Frontend Engineer Test

**Prerequisite**

1. pnpm version ^8.0

**Before you start**

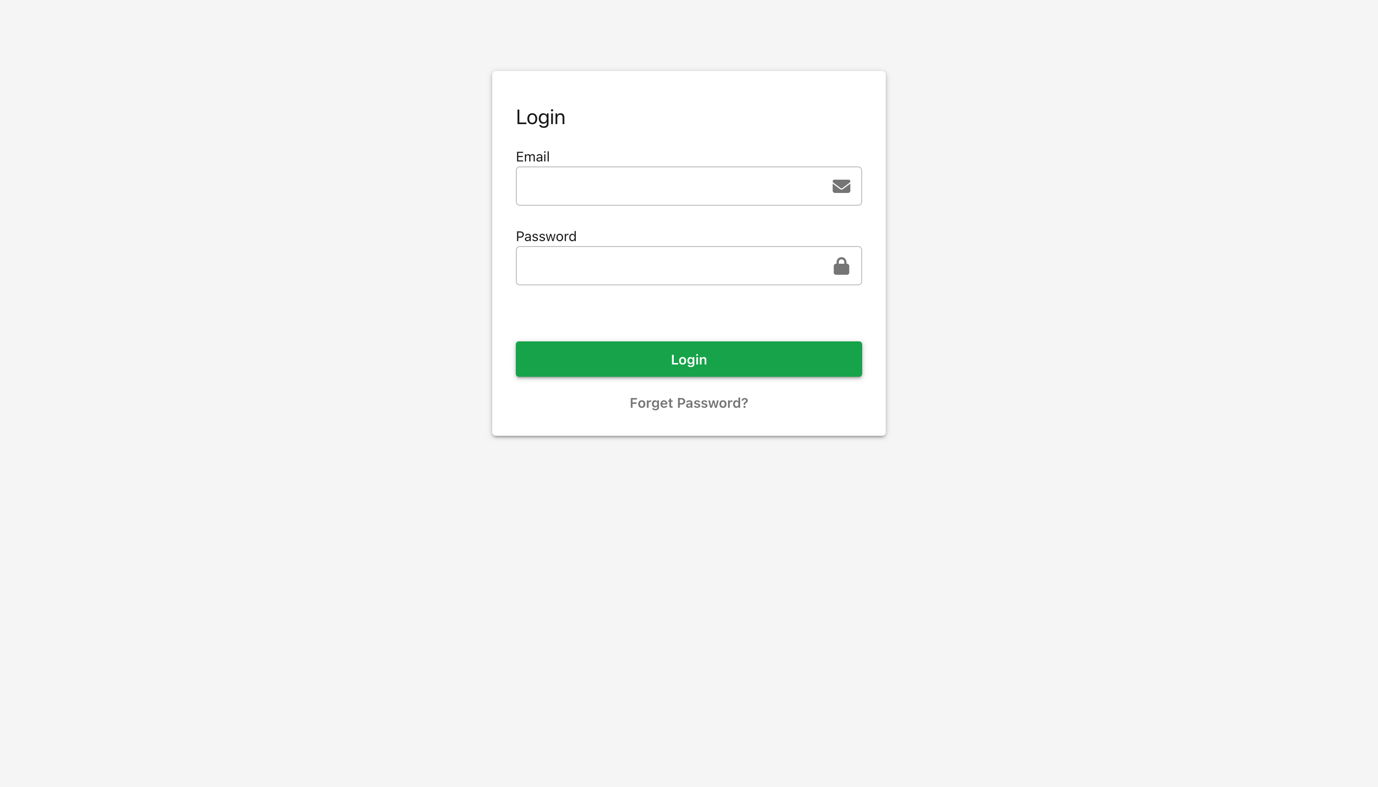
You are expected to:

* develop the interfaces using the Quasar library (<https://quasar.dev/>)
* use Pinia as the Vue store (if required)
* use Vue Router for page navigations
* handle common error handling on user input (e.g. invalid email format on a text input, text on a numeric input, etc.) and server response
* ensure that all interfaces are mobile friendly

**Tasks**

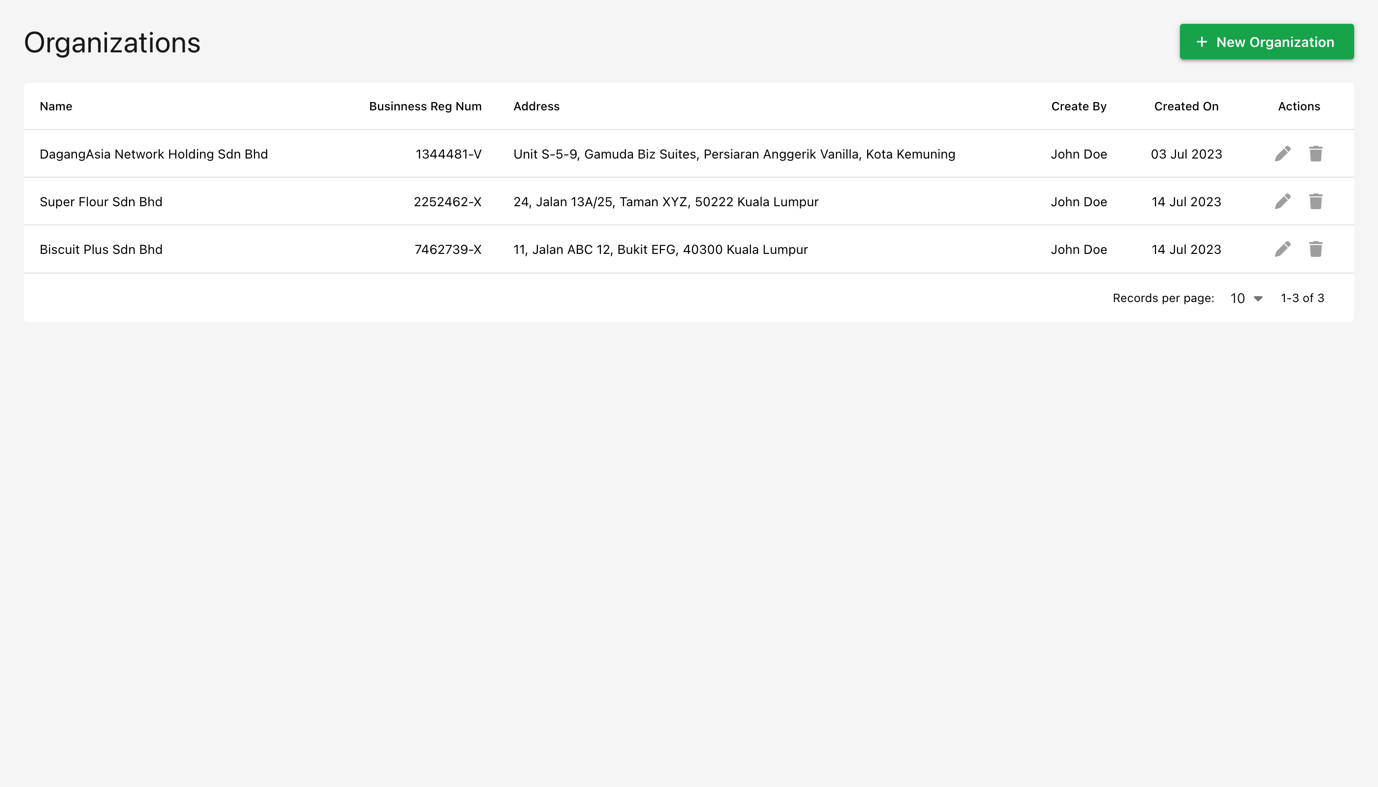
The Vue 3 + Typescript codebase provided contains some models and a mock API helper. Based on the pattern in the codebase, create necessary files and codes for the following:

1. Create a Login page following the design shown below. A mock login API can be found in the API helper. Upon successful login, redirect user to the Organization page.



1. Create an Organizations page following the design shown below. A mock API that returns a list of organization can be found in the API helper. The page should show a list of organizations in a table with the following columns:
   1. Name
   2. Business Reg Num
   3. Address
   4. Create By (first name + last name of user)
   5. Create On
   6. Actions (with 1 Edit + 1 Delete button)

The ‘Edit’ button redirects the user to the ‘Organization Detail’ page, while the ‘Delete’ button removes the organization from the list. The ‘New Organization’ button redirects the user to the ‘New Organization’ page.



1. Create an Organization Detail page following the design pattern in previous tasks. The page should show all the information for the selected organization. The page should allow user to modify the organization information after clicking on the ‘Edit’ button. After clicking on the ‘Save’ button, the information is submitted.
2. Create a New Organization page following the design pattern in previous tasks. The page should allow user to create a new organization after entering all the required information. Upon succesful creation, redirect user to the Organization Detail page for the organization created.
3. Create a navigation interface that includes a ‘Organizations’ button that redirects to the Organizations page, and a ‘Logout’ button that redirects to the login page. The navigation interface should only be shown for users who has logged in.
4. Assume that each user has a set of permissions (e.g. Organization.View, Organization.Create, etc.), propose in detail, an authorization method to control what each user can see or perform on the frontend. (Note: you do not need to code this, snippets of code/pseudocode can be shown to assist in explanation)