Recipea

I. Project Vision

Social media websites are some of the most popular in the world. They support pushing popular opinions and content to the top of peoples' feeds. We see this as a unique opportunity to convert recipe sharing into a social media tool that can help everyone discover the perfect recipe for them. Recipea allows for a wide range of audiences to effortlessly reach recipes which are relevant to their skill levels, flavor palettes, and income level. At the same time, the website will be an opportunity for content creators to increase their followers and show off their cooking skills and recipes.

Recipea functions by allowing cooks to use their profile or other mechanism to upload recipes which are posted directly to theirs and others' feeds. The recipes are ranked for user interest based on the tags applied to them. Additionally, user likes will influence the ranking of a recipe. Tags are important because they can be used for search terms such as vegetarian which implies a vegetarian meal or college which might imply a low budget or lack of cooking knowledge.

Some of the other interactive features of the website are following users whose recipes you like, liking recipes, reporting posts, creating a cookbook, and commenting on recipes. These features are available for registered users only. Creating quality interaction and a dedicated user base will increase the time an average user spends on the website. This is important for the sustainability of the website.

The website has the opportunity to spill over into other markets such as e-commerce for cookware and food products. There also exists the step of converting the website to a mobile application which seems to be the preferred access method for many social media platforms. We currently see starting as a website as the most reasonable since recipes require a larger amount of writing than a typical social media post might.

II. Monetization

There are a multitude of ways for making the website profitable. Many of them rely on revenue generated by advertisements. Due to the nature of the website, we see cookware companies, premium food companies, and grocery stores being the main advertisers. That being said, health conscientious products might also find a niche on our platform for those who use it to plan better meals. It would be a stretch goal, but we could link listed products to local grocery stores or have a feature that generates a grocery order for pickup which many large chains now offer.

Since content creators will be an important aspect of the website, we see it possible to include a subscription option where followers of a creator can view premium recipes posted by the creator they subscribe to. This is one of our stretch features to implement after we are able to complete the original features proposed in the project vision.

A final method of monetization is offering analytics. The users are not exposed to view any of these, rather, researchers might pay for certain information or advertisers might want customized information that they wouldn't otherwise be exposed to. It is worth considering the controversial nature of involving user data in monetization. We believe that it is necessary for us to succeed, but can't truly judge with need without the demand.

III. Pages (highlighted in bold)

Guest View:

Guests have the ability to scroll through **feeds**. These may or may not be ranked towards their preferences. In the case that we use cookies to track a guest user, we can create statefulness between consecutive visits to the website and serve tailored content that way. Otherwise, users will browse as if they were a first-time user. Guests can view **recipe pages** and other users' **profiles** but can't interact with content in any other way than viewing it. They will not have access to the **member pages** of content creators (stretch goal). Guest's will have access to the **login** and **register page** so they may become a registered user. The register and login page should allow a user to use another account such as from google or microsoft to sign in.

Registered User View:

A registered user (user) will have a feed that is tailored specifically to them. They have the ability to like, comment, and **post recipes**. The feed is anticipated to be the page users spend the most time on. Users can also subscribe to one another to receive new posts from that specific user. Also, a user can become a member of another user's private page through paid options (stretch feature). This gives access to private recipes by that user or the ability to interact with the creator. In the future, we plan on adding a way to signify that a particular user has received a certain amount of followers or members to set them apart from normal users. There is an opportunity here for icons to distinguish professional chefs or restaurant owners, etc.

Registered users will be able to view their **post history** in a separate page that breaks down the comments they've made and the recipes they've posted. Users will also have an **alerts page** that lets them know of people who responded to their post or message and the like.

Administrator View:

The administrator view can perform the same actions as a registered user but does not face any barriers for joining the private group of a user. An administrator can edit and remove posts, view **reported content**, view the same pages that an advertiser and user can, and edit advertiser and user details. The administrator should have a **searchable page** for users and advertisers.

Advertiser View:

An advertiser can decide how long they want to run their campaign, what content they run, and view analytics with regard to their campaign and how the user's interact with their advertisement. The advertiser is also considered a superset of the regular user, since an advertiser should also be able to make posts. The pages involved in the advertiser specifically, is one to **create a campaign**, one to **view all campaigns**, and one for **each campaign** that displays its effectiveness.

The advertiser and registered user might eventually be presented with a **payment page**, so we will also design a payment page. We are unsure whether we will incorporate actual payment methods such as paypal and debit or if we will include the template for payment information and simply make it so that fake money is transferred or the transaction is never sent in the case that we can implement payment methods without a paid web server.

IV. Tools

We plan to use ReactJS for our frontend. While the team is not familiar with React, we are all looking forward to learning the framework. React's benefit is in its ability to create common components with less code than using vanilla JavaScript. Of course, while using web development one cannot escape CSS and HTML so we expect to use these technologies on the frontend. Before actually starting either the frontend or backend, we need to decide what types of data we will be transferring, how each webpage will look and interact with each other, and use cases. For these steps design tools like Figma and StarUML will be invaluable.

The team is not entirely familiar with the proper tools for a backend. So we will eventually have to decide which backend will work the best for us. Our current thoughts are that we continue to use JavaScript on the backend since we are already familiar with working with JavaScript on the frontend. We will surely need APIs to access the data from our backend and serve dynamic web content based on a user's calculated feed. Thus, through the guidance of the professor, we have been instructed to learn about REST or GraphQL APIs or both. In this instance, preliminary research seems to suggest that GraphQL brings faster performance, but does not have the same structure and history that RESTful APIs do. REST APIs can serve content in multiple structures, and have wider support by third party tools, while GraphQL is only limited to strings and has had less time to be made compatible with other resources, thus at this point using the REST constraints seems like a reasonable option.

The team will host their code base on github under a private repository. The team will use git to manage version control. In order to assign individual team members to tasks, we are going to use the project feature of github which offers varying kinds of kanban boards for tracking development.

V. Limitations and Considerations

The team does not have access to a server machine so they will rely on their own machine to perform the job of both a server and client. We are unable to acquire a server since it requires money to run.

The team's development devices are a mix of Windows and MacOS. This means that the backend application we write, since not running on its own server machine, will need to be cross-platform. We will need to take this into consideration during development so that deployment is as smooth as possible.

Accessibility is an important factor in professional web design. Since we aim to be professionals, we too will consider whether the features we are adding support accessibility at least to some extent. We will use the standards defined in the Web Content Accessibility Guidelines version 2.0 during our initial design and then step up to 2.1 as suggested by Mozilla Developer Network.

VI. Team Members

Logan Kloft has basic knowledge about HTML, CSS, and JavaScript. He looks forward to learning new tools for this project and contributing across the board to both the frontend and backend.

Matthew Gerola is new to HTML, CSS and Javascript. He looks forward to learning new frameworks for the project and helping in the frontend and backend development of the project.

Manjesh Puram has entry level experience with HTML, CSS, and JavaScript. He is excited to be working with new tools and frameworks which will allow him to gain real-world knowledge to be successful. His main aim is to develop his front end skills but is eager to put in the effort to work on backend and learn the full-stack structure.

Jeremiah Lynn has intermediate knowledge about HTML, CSS, and Javascript. He has previous experience working with web development (in an offline environment). He worked with angular and typescript, along with extendscript (which is an extended ES3 Javascript). He looks forward to gaining experience working with backend technologies and learning react.