Share Pixels

The Team

Team **NotPixar**:
William Cao: Project Manager & Flask/Database
Ethan Chen: Frontend and Graphic Design
Joseph Yusufov: Frontend & Flask/Database

Objective

To create a platform for people to share their artwork similar to Imgur, Tumblr, and Reddit. Users will create pixel arts, which will automatically be shared with others. Most websites are based on creating artwork on an external application then uploaded online. This website will combine those two steps to increase availability and ease to use.

Outline

- **Flask**: Backend framework. We have been working with the framework since the beginning of the course and most familiar with this.
- **Mongo**: Database: The flexibility allows us to quickly prototype ideas and create them.
- **Bootstrap**: Frontend CSS framework. We are most familiar with this, and it has all the features we need to quickly create a website.
- **Vanilla Javascript**: We are not planning to use any frontend frameworks (React, Vue, etc.) for simplicity.

Features

Minimal product

- Choose from 16 preset colors and fill in a grid 25x25.
- Upload the work to save as a ppm file and a png for sharing, modifying, download in the future.
- Anyone can visit the website to see a gallery of the pixel art work

Final product

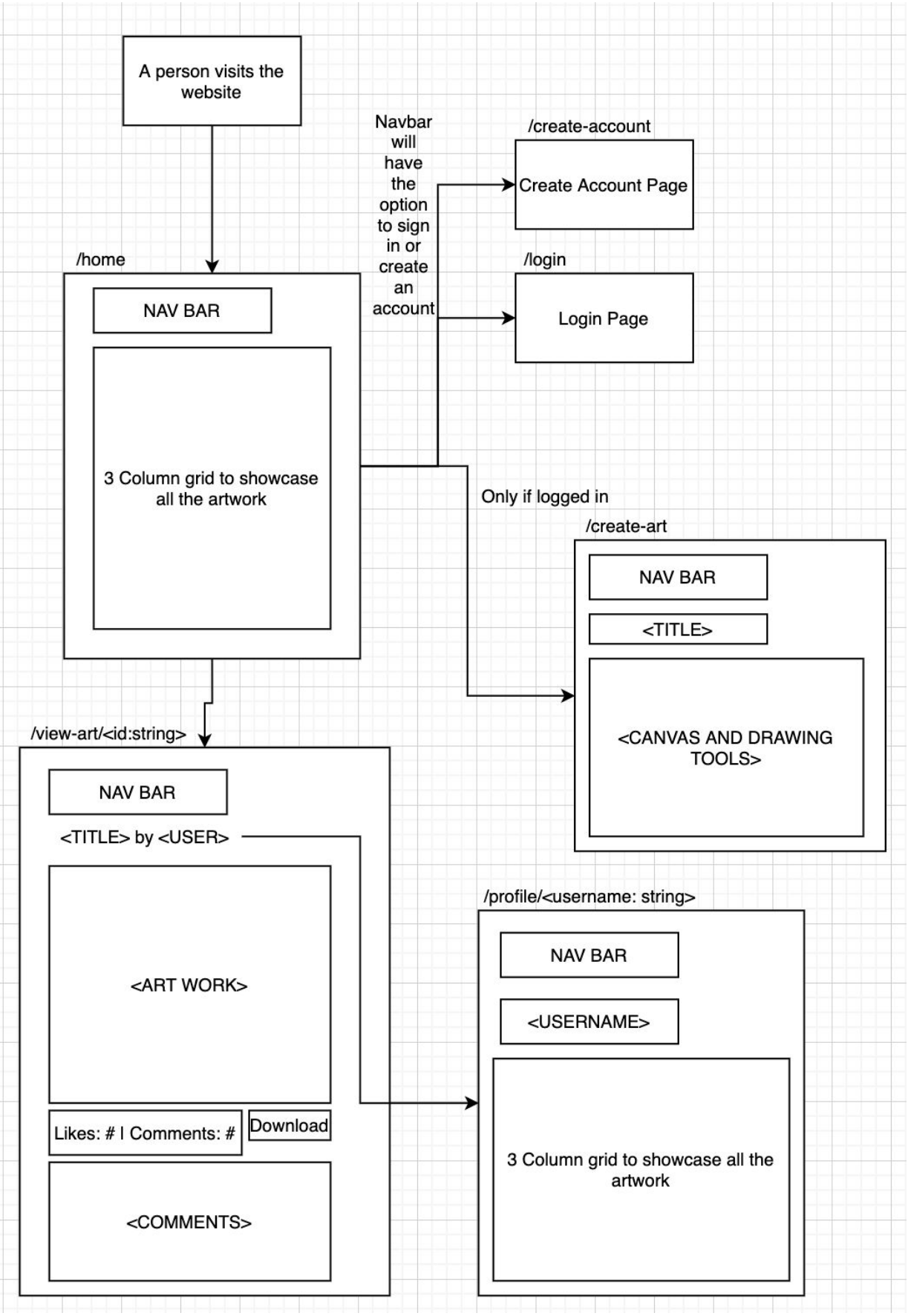
- Can comment on each others' work
- Can like images
- Showcase all the work created in the last week
- Can view a user's profile and see all the artwork created
- Can follow other people
- Have two feeds: A public and private
    - Public: Anyone can see this, for top works for past week
    - Private: List of all the most recent works by users the current user is following

Nice to have

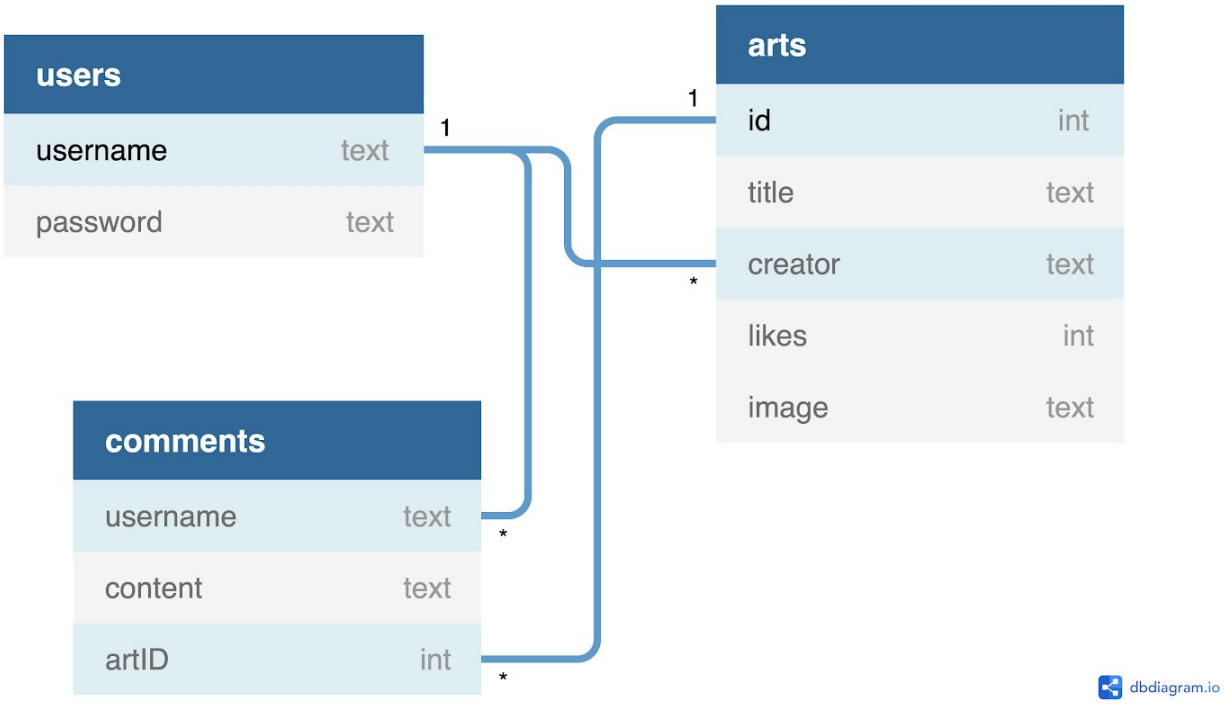
- Have a color picker to allow for greater choice of colors

- Have multiple sized pixel arts
- Built in button to download either .ppm file or .png file
- Have a short description on each artwork

Sitemap


A person visits the website
Navbar will have the option to sign in or create an account
/create-account
Create Account Page
/login
Login Page
/home
NAV BAR
3 Column grid to showcase all the artwork
Only if logged in
/create-art
NAV BAR
<TITLE>
<CANVAS AND DRAWING TOOLS>
/view-art/<id:string>
NAV BAR
<TITLE> by <USER>
<ART WORK>
Likes: # | Comments: #
Download
<COMMENTS>
/profile/<username: string>
NAV BAR
<USERNAME>
3 Column grid to showcase all the artwork

Database Design



Web Routes (All have GET requests for html)

- /home home.html
    - Template Variables:

```
{
	// 2D array of image links
	images: [
	{
		image: string,
		likes: string,
		creator: string,
		title: string,
		art_id: string,
		num_comments: number,
		hasLiked: boolean
	}
	]
}
```

    - Who can access:
        - Anyone

- /view-art/<string:id> view-art.html
    - Template Variables:

```
{
      title: string,
      creator: string,
      image: ppm string
      comments: [
            {
                  username: string,
                  content: string
            }
      ],
      likes: number,
      hasLiked: boolean
}
```

- /profile/<string:username> user-profile.html
    - Template Variables:

```
{
      // 2D array of image links
      images: [
            [], // first column
            [], // second column
            []  // third column
      ],
      username: string
}
```

    -
- /create-art create-art.html
    - Who can access:
        - Only those logged in
    - Redirects:
        - When clicked create, redirects to “/art/<id:int>”
- /login login.html
    - Flashes:
        - Successfully logged in
        - Wrong username/password
    - POST:
        - FormData with “username” and “password”
    - Who can access:
        - Anyone not logged in
    - Redirects:
        - Goes to previous route if already logged in
- /create-account create-account.html
    - Flashes:
        - Successfully logged in

        - Username taken/repeated password don’t match
    - POST:
        - FormData with “username”, “password”, “passwordRepeat”
    - Who can access:
        - Anyone not logged in
    - Redirects:
        - Goes to previous route if already logged in

API Routes

- POST: /api/image/create
    - Send to server:

```
{
      title: string,
      image: string
}
```

    - Received from server:

```
{
      // id of the art created
      id: string
}
```

- POST: /api/comment/create

```
{
      art_id: string,
      content: string
}
```

- POST: /api/<string:id>/like

NOT DATA NEEDS TO BE SENT
Toggles like and unlike

Detailed Outline

Session Data:

```
{
      // If the user is logged in, this is the username. If not logged in, this field
does not exist
      "username": string
}
```

PPM file:

http://netpbm.sourceforge.net/doc/ppm.html
https://en.wikipedia.org/wiki/Netpbm#PPM_example

- Do not include “#”

- Ignore 70 character limit per line

Timeline:

- To be done 6/1
    - Joseph:
        - Canvas with a white background, and cells. (25x25 pixel canvas).
        - When you click on a square, it will turn black
        - (0, 0) → top left
        - Row: y, Column: x
        - (3, 2) -> 3 right, 2 down (x, y)
        - Convert the 2d array representing the canvas to a .ppm file.
        - Function that takes in a id, and that id is the canvas. Add the event listeners and other stuff needed. Return statement -> object
    - Ethan:
        - Each pixel art will be 250x250.
        - 3 columns going down
        - Each column will have an image.
        - If you are given 10 images: the first column will have 4 images, second have will have 3, third will be 3.
        - Populate by rows
    - William:
        - Boilerplate code
        - Login code working
        - Work on taking in ppm file and converting to images
- To be done 6/3
    - Finish minimal product (look at features section)
- To be done 6/8
    - Finish Final product
- To be done before 6/11
    - Testing websites for any last bugs
    - Would like to have “Built in button to download either .ppm file or .png file” feature working. Any other extra features are if we are very ahead of time.
    - Due date