Sugarizer VueJS Core

Project Proposal

Basic Details

Project Proposal By : Hazem Khaled Kasim
Email : hazemkasim3@gmail.com
Linkedin : https://www.linkedin.com/in/hazemkak
Github: https://www.github.com/hazemkak
My CV: Hazem_Kasim.pdf
Time zone: Eastern European Standard Time (GMT+2)
Country : Egypt

Sugarizer activity video :
https://discord.com/channels/1078051575580336249/1078054265517506681/1080645352010559498

More About Me:
I'm a student at Cairo university faculty of engineering. My major is computer engineering.
I have 2 years experience in developing multi language frontend websites with a solid experience in optimization techniques such as **optimising images** , **lazy loading** , **dynamic imports** & **caching in browsers**.
I have developed various types of websites such as Dashboards , E-commerce , Games & Entertainment.
I'm aware of performance of websites and uses some tools to be able to analyse website performance as Lighthouse and Webpack analyzer bundle.
My Technical Experience:

- **Software Engineer at Sumerge**: Feb 2023 - PRESENT
    - My responsibilities there were to participate in designing project architecture using DDD approach. I participated in adding features to the project in an Agile environment where I developed some features in Frontend along with developing logic in microservices. We use TDD to ensure high quality and well tested code.
    - Technologies :
        - Node.js , Nest.js , Next.js , Redux , Docker , Apache Kafka , AWS , React.js
- **Fullstack Engineer at Intella** : Feb 2022 - Feb 2023
    - Working with ui/ux designer to develop modern designed websites.
    - Optimizing an inefficient web application's React.js , Css & assets delivery. Improving the page speed. Decreasing blocking time.

    - Debug issues with existing software to locate underlying problems and provide a fix.
    - Improve Project’s architecture and design.
    - Implementing unit/integration tests for the application.
    - building microservices , implementing graphql/rest endpoints , implementing message broker queueing for performance enhancment.
    - Technologies: Typescript, React , Next.js , QraphQL , Redux , Ruby on rails , Postgresql , Django, Celery, RabbitMQ, AWS, Node.js & Nest.js
    - **Check This Website where I built it from scratch** collaborating with ui/ux designer at the company https://www.anakoon.com I used in it Next.js along with Apollo Graphql & Redux
- **Fullstack Engineer at Ajax business solutions** : Aug 2021 - Jan 2022
    - Developed applications & systems for companies & factories.
    - Designing Responsive Pages with a good UI/UX.
    - Implementing Restful API.
    - Technologies : Typescript React , Go-lang ‘Fiber’ & Mongodb

Project Details

What am I going to make:

First I’m going to start from a business point of view by displaying the user stories which will be needed in the project.

First Screen User stories:
- I want to be able to see tutorial tour to make me familiar with the screen
- I want to be able to register as a new user
- I want to be able to login as an existing user

Login Screen User stories:
- I want to be able to enter my username and password to login
- I want to be notified with error message if credentials are invalid
- I want to be able to proceed to my profile after a successful login

Register Screen User stories:
- I want to be able to enter my username and password
- I want to be notified with error message if anything gone wrong ex: form input is empty while being required or username is already exists
- I want to be able to choose my avatar colour after successful registration
- I want to be able to proceed to home page after successful registration

Home View User stories:
- I want to be able to see a tutorial tour to make me familiar with navigating through the screen
- I want to be able to click on a button to see the tutorial tour again
- I want to be able to see all Sugarizer available activities in my home screen

- I want to be able to see if I’m connected to a server or not
- I want to be able to see some information about the server I’m connected to when hovering on it for example
- I want to be able to see my history in Sugarizer activities when hovering on them
- I want to be able to switch between home screen and neighbourhood screen
- I want to be able to search on a specific activity on my home screen
- I want to be able to navigate to my journal
- I want to be able to start and navigate to any Sugarizer activity
- I want to see notification when someone is around in my neighbourhood “bonus”
- I want to be able to navigate to my settings
- I want to be able to logout

List View:
- I want to be able to list all Sugarizer activities
- I want to be able to search into activities
- I want to be able to favourite an activity
- I want to be able to remove an activity from favourites
- I want to be able to sort activities with respect to alphabets or favourite or not
- I want to be able to see the activities in a paginated manner

Settings User stories:
- I want to be able to adjust my personal details “username, password & avatar colour”
- I want to be able to view my computer details
- I want to be able to see information about the server I’m connected to
- I want to be able to manage the server I’m connected to “connect & disconnect from it”
- I want to be able to delete my account
- I want to be able to adjust my privacy settings
- I want to change my language to another language
- I want to be able to save my settings to be applied

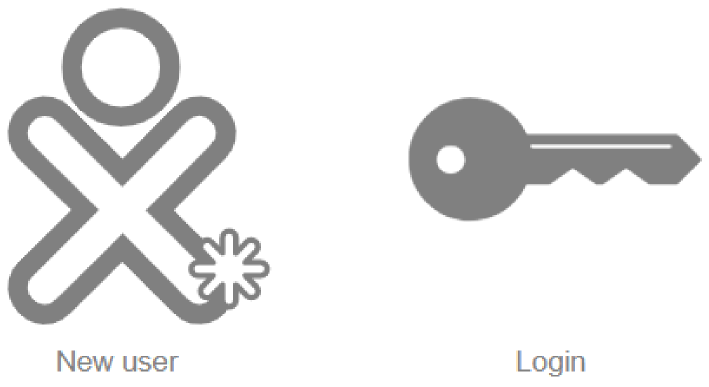
Second I’m gonna talk about the technical side of the project and let you know more about how I am going to implement the project and I will propose some ideas which would make the project more stable and mature.

Implementation:
I would suggest using Vue CLI since it’s built on top of webpack which will give us a lot of features such as bundling our application , js minify , css minify & a lot more which could make our application more reliable and easily deployed.
I’m going to use the TDD approach to ensure that my features are well tested and implemented so we can achieve good testing coverage. I’m going to use jest in implementing unit & integration tests.
I will refactor the project to implement a well structured architecture where we will be able to separate business logic from UI which will make code more loosely coupled to make it easier to be maintained in the future.
I will make documentation to my code to make it easier for others to understand it and use it and further implementation.
I will introduce Vuex to make it easier for global state management since we will need it sometime when the project becomes more complex.

I would also suggest to use Typescript which will make our life easier in debugging and maintaining the code when the project's complexity increase.

UI Components & API Calls Usage:

- We first may need some global components such as Global Alert & Global Loader to show Loader and alerts globally among any screen in the application.
- First Screen:
    - Components
        - Will need to implement Components to be used similar to original Sugarizer Application but will add to it a modern and stylish touch
        - Clicking on New User should redirect you to a register form
        - Click on Login should redirect you to a Login form
        - I will implement also a component which make the user able to view the tutorial tour again



    - API Calls:
        - GET /tutorial/steps :
            - Will just be used if we wish to get the tutorial tour steps dynamically from the server and not hard coded in the frontend source code
            - Main component in screen will be used on mount
- Login & Register Screen:
    - Components:
        - I will use Button 1 & Button 2 to display back & next buttons
        - I will use an input component to make user able to enter his/her username
        - I will use Password input demo component for user to enter it's password
    - API Calls:
        - POST /auth/login
            - Body => {username:string,password:string}
            - Response => {status:200 , userData:Object , userStarredActivities: Object , userSettings: Object}
            - Login button will be used to emit this event
    - Local Storage:

        - Will be used to save user data in response
- Home View :
    - Components:
        - Search input component for user to be able to search within activities
        - Button to be able to connect to server
        - Button Icons to represent activities
        - Dropdown component to be able to appear for options when user hover on any icon
        - Icon Button to be able to connect to server
    - API Calls:
        - GET /user/connect-server
            - Dialog box of server will be used to send request
        - POST /user/update-history
            - The activity icon button will emit the event to trigger the http request
    - Local Storage:
        - May be used to save user history instead of /user/update-history request
- Settings :
    - Components:
        - The dialog box component to show the options in settings
        - The dialog item to view the buttons of each option in settings
    - API Calls:
        - PUT /user/update-settings
            - The save button will emit the event
- List View:
    - Components:
        - The search component
        - List with an Item List component to view the activities in a list form
        - Start button icon to be able to star activities
    - API Calls:
        - PUT /user/activity/:activity_id
            - The star icon button will emit the event

How will this impact Sugar Labs:

It will make the Sugarizer application core more mature and stable and ready for more features to be added without being stuck in a strongly coupled project. I will make it loosely coupled by separating business logic from UI components , making unit & integration tests on logic by implementing testable logic by using TDD in implementing. I can use Typescript in implementing if approved to be used which will make code more readable and easier to be debugged and maintained . This will make the Sugarizer application able to last long without being deprecated and be more maintainable.

Technologies I will use:

- Vue.js
- Jest

- Javascript
- Docker
- CI/CD tool “ex: github actions”

Timeline

Week Number	Task Description
Week 1 & 2 29 May - 11 Jun	● Introduce myself to the mentor and the organization ● Refine the requirements of the project with the mentor and discuss the implementation details and documentation of the project ● Discuss the best technology tools that will help us in tasks
Week 3 & 4 12 Jun - 25 Jun	● Implementing First Screen user stories ● Writing unit & integration tests of First screen ● Implementing Register user stories ● Integrating Register with the server API ● Implementing Register unit & integration tests
Week 5 & 6 26 Jun - 9 Jul	● Implementing Login user stories ● Integrating Login with the server API ● Writing unit & integration tests of Login ● Starting in implementing some of Home view user stories
Week 7 & 8 10 Jun - 23 Jun	● Continue in implementing the rest of Home view user stories ● Writing unit & integration tests for the Home view ● Introducing the global state management VueX in the project
Week 9 & 10 24 Jun - 8 Aug	● Implementing Settings User Stories ● Integrating Settings with the server API ● Writing unit & integration tests for Settings
Week 11 & 12 11 Aug - 25 Aug	● Implementing List View User Stories ● Integrating List View with server API ● Writing unit & integration tests for List View
Week 13 & 14 26 Aug - 9 Sept	● Writing Documentation ● Having more time for debugging and fixing any errors ● Dockerize the application to be easier in deployment

Planned GSoC working hours:
I will be able to work an average of 5-6 hours a day and even more if the project turns out to demand more working hours from me.
Important Note: For the first 2-3 weeks, I will be having my final exams so I will be able to work around 2-3 hours a day.
If I was late because of my finals I will work really hard after it to catch up.

Post Plans:
I would like to continue contributing to Sugar Labs after GSoC and implement more features for the project.

Planned Absence/Vacation days:
I’m not willing to have any absence days during the period of the GSoC project.
If any emergency occurs , then I will inform my mentor.