Teamwork Plan

George Ezenna, Stancellous Matoreva, Kairat Ashim, Eric Manzi, Favyen Bastani

Stakeholders

Use of this application is limited to the MIT community. More specifically, our stakeholders are makers. Makers make projects and post them to the application. These could range from undergrads and grad students to professors and can be designers, developers, etc. They post their project looking for feedback, help or partners on their project. They also comment and give feedback on projects by other makers in the community.

Tasks

Overall task list:

- Particular front-end tasks
    - Overall website theme (6hr; George)
    - Activity Feed view (4hr, Eric)
    - Project dashboard view (6hr; George)
        - Assets layout (images, videos, URLs)
        - Post/comment layout
    - Trending Projects / search projects view (2hr, Favyen)
- Project (Kairat)
    - Model
        - Add project with name/description/tags/assets (1.5hr)
            - Asset includes type enum: photo, video, website URL
        - Update name, update description (1hr)
        - Add tags (3hr)
        - De-activate project if no longer being worked on (1hr)
        - Search by word, match with name/description (2hr)
    - Controller (2hr)
        - Get model instance and pass template parameters
        - Handle add/update/delete
- User (Eric)
    - Model
        - Register, authenticate (1hr)
        - Verify e-mail address (1hr) + use Nodemailer to send activation url
        - Reset password (2hr)
            - Reset password request (enter username/e-mail address)

            - Reset password confirm (pass username, confirmation code, and new password)
        - Follow/unfollow other users (0.5hr)
        - Favorite/unfavorite projects (0.5hr)
        - Upvote/recall project vote (0.5hr)
        - Upvote/recall post vote (0.5hr)
    - View (2hr)
        - Show profile with username and basic description for projects that user is managing
- Search (3hr; Favyen)
    - Controller: pass search phrase to the project model search function
    - View
        - Search form
        - Display search results
- Post (Favyen)
    - Discussion Model (2.5hr)
        - Add post
        - Add comment by user to discussion
        - Get discussions for a project
    - Comment Model
        - has text
- Security (Favyen)
    - Rate-limit or captcha for sign-ups (1hr)
    - Rate-limit login attempts (0.5hr)
    - CSRF tokens (1hr)
    - Password hashing (0.5hr)
    - E-mail address verification (1hr)

A rough schedule for task progress, and task assignment:

- By 15 November:
    - Basic application framework
    - Define model interfaces
- By 18 November:
    - Initial user model, with support for registration and authentication
    - Initial project model, with support for name, description, posts
    - Post model: support adding/editing posts, adding/editing comments, get comments for a post
    - Initial/bare project dashboard controller and view
    - Initial/bare project page controller and view (incl. adding posts, adding comments)
    - Initial/bare navigation bar, login, sign-up controller and views
    - Initial/bare activity feed controller and view

- By 22 November:
    - User model: upvoting, favorite a project, follow a post
    - Project model: storing various assets
    - Project controller/view: asynchronously fetch more posts/comments, allow updating name/description, updating assets
    - Trending projects: show projects with recent activity and recent upvotes
- By MVP due date:
    - Add initial theme to views
    - E-mail verification system
    - Make sure all code has good unit tests
- By 29 November:
    - Security: add CSRF tokens to forms, rate-limit sign-ups/logins
    - Project model: support searching by string, should check for projects matching in name or description
    - Tags: add tag to project, sort by tag on trending projects page
    - Search controller/view: display search results, with autocomplete
    - Featured projects, terms of service
- By 3 December
    - Security: rate-limit anything else that needs to be rate limited, make sure asynchronous requests don't cause XSS
    - Theme: polish website theme and make views consistent
    - Password reset system
    - Go through unit tests, make sure they all look good

Risks

1. Since this is a very extensible project, there are a lot additional features that ambitious people in the team might want to include which could mean that a lot of time is spent on features that are not central to the design. We will avoid this by clearly listing out all of the features that we plan to add before we begin implementation and only adding additional features if the entire team agrees that the proposed features is absolutely necessary or is totally awesome.
2. Users might steal ideas from projects they find on the app and implement it themselves. To reduce the chance of this happening, we will include a terms of service page that explains that the ideas posted on the app are intellectual property of the posters and if user must first seek permission from the original posters to work on an idea.
3. If the platform doesn’t attract enough users, it won’t be very useful. One possible mitigation would be to promote the app among students and advertise it to professors and grad students. To get the ball rolling, we could add suitable projects from Anne Hunter’s mailing list or from the UROP listings (with the poster’s permission)
4. We plan on having the default sorting of the projects be by votes so that the most upvoted projects are shown first, and add an option of sorting by date. However, this means that old projects and projects with no or only a few votes won’t get any exposure.

We plan to mitigate this by having a “featured hacks” page that shows random projects that were picked from the oldest/lowest-vote projects.

5. Most applications with public forums tend to be susceptible to trolls and pessimists. Some people might want to downvote ideas that aren’t necessarily bad ideas. We think this goes against the primary goal of this app which is to allow people to get constructive feedback on their ideas. To reduce non-constructive feedback, we have decided to remove the downvote feature for a product.

Minimum viable product

Our minimum viable product will deliver the core functionality of our app so users will be able to achieve their primary goal: to post and get feedback on projects.

Concepts included:

- Post -- A user makes a post that other users can see. A post can be feedback or a description for a new project feature
- Asset -- A project has assets (textual, images, video) which help convey its main idea
- Favorite -- If a user favorites a project, they can easily refer back to it and receive updates on the project

Concepts postponed:

- Tag -- the type/category of a project. Tags make it easier for the user to find projects that they might be interested in
- Upvote -- A count of how many people find a project interesting

Implementation included:

- Creating user accounts, login/logout
- Profile pages for users
- Posting new projects, with textual description and photos, viewing projects
- Upvoting projects
- Favoriting projects

Implementation postponed:

- Security
- Authenticating that they are MIT, verifying e-mail address
- Rich UI styling
- Searching for projects and filtering by project type won’t be implemented.
- Terms of service (to protect intellectual property rights)
- Featured projects - so old / low vote projects get more visibility
- Trending projects
- Comment on and upvoting posts
- Following users, activity feed