

# Vue, Part 1

Recitation 6

# Plan



**01** Front End Starter Code Walkthrough

**02** Discussion / Resources

**03** Questions / Wrap-up



# 01 Front End Starter Code!

# Create Your Repo:

**<https://github.com/61040-fa23/frontend-starter>**



# Getting Started

1. Run `npm install`
2. Copy over your `.env` file from your backend code into the root folder
3. Run `npm run dev:server` to start the server. Run `npm run dev:client` to start the client



**After recitation: Read the rest of the README for instructions on transferring over your backend code and deployment!**



# Repo Contents



# Overview

```
frontend-starter/  
├── api/  
├── client/  
│   ├── assets/  
│   │   └── images/  
│   ├── components/  
│   ├── router/  
│   ├── stores/  
│   ├── utils/  
│   ├── views/  
│   └── App.vue  
├── public/  
├── server/  
└── index.html/
```

**api** – connection with server-side routes (what you wrote last week)

**public** – top-level static assets

- Change your site favicon here!

**server** – your backend code!

**index.html** – app-level headers

# Overview

```
frontend-starter/  
├── api/  
├── client/  
│   ├── assets/  
│   │   └── images/  
│   ├── components/  
│   ├── router/  
│   ├── stores/  
│   ├── utils/  
│   ├── views/  
│   └── App.vue  
├── public/  
├── server/  
└── index.html/
```

**client** – all relevant frontend code

- **assets** – assets that are only compiled if necessary
  - store your images in **images/**
- **components** – your app's components
- **router** – page-level routing
- **stores** – keep track of app state (i.e. current user)
- **utils** – general utility functions
- **views** – your app's pages
- **App.vue** – app definition





# client/assets

- **images** – where you should store all your relevant images
- Contains global css files (including css variables)

```
# main.css ×
client > assets > # main.css > ...
Grace Huang, 1 hour ago | 1 author (Grace Huang)
1 /* Add some global styles here! */
2
3 :root {
4   --red:   ■ rgb(208, 67, 12);
5   --base-bg: ■ #f1f3f5;
6 }
7 Grace Huang, 1 hour ago • Initial commit
8 button.btn-small {
9   font-size: 80%;
10 }
11
12 button.button-error {
13   color: ■ white;
```

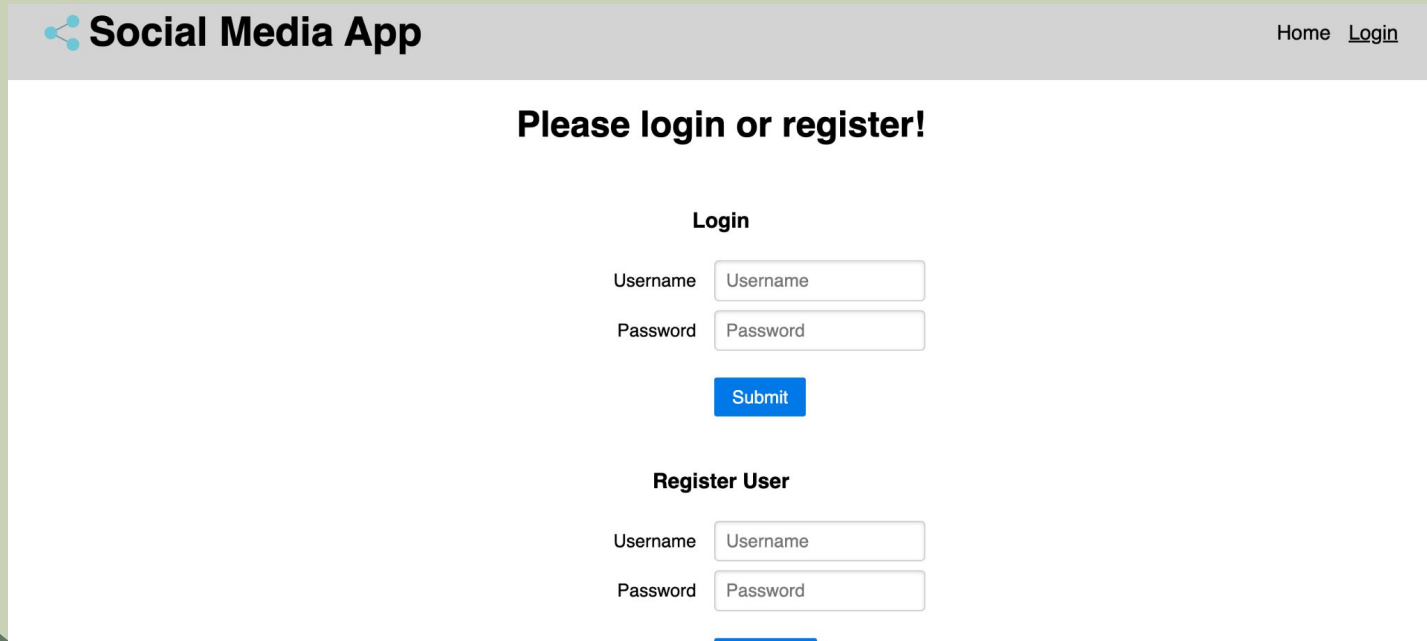


# client/components

- All the components for your app
  - Each component will generally correspond to a certain section of your app
    - Structuring is up to your discretion, but keeping things modular will make your life a lot easier!
- 
- 

# client/views

- Pages for your app, where each **view** corresponds to 1 page layout



The screenshot shows a web page for a "Social Media App". The header is grey and contains the app name on the left and navigation links "Home" and "Login" on the right. The main content area is white and features a central heading "Please login or register!". Below this heading are two sections: "Login" and "Register User". Each section contains a "Username" label and a text input field, a "Password" label and a text input field, and a blue "Submit" button. The "Register User" section is partially cut off at the bottom of the image.

**Social Media App** [Home](#) [Login](#)

## Please login or register!

**Login**

Username

Password

[Submit](#)

**Register User**

Username

Password



# client/router

- Client-side routing to move between your different pages
- We use vue-router: <https://router.vuejs.org/guide/>

```
TS index.ts 9+ X
client > router > TS index.ts > [🔍] router > 🔑 routes
9
10  const router = createRouter({
11    history: createWebHistory(),
12    routes: [
13      {
14        path: "/",
15        name: "Home",
16        component: HomeView,
17      },
18      {
19        path: "/setting",
20        name: "Settings",
21        component: SettingView,
22        meta: { requiresAuth: true },
23      }
24    ]
25  })
```





# client/stores

- Variables/functions for keeping track of the **state** of your app
  - This should only store information that should persist across refreshes on a user's device
    - **Not a substitute for mongodb**
- 
- 



# client/utills



- **Utility methods** that you might find useful across different components/concepts
  - Feel free to make your own functions! If you find yourself writing the same processing code for many different areas of the application, it'll make your life easier if you extract them into a utills file
- 
- 

# Component Deep Dive





# client > components > Post

- An example of a set of components that helps a user create/edit/view posts
- 
- 






# Styling

Generally **scoped** within the file it's in

- Applies only to the **specific component**

```
✓ <style scoped>
✓ form {
  background-color: var(--base-bg);
  border-radius: 1em;
  display: flex;
  flex-direction: column;
  gap: 0.5em;
  padding: 1em;
}
```



# Backend Communication

- Wrapper around fetch using `fetchy` (in `utils/fetchy.ts`)
- `emit` communicates events back to their parent components

```
8   const createPost = async (content: string) => {
9     try {
10      await fetchy("api/posts", "POST", {
11        body: { content },
12      });
13    } catch (_) {
14      return;
15    }
16    emit("refreshPosts");
17    emptyForm();
```


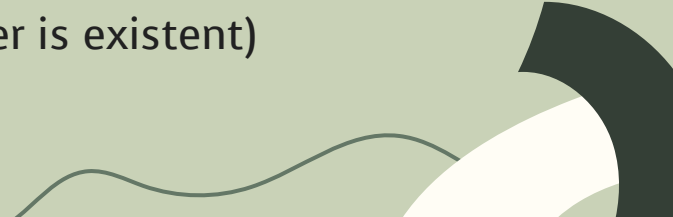


**02 Discussion /  
Resources**



# Client-side Validation



Best practices:

- Frontend shouldn't be more restrictive than the backend – if you allow certain responses in the backend, you should be able to handle that in the frontend as well
  - Validating things that can be **easily** checked on the frontend side can make you make less irrelevant requests
    - I.e. checking input format
    - But you shouldn't do things that would require a database call anyways (i.e. checking whether a queried user is existent)
- 
- 



# General Coding Resources





- Vue 3 Tutorial:  
<https://www.youtube.com/playlist?list=PL4cUxeGkcC9hYYGbV60Vq3IXYNfDk8At1>
  - Vue Style Guide: <https://v2.vuejs.org/v2/style-guide/?redirect=true>
  - Vue guide: <https://vuejs.org/guide/introduction.html>
  - Our router: <https://router.vuejs.org/guide/>
- 
- 



# Styling/Formatting Resources



- Icon library: <https://fontawesome.com/>
  - MDN web docs:  
<https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>
  - Pure.css: <https://purecss.io/>
    - What we currently use in the front end, but feel free to ignore it and just use plain CSS
- 
- 



**03** Questions /  
Wrap-up



<https://forms.gle/6pRoSsZ8Wy5zZBku7>