show the info for 1 item of clothing and then all the items of a certain category respectively	At first the query would only work for a fetch all where in the case where i wanted 1 item of clothing i wanted a fetch one, and since the fetch one needed some data (a int) this was a fail. In order to fix this problem i added the 2 conditions of data=none and fetchone=false and when i wanted to do a fetchone i would make it true	This would allow me to shorten all my queries and not have to rewrite all of my do-queries again.
That the website displays all the info for one item of clothing (Code snippet 3 and 6)	By manually adding the id of the item to the url	For the list of information (id, name, photo, description) to show up.	All of the items showed up	All of the things showed up but due to styling everything was on the right, will add CSS to make

Clothing Site (Chaos Clothing) - Planning and Design Doc

				everything centered.
Grabbing all the colors for the respective clothing item and displaying them (code snippet 6)	By manually adding the id of the item to the url	For the page to add the colors (white, black, blue, etc…) and for them to be displayed	This worked for items that only had 1 color (like only white, black)	The display worked but needed to be altered so that items with multiple colors would display all of the multiple colors.
Grabbing all the colors for the respective clothing item and displaying them (the difference to the one above is that I made fetchone = false) (code snippet 6)	By manually adding the id of the item to the url	For the page to add the colors (white, black, blue, etc…) and for them to be displayed	This worked and added items that were linked with multiple colors.	Made the title change from “Color” to “Color(s)” in order to express
Making sure all navbar and all its links work (code snippet 8)	By clicking the different links to make sure they work	When clicked each word should take the user to the specific page mentioned where different clothes are displayed.	This is a pass as these links work. At first I ran into the problem where if you pressed a nav bar button twice it would continue on the url instead of replacing it. This problem was solved by changing the	This is still a bit buggy as due if there is a url with 2 “/” the link only replaces the last “/”. To fix this I will try to get rid of the first “/”

Clothing Site (Chaos Clothing) - Planning and Design Doc

			link of it from eg “shirts” to “/shirts”	
The if statements that control if the user is directed to the error page or if they are directed to the clothing item page (if statement in snippet 3)	By running this there is a print statement in the debug console.	If it's below the number (int>15) it prints fail, if not it prints success		After this the code redirects the user to a specific template
The do queries for the site to display all the items for one specific category of clothing. (code snippet 4 and 7)	By running /shirts	The page displays all the data for anything under the category of shirts	It displays the data (id, name, price and photo) for all shirts. Pass	This only displays the data. CSS flex and containers will be used to make the data more constructed helping the aesthetic of the page

Relevant conventions to improve the quality of the outcome:

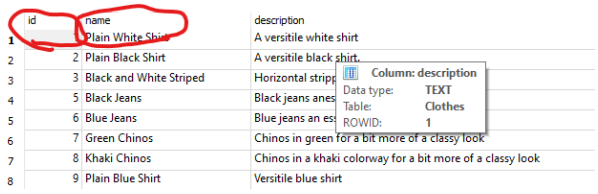
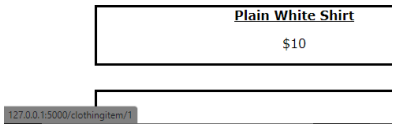
At first I had displayed my data for my shirts page by using a table. This method was outdated due to there being better options for displaying data. I used flexboxes to display the data. At first I made the content of the site (the photo, price and name) into a container. I added a border to separate the content. By doing this I made sure that all content had a separate container due to there being a for loop. After that I made the display of all the containers into flex boxes. This made it go into a line. After that I made it wrap so that the content would stop when the page finished causing less spacing. I added spacing so that they

looked separate. This caused improvement as at first it had been less professional and with the tampering made to my CSS it caused the site to look exactly how I envisioned it in my plan.

Another way I improved the aesthetics of my page is by making the content go in different lines and displaying it in the center. At first I had all of my content in the “single item” page and it was all displayed to the right. I made all the text centered by and the images centered, this development made the page look exactly like what I had in my plan.

Data integrity:

All of the pictures used are under a creative commons license which means that I am allowed to use this for this project. In the table below we see that the data is coming directly from the database straight to the website. By doing these tests we ensure that all the data is being processed and displayed correctly.

Information	Website	Pass/fail
		The id links directly to the clothing item page that I want it to. This means that this is a successful test. The descriptions shown are of my own writing further proving the originality of my data,

Clothing Site (Chaos Clothing) - Planning and Design Doc

typeid 2 2 g 2 1 1 1 1 2 g 2 2 2 3 3 3 id name describtion 1 1 Pants Essential trousers to add in any wardrobe 2 2 Shirt A tshirt comes in many diffferent sizes, shapes and colors. A essential in anyones wardrobe. 3 3 Sweater A long sleeved sweater, mainly used to keep you warm in winter and autumn.		Since i used the name to link all the pages together, in this screenshot you can see that the shirts page matches the name. This displays all the info for the specific typeid This method is repeated in my pages for the pants and sweaters.

Clothing Site (Chaos Clothing) - Planning and Design Doc

Comments on decisions made to improve the quality of the outcome

When getting images for my website I tried to get images with the same white background. This is so that the overall layout of the website looks good. This caused me to pick as many images with a white background as possible. Overall this improved the quality of my website by making the layout of it look consistent whereas if the images had different colors this would cause them to look weird and totally different.