● The *product backlog*: a complete list of all functionality (i.e., the actions) of your project, and an English description of each action. We strongly suggest that you organize these features into groups/modules based on related functionality.

Registration and Authentication:
- User can register for accounts and login on the website
- User can upload an image and a bio to personalize his own profile
- User can follow and unfollow other users on profile pages

Image Generation:
- User can enter a prompt and get images, like Dall-E through calling API or using tools like puppeteer to parse the response webpages
    - Using Dall-E's API might not be possible right now.
    - Using Stable Diffusion API

https://replicate.com/stability-ai/stable-diffusion

Image Creation and Updating:
- User can view images that they have created
- Recently generated images will be saved automatically
- Images can be added to a favorites list
- Images can be labeled with user-defined tags

(Django models will be used for the above)

Communication:
(These features are not a priority, can be cut down)
- Discussion board where multiple users can share images and comment on images (similar to social media platforms)
- User can chat with any other content producer (community member)

● The first *sprint backlog*: a complete list of the functionality you will complete during your first sprint, and how that work is allocated among your team members.

- Find a working API for generating images (even if not ideal)* [xinyew]
- Implement basic prompt entry using that API, so we know it works* [todo]. No need for a complex interface, just displaying the 4 pictures is enough
- Use react.js for frontend development [xinyew]
- Git- quick rundown, conventions. Code conventions. [ejwu]
- Write starting functions/code structure for all views, models [ejwu]

Starred items are prioritized

● The name and Andrew ID of the *product owner* for the first sprint.

- ejwu

● A complete implementation of the data models used by your application. This may be written in as Django models, SQL, or some equivalent style implementation if you use another framework.

Rough draft:

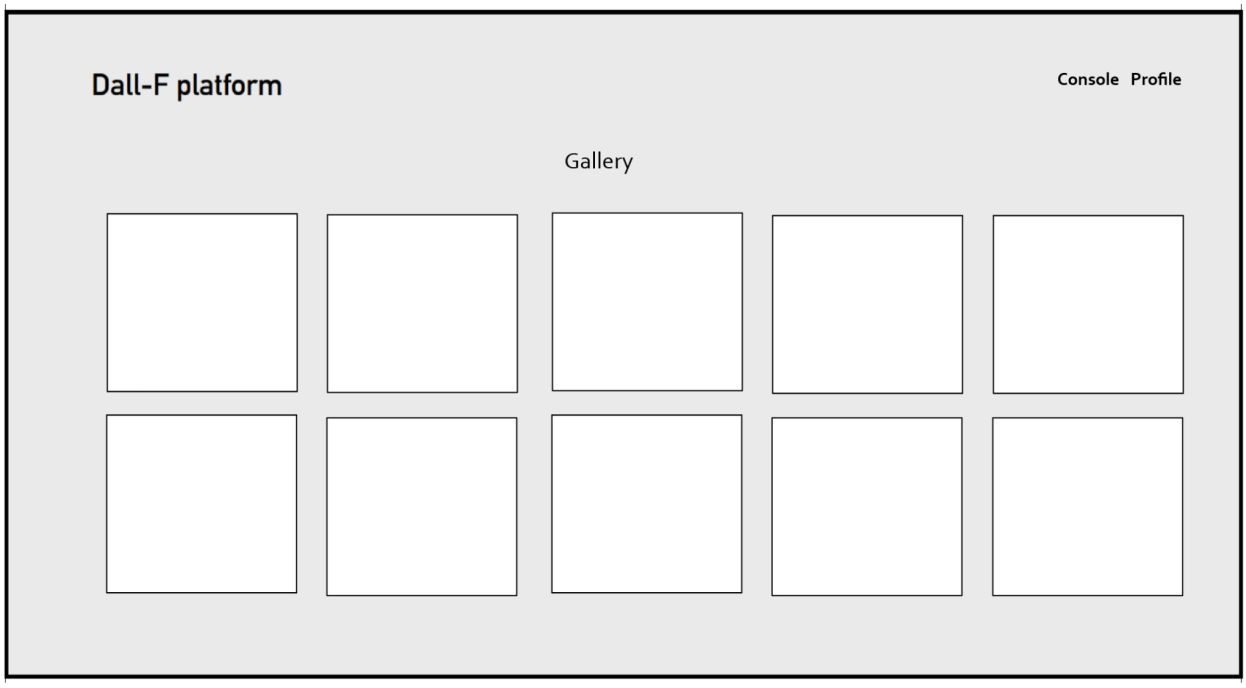
- User model: use django settings to set a custom user model. Contains username, password, email; profile fields (bio, profile picture)
- Image model: favorited, tags, date created. Created_by (many-to-one to user). prompt.
- Image group (represents a group of images generated by the same prompt). Images (one-to-many to image)
    - This is necessary because we need to display the most recent groups of images.

For discussion/chat, more models will be needed.

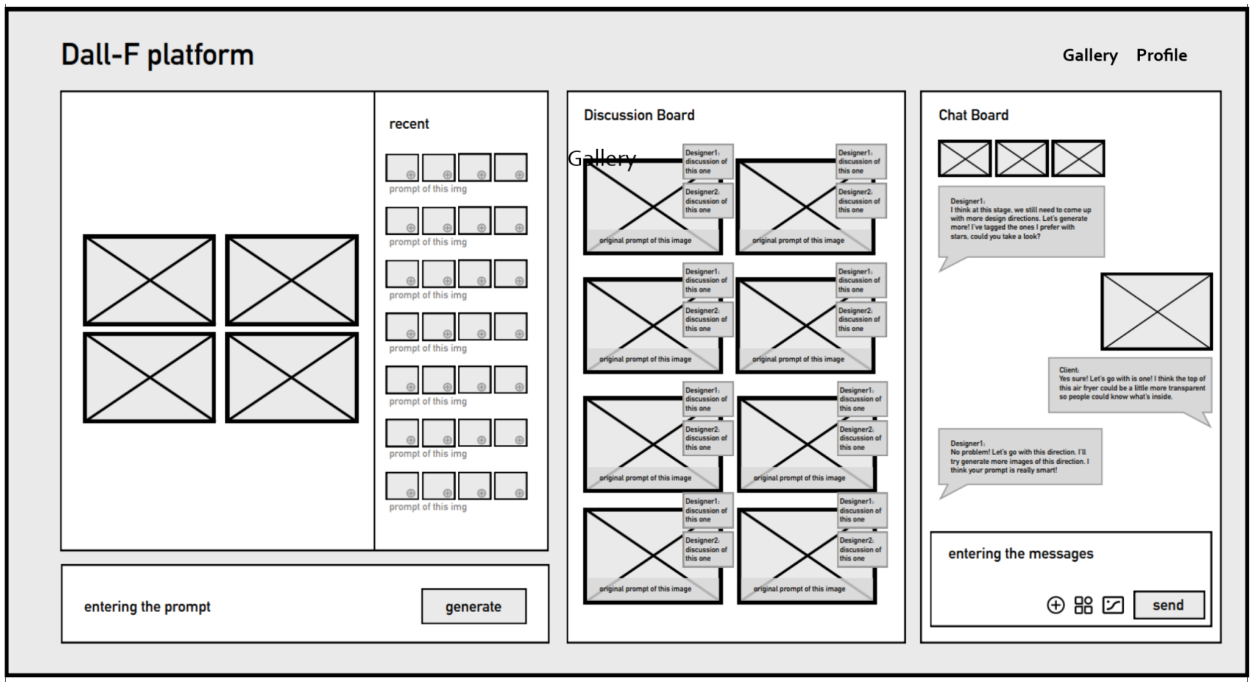
(Add Django code later)

● A complete set of drawn wireframes or HTML mockups for your application, for all non-trivial views within the application.

Gallery (Home Page):



User Console:

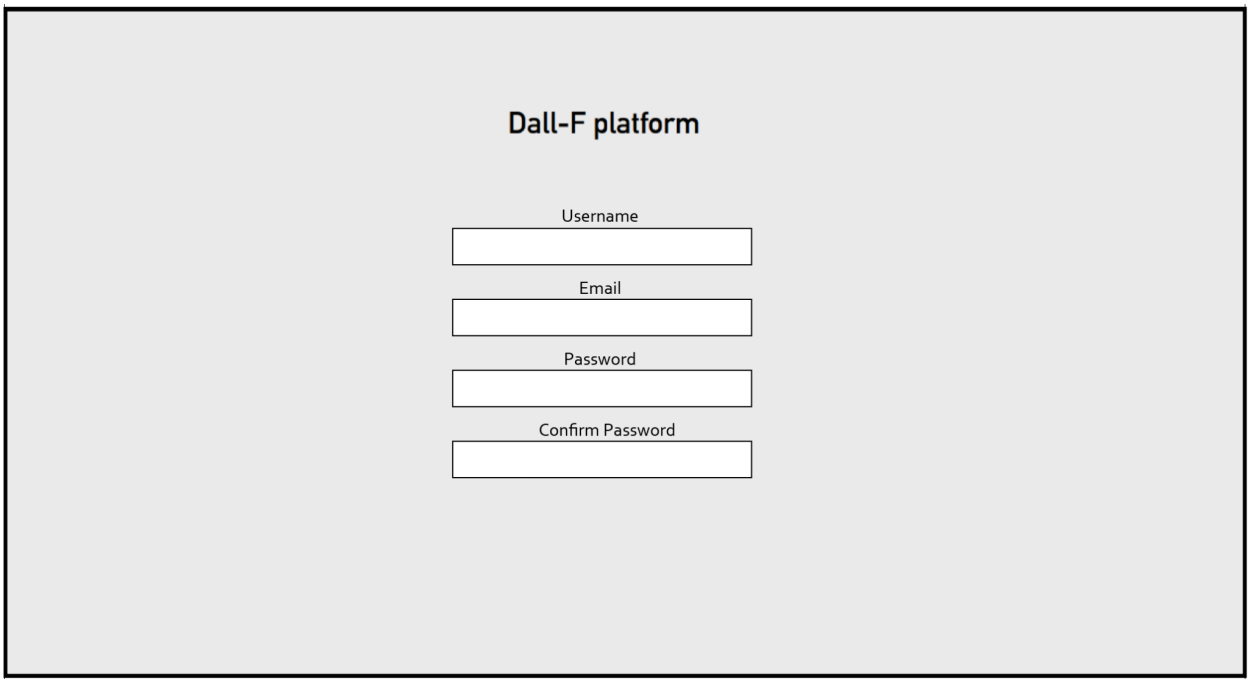


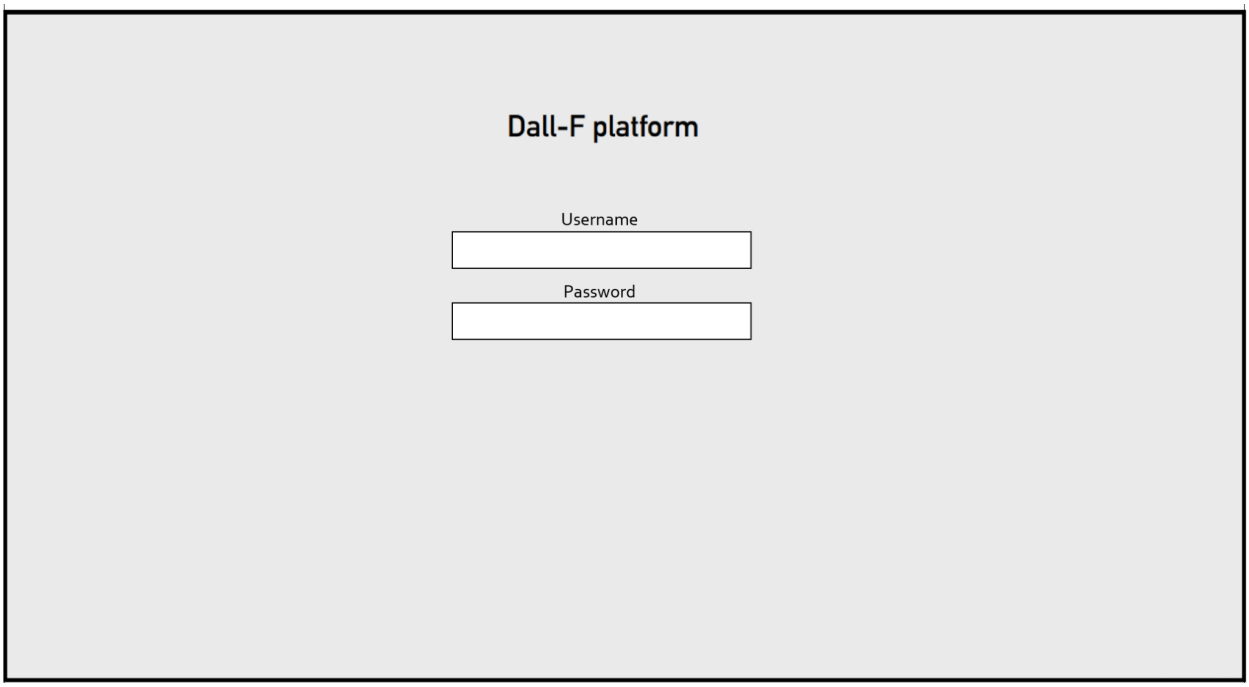
For now we have a one-page application, all sections are displayed at once.

Additional goals:

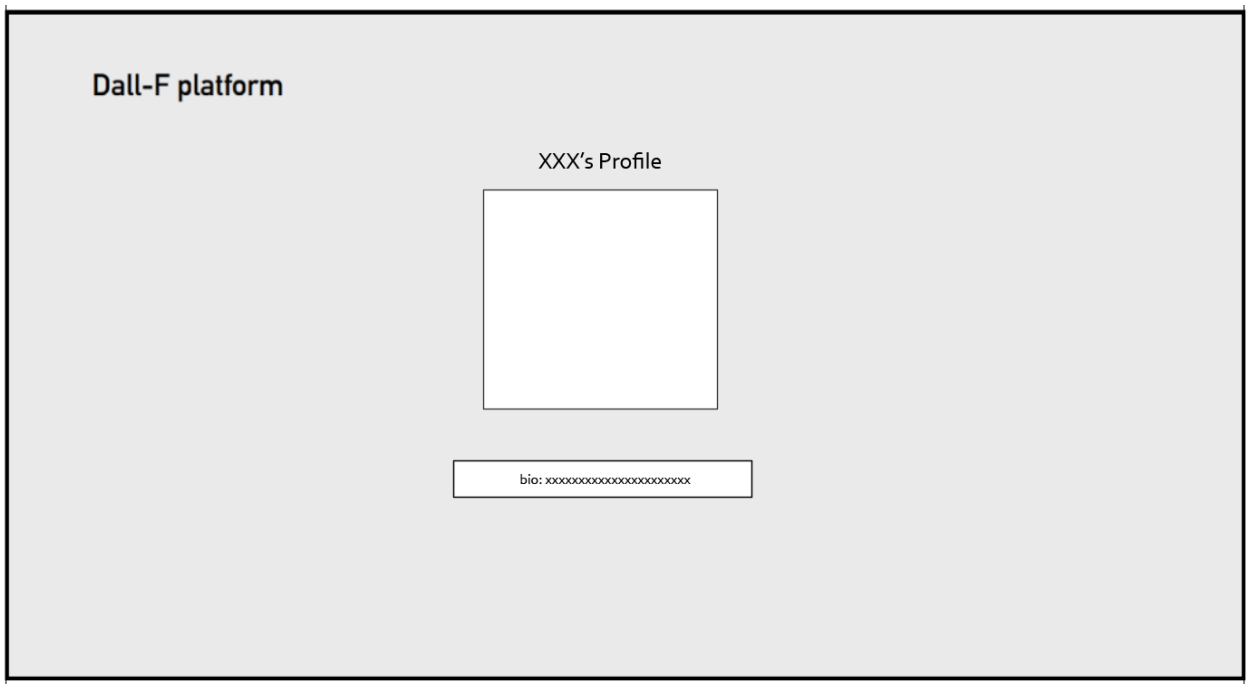
- Click to expand one section
- Movable dividers
- Animations for transitions

Register/Login:





Profile:



Git conventions

We will use a merge-based workflow. (no rebasing)

Personal branches for each. (not feature-based) Use your name in the name of the branch.

Style

Install autopep8 and use the provided config