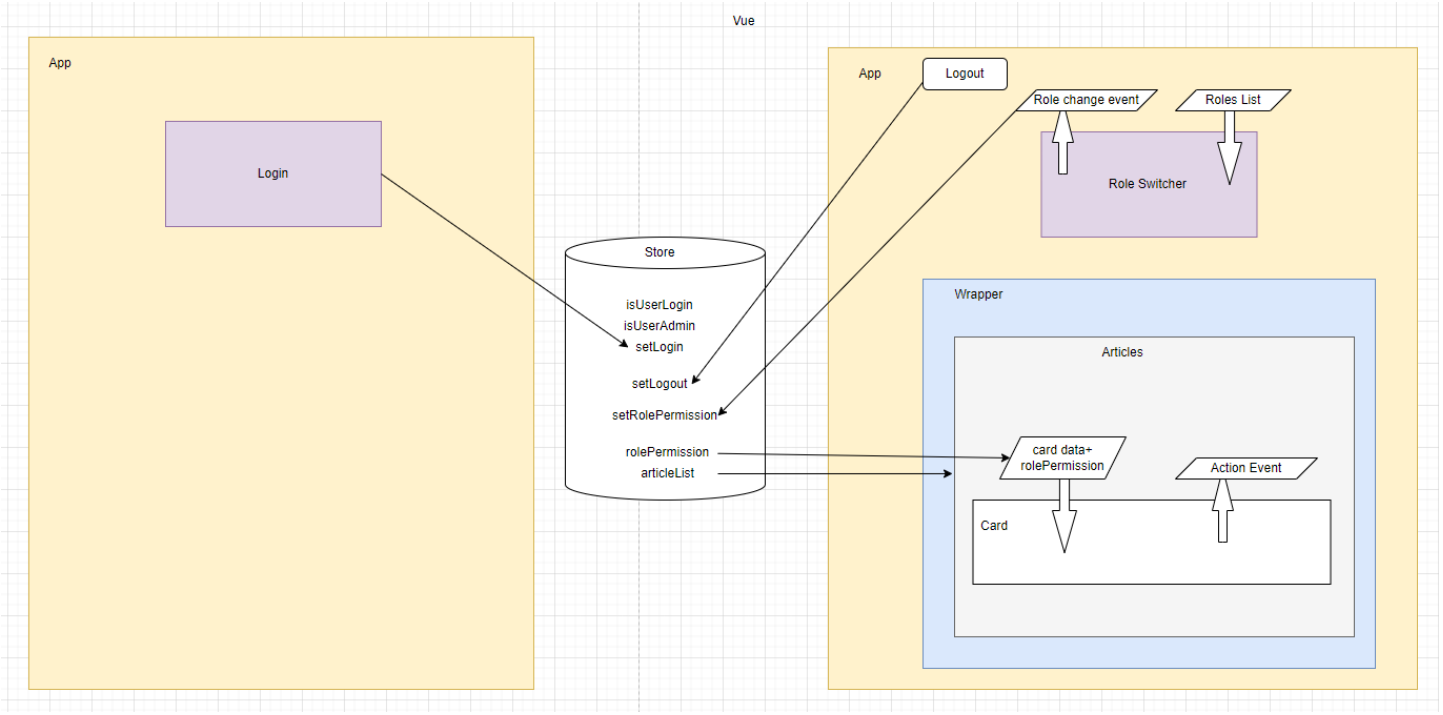
High level design of the Front-End system



Workflows:

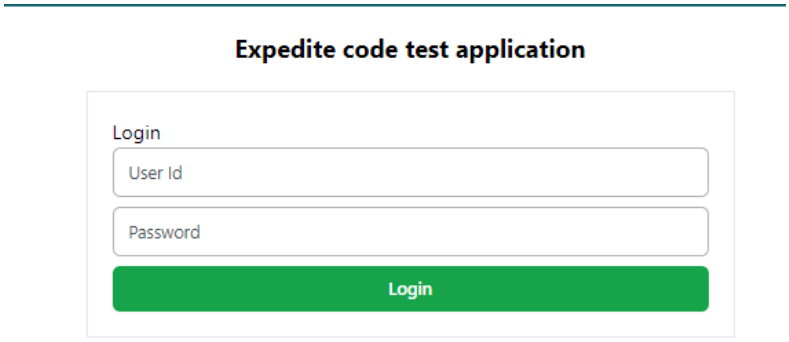
Public User:
Login→ can see listed articles → no actions can be performed.

Admin User(john):
Login→ Role selector widget is visible.

On selection of role the role is set in store and permission is captured by Articles component

It passes the Role to Card → Card renders relevant action button as per role.

Login Component [Stand Alone]



The Login is statically managed here, in real world it will be validate from BE.
For now use these login credentials

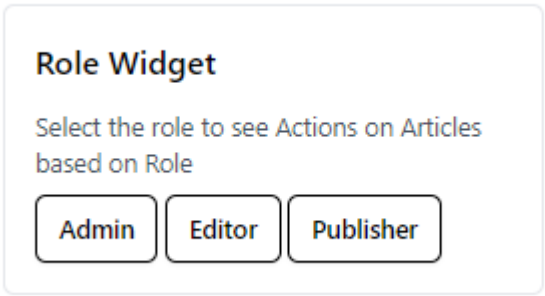
Admin:
User Id: john

Password: 123

Public:
User Id : <anytext>

Password:123

RoleWidget Component: [Stand Alone]



Input Data:
This component takes roles and renders them.

Event:
When user clicks on role, It returns event with selected role to parent component

Card Component: : [Stand Alone]

Input :
Card data Object and role-permission

Event:
It returns click event on relevant buttons[edit/delete/publish] with id to take action by parent.

Wrapper: [A wrapper component to created nested component and prop drilling scenario]

Articles: [Renders all article by using Card component and handle events coming from Card]

Article takes data from store which is set at time of login**,** to avoid prop drilling

Scalability:

- The Role widget can take dynamic roles coming from API and once a role is selected, it will give event on that.
- The Card component sends event to parent component so in future if more action required we can just add the action button, it is not coupled to action logic

Improvements:

- Login is static and with limited validation.it can be improved when connected to BE.
- Card component can be designed to take Dynamic list of action buttons.
- Router can be used to separate login, admin and public routes.
- Validation logic based on Role through BE APIs