Name:
Pixels

Team Number:
011 - 7

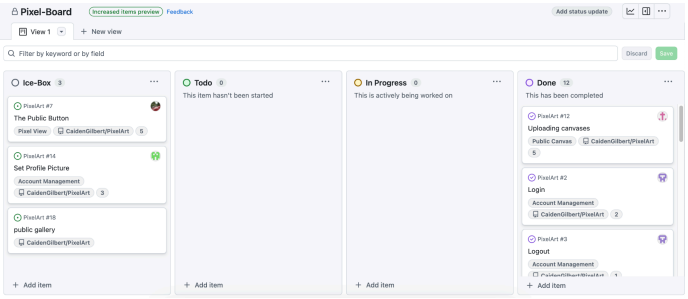
Team Members:
Caiden Gilbert, Davion Miller, Conrad Sadler, John Tran, Jingda Yu

Description:

We created a pixel art software where you can create a plain canvas that automatically saves when the artist exists(only on desktop). This app has a private gallery, a collaborative canvas, and a blank canvas. These functionalities will allow users to privately and collaboratively color pixels. Other additional features include a registration page and a login page.

In order for artists to enter a collaborative canvas, one artist will have to make a canvas by entering a code they choose in the navigation bar. Then they will press join. After one artist has made a collaborative canvas, they can share the exclusive canvas code with other artists. Other artists can easily join by entering that same code in the navbar and clicking join. Users can easily navigate the application via the navigation bar on the top. If a user clicks the logo, they will be directed to the home page, if a user clicks their username, a drop-down menu appears with the options of private gallery and logout. If the user decides to go to their private gallery, they will be able to see all properly saved artworks (saved by clicking the save button in any given canvas) and auto-saved artworks (thumbnail is not saved, but artwork is saved).

Project Tracker:
https://github.com/users/CaidenGilbert/projects/1



Video:
https://github.com/CaidenGilbert/PixelArt/blob/main/MilestoneSubmissions/demo.mp4

Github Repository:
https://github.com/CaidenGilbert/PixelArt

Contributions:

Conrad's Contributions:
For this project, I was mostly responsible for implementing the websocket functionality. The features that depend on websockets were the real time collaborative canvases, saving a collaborative canvas, and ensuring that users who enter a collaborative canvas even after the canvas has been colored will see the colored canvas. Besides Websockets, I helped with integration of other team members' parts, and I also fixed multiple bugs towards the end of the project including a couple bugs related to saving the thumbnails.

Caiden's Contributions:
For this project, most of my work was on the front end. I made the wireframes, did the entirety of the styling, and all of the navigation for our website. This took up the majority of my time however I also created the canvas functionality not including the color. I helped out a lot with integration and was able to squash a few bugs like the palette feature not showing up consistently and the canvas being twice as tall as it wide. I also created the presentation and worked on the project report.

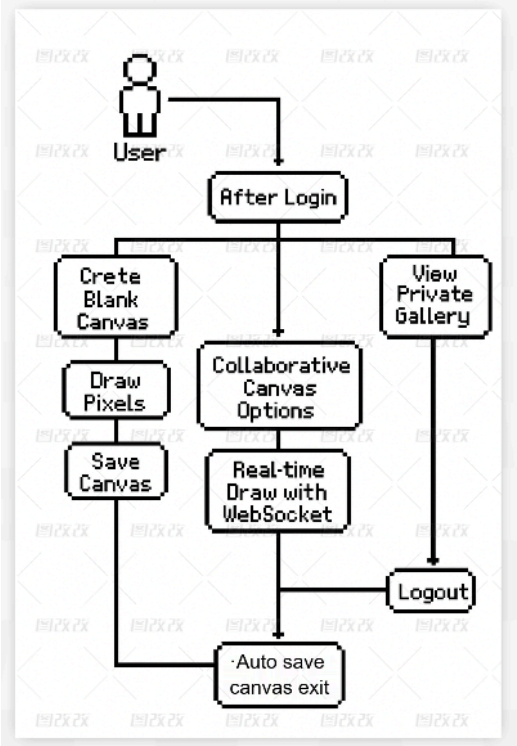
John's Contributions:
I worked on the color picker, palette, and loading canvases from the database. I wrote the entirety of the color picker code and its functionality, including the look of the HSV sliders and the preview. I did the entirety of the palette, which saves current colors (and prevents duplicate colors from being added). I integrated these two things with the actual canvas. The last thing I worked on was taking the canvas data from the database and loading it onto a blank canvas. I created thumbnails for each image so that the user knows what the canvas looks like beforehand.

Davion's Contributions:
Most of my work was with API requests to get the databases to save and store information. I worked on the upload process to store canvases and worked on the global gallery. Integrated the saving process with the upload process and made it so you could update your canvas if you submitted the same one twice. For the global gallery I also stored the date and time you uploaded it to main as well as which user uploaded the canvas; which will display on the global gallery. Besides that I worked on the initial wireframes and on render to get the servers running for testing.

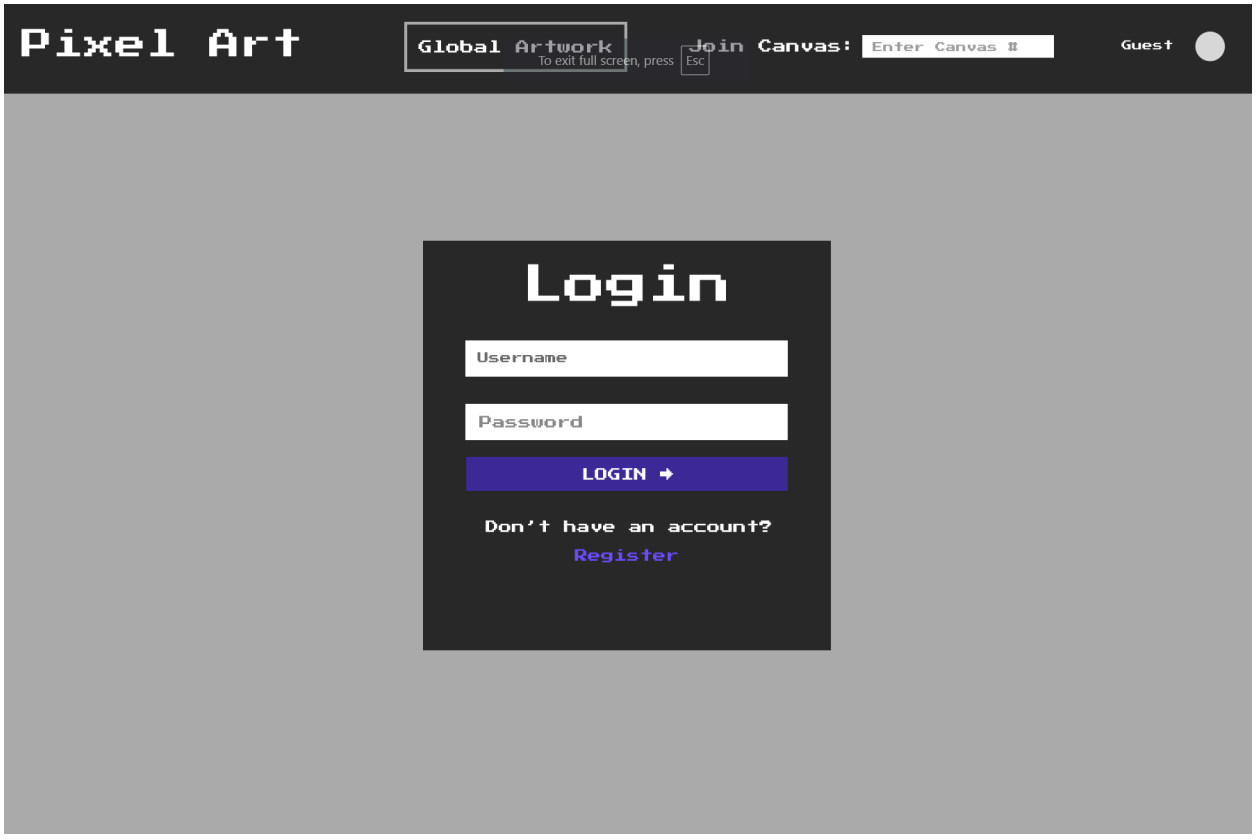
Jingda's Contributions:

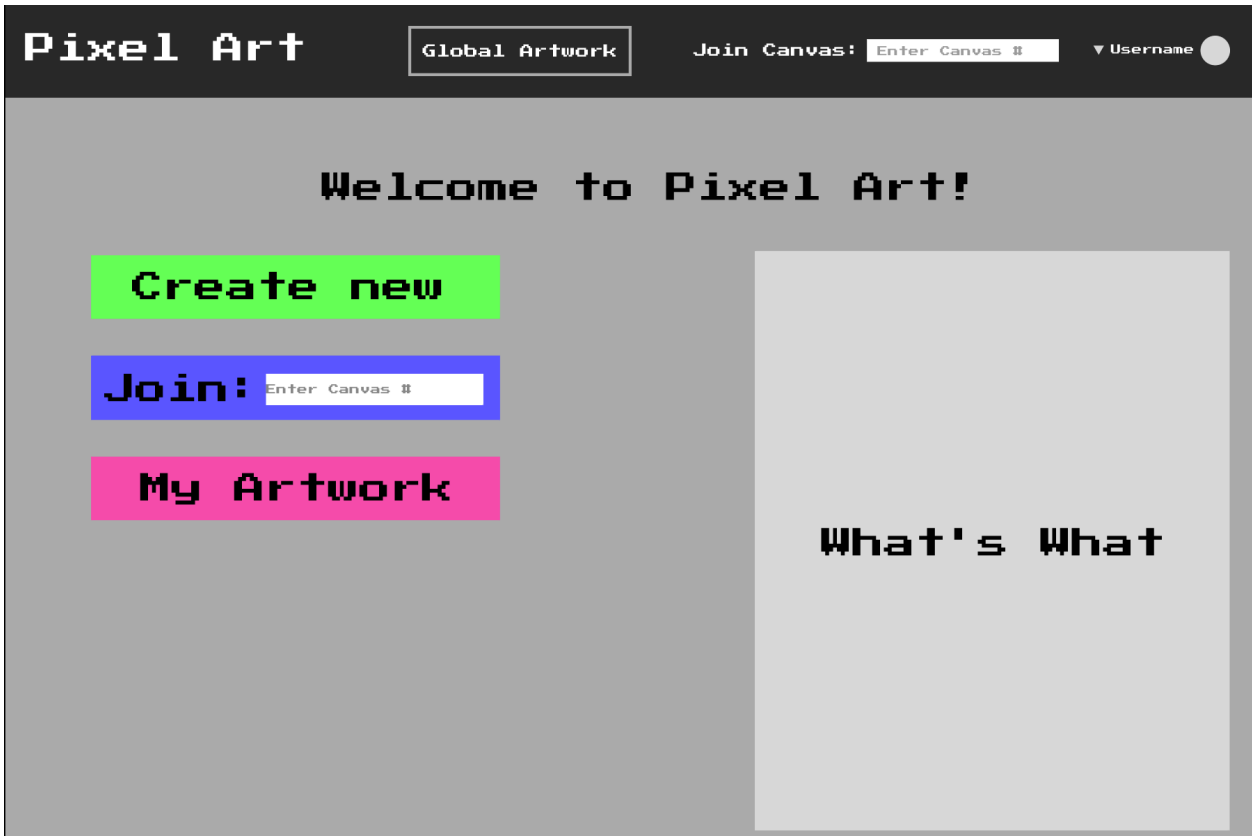
I focused on frontend visual optimization and layout refinement, specifically the CSS architecture for front pages including the login/registration interfaces and gallery grid system. I implemented responsive hover states for navigation components. Due to non communicating in time in frontend scope planning, my styling work inadvertently duplicated Caiden's initial implementation.

Use Case Diagram:

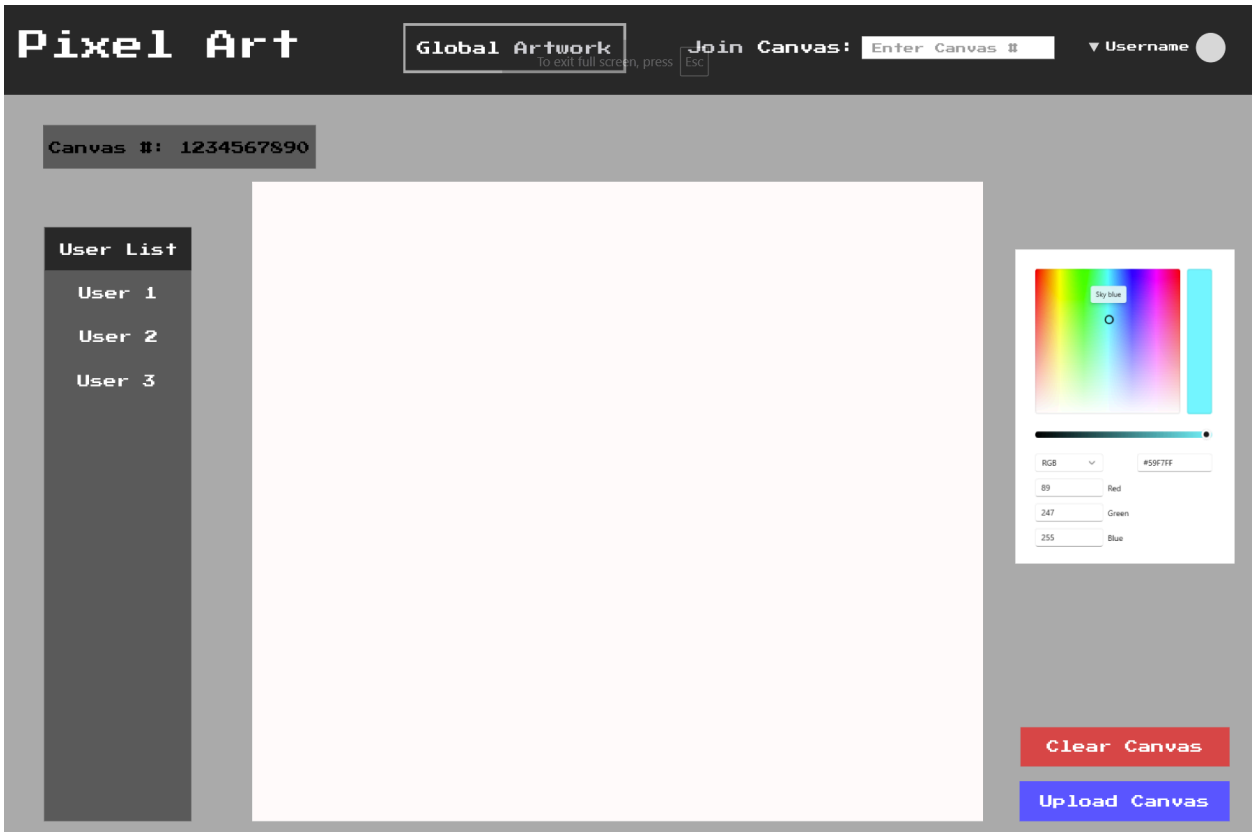


Wireframes:
https://www.figma.com/proto/cPCJG6GFZoZ4g3ELcEa9EM/PixelArt?node-id=3-166&starting-point-node-id=1%3A3&t=eQ1UsQBnAW3Z9eiw-1


Pixel Art
Global Artwork
To exit full screen, press Esc
Join Canvas:
Enter Canvas #
Guest
Login
Username
Password
LOGIN
Don't have an account?
Register


Pixel Art
Global Artwork
Join Canvas:
Enter Canvas #
Username
Welcome to Pixel Art!
Create new
Join:
Enter Canvas #
My Artwork
What's What


Pixel Art
Global Artwork
To exit full screen, press Esc
Join Canvas:
Enter Canvas #
Username
My Art


Pixel Art
Global Artwork
To exit full screen, press Esc
Join Canvas:
Enter Canvas #
Username
Canvas #: 1234567890
User List
User 1
User 2
User 3
Sky blue
RGB
#59F7FF
89
Red
247
Green
255
Blue
Clear Canvas
Upload Canvas

Test results:

Conrad had his friend Gabe be the user acceptance tester. His knowledge of programming is limited to MATLAB and C++. To my knowledge he has never built and deployed an application online. Given that his major is in computer engineering, he might not have to take software development.

UAT For Exclusive Canvas:

Conrad was able to set Gabe up in order for him to conduct this test. While conducting this test, Gabe was able to navigate around the webpage with little explanation needed. It seems like the learning curve is pretty low for the application. Gabe completed the test. However, after Gabe finished the test for the exclusive canvas he was able to find a bug that was unrelated to the user test. This bug is related to how the thumbnail is saved for each artwork. If you conducted a certain sequence of events, then you could save a new thumbnail to an old canvas. Conrad was able to fix this after Gabe concluded his testing.

UAT For New Canvas Button:

Conrad was able to set Gabe up in order for him to conduct this test. During this test, he required additional explanation which indicated that using the palette to save certain colors has a steeper learning curve than any other aspect of the blank canvas. However, after some explanation he was able to conduct the test without trouble. After the test Gabe did want to have an eraser that would reset one pixel instead of having to clear the canvas everytime.

UAT For artist Session:

This test was rather straightforward. A user acceptance tester that was using a brute force way of checking all possible pages might have found a vulnerability and potentially crashed the web page. Gabe clicked every possible button on the home page and even used the back arrow to try and edit a canvas while he was logged out. The application did not crash despite Gabe’s efforts. The most notable thing from this test is that Gabe was able to send malicious queries to the database which were rejected. This resulted in the program entering a catch statement where it printed an error to the console but the application did not crash.

UAT for auto upload:

Gabe had the most fun testing the auto upload on the cloud. The auto upload worked as expected. However, Gabe was expecting the canvas to automatically save without adding a name, but Conrad had to explain to him why a name was needed. During this test Gabe also mentioned the poor layout of the private gallery. After Gabe had finished his testing, Conrad implemented a bootstrap grid to better organize the saved artwork in the private gallery. The test ended with Conrad taking a video of Gabe having fun clicking pixels on his phone and watching them update on his computer and Conrad's computer.

Deployment:
https://tinyurl.com/2p9mxh98