САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет по лабораторной работе №3
Разработка одностраничного веб-приложения (SPA) с использованием фреймворка Vue.JS

Выполнил:
Елистратов В. Д.
Группа К32392
Проверил:
Харитонов Антон

Санкт-Петербург
2024 г.

Содержание отчета

Постановка задачи

Мигрировать ранее написанный сайт на фреймворк Vue.JS.

Минимальные требования:

- Должен быть подключён роутер
- Должна быть реализована работа с внешним API
- Разумное деление на компоненты

1) API

1.1) Vault API

```
class CapsulesApi {
    constructor(instance) {
        this.API = instance;
    }

    getOneCapsule = async(id) => {
        return this.API({
            method: "GET",
            url: `/664/capsules?id=${id}`,
        });
    }

    getAllCapsules = async () => {
        return this.API({
            method: "GET",
            url: "/664/capsules",
        });
    };

    getMyCapsules = async (userId, accessToken) => {
        return this.API({
            method: "GET",
            url: `/664/capsules?userId=${userId}`,
            'Authorization': `Bearer ${accessToken}`,
        });
    };
    getNotMyCapsules = async (userId, accessToken) => {
        return this.API({
            method: "GET",
            url: `/664/capsules?userId_ne=${userId}`,
            'Authorization': `Bearer ${accessToken}`,
        });
    };

    getOpenedCapsules = async () => {
        return this.API({
            method: "GET",
            url: `/664/capsules?openDate_lte=${new Date().getTime()}`,
        });
    };
```

```
    getClosedCapsules = async () => {
        return this.API({
            method: "GET",
            url: `/664/capsules?openDate_gte=${new Date().getTime()}`,
        });
    };

    createCapsule = async (data, accessToken) => {
        return this.API({
            method: "POST",
            url: "/664/capsules",
            data,
            headers: {
                "Content-Type": "application/json",
                'Authorization': `Bearer ${accessToken}`
            },
        });
    };
}

export default CapsulesApi;
```

Метод **getOneCapsule** принимает ID и возвращает капсулу, соответствующую этому ID

Метод **getAllCapsules** возвращает все капсулы из БД

Метод **getMyCapsules** принимает ID пользователя и его AccessToken и возвращает капсулы, созданные этим пользователем, доступно только залогинившимся

Метод **getNotMyCapsules** принимает ID пользователя и его AccessToken и возвращает капсулы, созданные не этим пользователем, доступно только залогинившимся

Метод **getOpenedCapsules** возвращает уже открывшиеся капсулы

Метод **getClosedCapsules** возвращает только те капсулы, срок открытия которых еще не наступил

Метод **createCapsule** принимает необходимые для создания данные - data и AccessToken пользователя, доступно только залогиненным

1.2) User API

```
class UsersApi {
    constructor(instance) {
        this.API = instance;
    }

    userLogin = async (data) => {
        return this.API({
            method: "POST",
            url: "/login",
            data,
            headers: {
                'Accept': 'application/json',
                'Content-Type': 'application/json',
            }
        });
    };

    userUpdate = async (accessToken, data) => {
        return this.API({
            method: "PUT",
            url: `/users/${data['id']}`,
            data,
            headers: {
                'Accept': 'application/json',
                'Content-Type': 'application/json',
            }
        });
    };

    userRegistration = async (data) => {
        //console.log("Api", data);
        return this.API({
            method: "POST",
            url: "/users",
            data,
            headers: {
                'Content-Type': 'application/json'
            }
        });
    };
}

export default UsersApi;
```

Метод **userLogin** принимает необходимые для логина данные - data (пароль и email) и возвращает полные данные о пользователе, а также AccessToken

Метод **userUpdate** принимает данные для замены - data, а также AccessToken и отправляет запрос на обновление данных в БД

Метод **userRegistration** принимает данные для регистрации пользователя - data, а также AccessToken и отправляет запрос на добавление данных в БД.

Если Данные успешно добавлены, пользователь будет авторизован (запрос вернет полную информацию о пользователе и AccessToken)

2) Stores

Хранение реализовано через библиотеку Pinia

2.1) Vault Stores

```
import { defineStore } from "pinia";
import { capsulesApi } from "@/api";

const capsulesStore = defineStore("capsules", {
    state: () => ({
        capsules: [],
        capsule: {
            "name":"None",
            "openDate":1701644520000,
            "description":"None",
            "access":false,
            "text":"None",
            "files":"None",
            "userId": undefined,
            "userName": "None",
            "id": undefined
        },
        selector: "All",
    }),
    actions: {
        async loadCapsules() {
            const data = JSON.parse(localStorage.pinia_users);
            const accessToken = data['token'];
            if( accessToken !== "" ) {
                const user = JSON.parse(data['user'])
                if (this.selector === "My") {
                    const response = await
capsulesApi.getMyCapsules(user['id'], accessToken);
                    this.capsules = response.data;
                    return response;
                }
                if (this.selector === "notMy") {
                    const response = await
capsulesApi.getNotMyCapsules(user['id'], accessToken);
                    this.capsules = response.data;
                    return response;
                }
            }
            if(this.selector === "All"){
                const response = await capsulesApi.getAllCapsules();
                this.capsules = response.data;
                return response;
            }
            if(this.selector === "opened"){
                const response = await capsulesApi.getOpenedCapsules();
                this.capsules = response.data;
                return response;
            }
            if(this.selector === "closed"){
                const response = await capsulesApi.getClosedCapsules();
                this.capsules = response.data;
                return response;
            }
        },
```

```
        async createCapsule(data, accessToken) {
            const response = await capsulesApi.createCapsule(data,
accessToken);
            this.capsules = response.data;
            return response;
        },

        async loadOneCapsule(id){
            const response = await capsulesApi.getOneCapsule(id);
            this.capsule = response.data[0];
            return response.data[0];
        }
    },
});

export default capsulesStore;
```

Состояние хранит:

- Список капсул отображающихся на странице – **capsules: []**
- Капсулу отображающуюся в модальном окне при открытии капсулы – **capsule: {}**
- Состояние селектора для сортировки - **selector**

Метод **loadCapsules** в зависимости от состояния селектора вызывает один из API запросов и записывает результат в capsule

Метод **createCapsule** вызывает метод добавления капсул

Метод **loadOneCapsule** принимает ID капсулы и записывает данные о ней в capsule

2.2) User Stores

```
import { defineStore } from "pinia";
import { usersApi } from "@/api";

const usersStore = defineStore("users", {
    state: () => ({
        user: "",
        token: "",
    }),

    getters: {
      authCheck: state => {
          return (state.token !== "")
      }
    },

    actions: {
        async loginUser(data) {
            const response = (await usersApi.userLogin(data))['data'];
            const {accessToken, user} = response
            this.user = JSON.stringify(user);
            this.token = JSON.stringify(accessToken);
            return response;
        },
        async logoutUser() {
            this.user = '';
            this.token = '';
            return "success";
        },
        async createUser(data) {
            const response = (await
usersApi.userRegistration(data))['data'];
            const {accessToken, user} = response
            this.user = JSON.stringify(user);
            this.token = JSON.stringify(accessToken);
            return response;
        },
    },
});

export default usersStore;
```

Состояние хранит:

- Текущего пользователя если пройдена аутентификация и не хранит ничего в противном случае – **user: ""**
- AccessToken пользователя если пройдена аутентификация и не хранит ничего в противном случае – **token: ""**

Метод **loginUser** отправляет запрос на вход, разбирает ответ от API на user-а и AccessToken, записывает их в state

Метод **logoutUser** очищает state и перенаправляет пользователя на главную

Метод **createUser** служит для создания нового пользователя, при успешной регистрации разбирает ответ от API на user-а и AccessToken, записывает их в state.

3) Views

3.1) Main Page

```
<template>
  <header class="rounded">
    <div class="row justify-s-a" v-if="checkLogin">
      <logged-main-header/>
    </div>
    <div v-else class="row justify-s-a">
      <main-header/>
    </div>

  </header>
  <main class="rounded">
    <div class="container">
      <div class="row justify-c" v-if="checkLogin">
        <logged-selector/>
      </div>
      <div v-else class="row justify-c">
        <selector/>
      </div>
      <hr class="content-dividing-line rounded mt-3 mb-5"/>
      <div id="vaultList" class="row">
        <div class="col-xl-3 col-md-4 col-sm-6 mb-3" v-for="capsule in
this.capState.capsules" :key="capsule.id">
          <div v-if="getCapsuleStatus(capsule.openDate)">
            <vault-opened
                :id="capsule.id"
                :name="capsule.name"
                :description="capsule.description"
                :creator="capsule.userName"
                :open-date="new Date(capsule.openDate)"
            />
          </div>
          <div v-else>
            <vault-closed
                :id="capsule.id"
                :name="capsule.name"
                :description="capsule.description"
                :creator="capsule.userName"
                :open-date="new Date(capsule.openDate)"
            />
          </div>
        </div>
      </div>
    </div>
  </main>

  <footer>
    <footer-c/>
  </footer>

  <open-vault-modal/>

</template>

<style scoped>

</style>
```

```
<script>
  import MainHeader from "@/components/Main/Header/mainHeader.vue";
  import capsulesStore from "@/stores/capsules.js";
  import {mapActions, mapGetters, mapState} from "pinia";
  import RegModal from "@/components/auth/regModel.vue";
  import LoginModal from "@/components/auth/loginModel.vue";
  import LoggedSelector from
"@/components/Main/Selector/loggedSelector.vue";
  import Selector from "@/components/Main/Selector/selector.vue";
  import usersStore from "@/stores/user.js";
  import LoggedMainHeader from "@/components/Main/Header/loggedHeader.vue";
  import VaultCreationModal from
"@/components/Main/Vaults/creationCapsuleModal.vue";
  import VaultOpened from "@/components/Main/Vaults/vaultOpened.vue";
  import VaultClosed from "@/components/Main/Vaults/vaultClosed.vue";
  import FooterC from "@/components/footerC.vue";
  import OpenVaultModal from
"@/components/Main/Vaults/openCapsuleModal.vue";
  import {Modal} from "bootstrap";

  export default {
    name: "homaPage",
    data(){
      return{
        usState: usersStore(),
        capState: capsulesStore(),
      }
    },
    components: {
      OpenVaultModal,
      FooterC,
      VaultCreationModal, LoggedMainHeader, LoggedSelector, LoginModal,
RegModal, Selector, MainHeader, VaultClosed, VaultOpened,},
    computed: {
      checkLogin(){
        return this.usState.authCheck
      }
    },
    methods: {
      getCapsuleStatus(date){
        if(((new Date(date)).getTime() <= (new Date()).getTime())){
          return true;
        }
        else{
          return false;
        }
      }

    },

    mounted() {
      this.capState.loadCapsules()
    },
  };

</script>
```

Представление для отображения главной страницы сайта Оно содержить только базовый html код с «разметкой» остальные элементы

подгружаются в виде отдельных компонент. В компоненты передаются необходимые данные из PiniaStores

3.2) Profile Page

```
<template xmlns="http://www.w3.org/1999/html">
  <header class="rounded">
    <profile-header/>
  </header>
  <main class="rounded">
  <profile-info
      :userName="JSON.parse(this.usState.user)['username']"
      :firstName="JSON.parse(this.usState.user)['firstName']"
      :email="JSON.parse(this.usState.user)['email']"
      :capCount="JSON.parse(this.usState.user)['capCount']"
  />
  </main>

  <footer>
    <footer-c/>
  </footer>
</template>

<style scoped>

</style>

<script>
  import usersStore from "@/stores/user.js";
  import capsulesStore from "@/stores/capsules.js";
  import ProfileHeader from
"@/components/Profile/Header/ProfileHeader.vue";
  import FooterC from "@/components/footerC.vue";
  import ProfileInfo from "@/components/Profile/info/profileInfo.vue";
  import VaultOpened from "@/components/Main/Vaults/vaultOpened.vue";

  export default {
    name: "profilePage",
    data(){
      return{
        usState: usersStore(),
        capState: capsulesStore(),
      }
    },
    components: {
      VaultOpened,
      ProfileInfo,
      FooterC,
      ProfileHeader
    },

  };

</script>
```

Представление для отображения профиля пользователя

4) Router

```
import { createRouter, createWebHistory } from "vue-router";

const router = createRouter({
    history: createWebHistory(import.meta.env.BASE_URL),
    routes: [
        {
            path: "/vaulttec",
            name: "main",
            component: () => import("../views/TimeVault.vue"),
        },
        {
            path: "/vaulttec/profile",
            name: "profile",
            component: () => import("../views/TimeVaultProfile.vue"),
        },
    ],
});

// экспортируем сконфигурированный роутер
export default router;
```

5) Components

5.1) Headers / Navbars

```
<script>
  import RegModal from "@/components/auth/regModel.vue";
  import VaultCreationModal from
"@/components/Main/Vaults/creationCapsuleModal.vue";

  export default {
    name: 'loggedMainHeader',
    components: {VaultCreationModal},
  }
</script>

<template>
  <nav class="navbar navbar-expand-lg">
    <div class="container-fluid">
      <h1>TimeVault</h1>
      <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <svg class="navbar-toggler-icon" viewBox="0 0 16 16">
          <path fill-rule="evenodd" d="M2.5 12a.5.5 0 0 1 .5-.5h10a.5.5 0 0
1 0 1H3a.5.5 0 0 1-.5-.5m0-4a.5.5 0 0 1 .5-.5h10a.5.5 0 0 1 0 1H3a.5.5 0 0
1-.5-.5m0-4a.5.5 0 0 1 .5-.5h10a.5.5 0 0 1 0 1H3a.5.5 0 0 1-.5-.5"/>
        </svg>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav border-start border-2 ms-auto">
          <li class="nav-item ms-2 mb-1">
            <button type="button" class="btn btn-lg btn-my-main" data-bs-
toggle="modal" data-bs-target="#CreateCapsuleFormModalId" tabindex="0">
              Создать капсулу
              <svg id="addIcon" class="icons" viewBox="0 0 16 16">
                <path d="M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14zm0 1A8 8 0 1 0
8 0a8 8 0 0 0 0 16z"/>
                <path d="M8 4a.5.5 0 0 1 .5.5v3h3a.5.5 0 0 1 0 1h-3v3a.5.5
0 0 1-1 0v-3h-3a.5.5 0 0 1 0-1h3v-3A.5.5 0 0 1 8 4z"/>
              </svg>
            </button>
          </li>
          <li class="nav-item ms-2 mb-1">
            <button type="button" class="btn btn-lg btn-my-main"
onclick="location.href='/vaulttec/profile';" tabindex="0">
              Профиль
              <svg id="profileIcon" class="icons" viewBox="0 0 16 16">
                <path d="M11 6a3 3 0 1 1-6 0 3 3 0 0 1 6 0z"/>
                <path fill-rule="evenodd" d="M0 8a8 8 0 0 1 1 16 0A8 8 0 0 1
0 8zm8-7a7 7 0 0 0-5.468 11.37C3.242 11.226 4.805 10 8 10s4.757 1.225 5.468
2.37A7 7 0 0 0 8 1z"/>
              </svg>
            </button>
          </li>
        </ul>
      </div>
    </div>
  </nav>
```

```
  <vault-creation-modal/>
</template>

<style>

</style>
```

Компонента содержит Navbar, который отображается залогинившимся пользователям на главной странице

```
<script>
  import RegModal from "@/components/auth/regModel.vue";
  import LoginModal from "@/components/auth/loginModel.vue";

  export default {
    name: 'mainHeader',
    components: {RegModal, LoginModal},
  }
</script>

<template>
  <nav class="navbar navbar-expand-lg">
    <div class="container-fluid">
      <h1>TimeVault</h1>
      <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <svg class="navbar-toggler-icon" viewBox="0 0 16 16">
          <path fill-rule="evenodd" d="M2.5 12a.5.5 0 0 1 .5-.5h10a.5.5 0 0
1 0 1H3a.5.5 0 0 1-.5-.5m0-4a.5.5 0 0 1 .5-.5h10a.5.5 0 0 1 0 1H3a.5.5 0 0
1-.5-.5m0-4a.5.5 0 0 1 .5-.5h10a.5.5 0 0 1 0 1H3a.5.5 0 0 1-.5-.5"/>
        </svg>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav border-start border-2 ms-auto">
          <li class="nav-item ms-2 mb-1">
            <button type="button" class="btn btn-lg btn-my-main" data-bs-
toggle="modal" data-bs-target="#loginFormModal" tabindex="0">Вход</button>
          </li>
          <li class="nav-item ms-2 mb-1">
            <button type="button" class="btn btn-lg btn-my-main" data-bs-
toggle="modal" data-bs-target="#regFormModal"
tabindex="0">Регистрация</button>
          </li>
        </ul>
      </div>
    </div>
  </nav>
  <reg-modal/>
  <login-modal/>
</template>

<style>

</style>
```

Компонента содержит Navbar, который отображается не залогинившимся пользователям на главной странице

```
<script>
  import RegModal from "@/components/auth/regModel.vue";
  import usersStore from "@/stores/user.js";
  import capsulesStore from "@/stores/capsules.js";
  export default {
    name: 'ProfileHeader',
    data(){
      return{
        usState: usersStore(),
      }
    },
    methods: {
      logout(){
        this.usState.$reset()
        localStorage.pinia_users = '';
        //console.log(this.usState.user)
        location.replace("/vaulttec")
      }
    },
  }
</script>
<template>
  <nav class="navbar navbar-expand-lg">
    <div class="container-fluid">
      <h1>TimeVault</h1>
      <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav border-start border-2 ms-auto">
          <li class="nav-item ms-2 mb-1">
            <button type="button" class="btn btn-lg btn-my-main"
onclick="location.href='/vaulttec';">На главную</button>
          </li>
          <li class="nav-item ms-2 mb-1">
            <button type="button" class="btn btn-lg btn-my-logout"
@click="logout">Выйти</button>
          </li>
        </ul>
      </div>
    </div>
  </nav>
</template>
<style>
.btn-my-logout {
  --bs-btn-color: var(--color4);
  --bs-btn-bg: var(--color2);
  --bs-btn-border-color: var(--color3);
  --bs-btn-hover-color: var(--color4);
  --bs-btn-hover-bg: var(--color12);
  --bs-btn-hover-border-color: var(--color12);
  --bs-btn-active-color: var(--color4);
  --bs-btn-active-bg: var(--color12);
  --bs-btn-active-border-color: var(--color12);
}
</style>
```

Компонента содержит Navbar, который отображается на странице профиля

5.2) Selector

```
<template>
  <div class="btn-group" role="group">
    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="All" id="RadioBtnAll" tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnAll" > Все </label>

    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="opened" id="RadioBtnOpened"
tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnOpened" > Открытые </label>

    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="closed" id="RadioBtnClosed"
tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnClosed" > Закрытые </label>

  </div>
</template>

<style scoped>

</style>

<script>
import {mapActions, mapState} from "pinia";
import capsulesStore from "@/stores/capsules.js";

  export default {
    name: 'selector',
    data(){
      return {
        capsuleState: capsulesStore(),
      }
    },
    computed: {
      reloadCapsules(){
        this.capsuleState.loadCapsules();
      }
    },
  }


</script>
```

Компонента содержит Selector, который отображается не залогинившимся пользователям на главной странице

```
<template>
  <div class="btn-group" role="group">
    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="All" id="RadioBtnAll" tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnAll" > Все </label>

    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="opened" id="RadioBtnOpened"
tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnOpened" > Открытые </label>

    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="closed" id="RadioBtnClosed"
tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnClosed" > Закрытые </label>

    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="My" id="RadioBtnMy" tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnMy" > Созданные мной
</label>

    <input class="btn-check me-1" type="radio" v-on="reloadCapsules" v-
model="capsuleState.selector" value="notMy" id="RadioBtnNM" tabindex="0"/>
    <label class="btn btn-my-main" for="RadioBtnNM" > Созданные не мной
</label>
  </div>
</template>

<style scoped>

</style>

<script>
  import capsulesStore from "@/stores/capsules.js";

  export default {
    name: 'loggedSelector',
    data(){
      return {
        capsuleState: capsulesStore(),
      }
    },
    computed: {
      reloadCapsules(){
        this.capsuleState.loadCapsules();
      }
    },
    mounted() {
    }
  }

</script>
```

Компонента содержит Selector, который отображается залогинившимся пользователям на главной странице

5.3) Auth Modals

Компоненты необходимые для регистрации и логина

```
<template>
  <div class="modal fade modal-my-config" id="regFormModal" data-bs-
backdrop="static" data-bs-keyboard="false" tabindex="-1" aria-
labelledby="regFormModal" aria-hidden="true">
    <div class="modal-dialog">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="staticBackdropLabel">Регистрация</h5>
          <button type="button" class="btn btn-my-main" data-bs-
dismiss="modal" aria-label="Закрыть">
            <svg id="close" class="close-btn"  viewBox="0 0 16 16">
              <path fill-rule="evenodd" d="M8 2a.5.5 0 0 1 .5.5v5h5a.5.5 0
0 1 0 1h-5v5a.5.5 0 0 1-1 0v-5h-5a.5.5 0 0 1 0-1h5v-5A.5.5 0 0 1 8 2"/>
            </svg>
          </button>
        </div>
        <div class="modal-body">
          <form id="RegistrationForm" @submit.prevent="registration">
            <div class="form-outline mb-4">
              <label class="form-label" for="registerUsername"
>Логин</label>
              <input type="text" v-model="form.username"
id="registerUsername" class="form-control" name="userName" aria-
labelledby="Введите свое Имя"/>
            </div>

            <div class="form-outline mb-4">
              <label class="form-label"
for="registerPassword">Пароль</label>
              <input type="password" v-model="form.password"
id="registerPassword" class="form-control" name="password"/>
            </div>

            <div class="form-outline mb-4">
              <label class="form-label" for="registerEmail">Email</label>
              <input type="email" v-model="form.email" id="registerEmail"
class="form-control" name="email"/>
            </div>

            <div class="form-outline mb-4">
              <label class="form-label" for="registerName">Имя</label>
              <input type="text" v-model="form.firstname" id="registerName"
class="form-control" name="firstName"/>
            </div>

            <button type="submit" class="btn btn-my-main btn-block mb-3"
data-bs-dismiss="modal">Зарегистрироваться</button>
          </form>
        </div>
      </div>
    </div>
  </div>
</template>
```

```
<script>
  import {mapActions, mapState} from "pinia";
  import usersStore from "@/stores/user.js";

  export default {
    name: 'regModal',
    data() {
      return {
        form: {
          firstname: '',
          username: '',
          email: '',
          password: '',
          capCount: 0,
        },
        usState: usersStore(),
      };
    },
    computed: {
    },
    methods: {
      async registration() {
        const resp = (await this.usState.createUser(this.form));
        await this.formReset();
      },
      async formReset() {
        this.form.password = ''
        this.form.email = ''
        this.form.firstname = ''
        this.form.username = ''
      }
    },
  }
</script>

<style scoped>
.modal-my-config{
  --bs-modal-bg: var(--color3);
  --bs-modal-color: var(--color4);
  --bs-modal-border-color: var(--color1);
  --bs-modal-header-border-color: var(--color1);
  --bs-modal-footer-border-color: var(--color1);
}


</style>
```

Компонента содержит модальное окно для регистрации

```
<template>
  <div class="modal fade modal-my-config" id="loginFormModal" data-bs-
backdrop="static" data-bs-keyboard="false" tabindex="-1" aria-
labelledby="loginFormModal" aria-hidden="true">
    <div class="modal-dialog">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="staticBackdropLabel">Авторизация</h5>
          <button type="button" class="btn btn-my-main" data-bs-
dismiss="modal" aria-label="Закрыть">
            <svg id="close" class="close-btn"  viewBox="0 0 16 16">
              <path fill-rule="evenodd" d="M8 2a.5.5 0 0 1 .5.5v5h5a.5.5 0
0 1 0 1h-5v5a.5.5 0 0 1-1 0v-5h-5a.5.5 0 0 1 0-1h5v-5A.5.5 0 0 1 8 2"/>
            </svg>
          </button>
        </div>
        <div class="modal-body">
          <form id="LogInForm" @submit.prevent="login">
            <div class="form-outline mb-4">
              <label class="form-label" for="LogInFormEmail">Email</label>
              <input type="email" v-model="form.email" id="LogInFormEmail"
class="form-control" name="email"/>
            </div>

            <div class="form-outline mb-4">
              <label class="form-label"
for="LogInFormPassword">Пароль</label>
              <input type="password" v-model="form.password"
id="LogInFormPassword" class="form-control" name="password"/>
            </div>
            <button type="submit" class="btn btn-my-main btn-block mb-3"
id="LogInFormBTN" data-bs-dismiss="modal">Sign in</button>
          </form>
        </div>
      </div>
    </div>
  </div>
</template>

<script>
import {mapActions, mapGetters, mapState} from "pinia";
import usersStore from "@/stores/user.js";
  export default {
    name: 'loginModal',
    data() {
      return {
        form: {
          email: '',
          password: '',
        },
        usState: usersStore(),
      };
    },
    computed: {
    },
    methods: {
      async login() {
        const resp = (await this.usState.loginUser(this.form));
        await this.formReset();
      },
```

```
      async formReset() {
        this.form.password = '';
        this.form.email = '';
      }
    },
  }
</script>

<style scoped>
.modal-my-config{
  --bs-modal-bg: var(--color3);
  --bs-modal-color: var(--color4);
  --bs-modal-border-color: var(--color1);
  --bs-modal-header-border-color: var(--color1);
  --bs-modal-footer-border-color: var(--color1);
}

</style>
```

Компонента содержит модальное окно входа в аккаунт

5.4) Vault

Компоненты, отвечающие за создание, открытие, отображение карточек капсул времени на главной странице

```
<template>
  <div class="modal fade modal-my-config" id="CreateCapsuleFormModalId"
data-bs-backdrop="static" data-bs-keyboard="false" tabindex="-1" aria-
labelledby="#CreateCapsuleFormModalId" aria-hidden="true">
    <div class="modal-dialog">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="staticBackdropLabel">Создание
капсулы</h5>
          <button type="button" class="btn btn-my-main" data-bs-
dismiss="modal" aria-label="Закрыть">
            <svg id="close" class="close-btn"  viewBox="0 0 16 16">
              <path fill-rule="evenodd" d="M8 2a.5.5 0 0 1 .5.5v5h5a.5.5 0
0 1 0 1h-5v5a.5.5 0 0 1-1 0v-5h-5a.5.5 0 0 1 0-1h5v-5A.5.5 0 0 1 8 2"/>
            </svg>
          </button>
        </div>
        <div class="modal-body">
          <form id="CreateCapsuleForm" @submit.prevent="createVault">
            <div class="form-outline mb-4">
              <label class="form-label"
for="CreateCapsuleNameID">Название:</label>
              <input type="text" id="CreateCapsuleNameID" class="form-
control" name="name" v-model="form.name" required/>
            </div>

            <div class="form-outline mb-4">
              <label class="form-label"
for="CreateCapsuleEndDateBoxID">Дата и время открытия:</label>
              <input type="datetime-local" id="CreateCapsuleEndDateBoxID"
class="form-control" name="openDate" v-model="form.openDate" required/>
            </div>

            <div class="form-outline mb-4">
              <label class="form-label"
for="CreateCapsuleDescriptionID">Краткое описание капсулы:</label>
              <textarea type="text" id="CreateCapsuleDescriptionID"
class="form-control" name="description" v-model="form.description"
required></textarea>
            </div>

            <div class="form-outline mb-4">
              <label class="form-check-label" for="RadioBtnMy"> Сделать
капсулу общедоступной: </label>
              <input class="form-check-input" type="checkbox"
id="capsuleAccessPoint" name="access" v-model="form.access" />
            </div>

            <hr class="content-dividing-line rounded mt-3 mb-3"/>
            <div class="form-outline mb-10">
              <p class="fw-bold">Содержимое капсулы</p>
            </div>

            <div class="form-outline mb-4">
```

```
              <label class="form-label"
for="CreateCapsuleTextID">Запечатанный текст</label>
              <textarea type="text" id="CreateCapsuleTextID" class="form-
control" name="text" v-model="form.text"></textarea>
            </div>

            <div className="form-outline mb-4">
              <label className="form-label"
htmlFor="CreateCapsuleFiles">Прикрепленные файлы</label>
              <input type="file" id="CreateCapsuleFiles" className="form-
control" multiple name="files" v-on="form.files"/>
            </div>


            <button type="submit" class="btn btn-my-main btn-block mb-3"
data-bs-dismiss="modal">
              Запечатать
            </button>
          </form>
        </div>
      </div>
    </div>
  </div>
</template>
//
<style scoped>
.modal-my-config{
  --bs-modal-bg: var(--color3);
  --bs-modal-color: var(--color4);
  --bs-modal-border-color: var(--color1);
  --bs-modal-header-border-color: var(--color1);
  --bs-modal-footer-border-color: var(--color1);
}
</style>

<script>



import {mapActions, mapState} from "pinia";
import usersStore from "@/stores/user.js";
import capsulesStore from "@/stores/capsules.js";
import timeVault from "@/views/TimeVault.vue";
import Icon from "@/components/icons/baseIcon.vue";

export default {
  name: 'VaultCreationModal',
  components: {Icon},
  data() {
    return {
      form: {
        name: '',
        openDate: '',
        description: '',
        access: 'false',
        text: '',
        files: '',
        userId: '',
        userName: '',
      },
      usState: usersStore(),
```

```
      capsuleState: capsulesStore(),
    };
  },
  computed: {
  },
  methods: {
    async createVault() {
      await this.getUser();
      const nData = JSON.parse(this.usState.user)
      nData['capCount'] += 1;
      this.usState.user = JSON.stringify(nData)
      //await this.usState.updateUser(oData);
      this.form.openDate = new Date(this.form.openDate).getTime();
      const resp = (await this.capsuleState.createCapsule(this.form,
JSON.parse(JSON.parse(localStorage.pinia_users)['token'])));
      await this.capsuleState.loadCapsules();
      await this.formReset();
    },
    async getUser() {
      const user = JSON.parse(this.usState.user)
      if (user) {
        this.form.userName = user['username'];
        this.form.userId = user['id'];
      }
    },
    async formReset() {
      this.form.name = '';
      this.form.openDate = '';
      this.form.description = '';
      this.form.access = false;
      this.form.text = '';
      this.form.files = '';
      this.form.userId = '';
      this.form.userName = '';
    },
    mounted() {
      this.loadCapsules();
    },
  },
}
</script>
```

Компонента содержит модальное окно создания капсул

```
<template>
  <div class="modal fade modal-my-config-light" id="openCapsuleFormModalId"
data-bs-backdrop="static" data-bs-keyboard="false" tabindex="-1" aria-
labelledby="openCapsuleFormModal" aria-hidden="true">
    <div class="modal-dialog">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title"
id="modalCapsuleOpenedTitle">{{this.capState.capsule.name}}</h5>
          <button type="button" class="btn btn-my-main" data-bs-
dismiss="modal" aria-label="Закрыть">
            <svg class="close-btn">
              <svg id="close" class="close-btn"  viewBox="0 0 16 16">
                <path fill-rule="evenodd" d="M8 2a.5.5 0 0 1 .5.5v5h5a.5.5
0 0 1 0 1h-5v5a.5.5 0 0 1-1 0v-5h-5a.5.5 0 0 1 0-1h5v-5A.5.5 0 0 1 8 2"/>
              </svg>
            </svg>
          </button>
        </div>
        <div class="modal-body" id="modalCapsuleOpenedBody">
          <div class="mb-4">
            <p class="fw-bold">Описание:</p>
            <div>{{ this.capState.capsule.description }}</div>
          </div>

          <hr class="content-dividing-line rounded mt-3"/>
          <div class="form-outline mb-10">
            <p class="fw-bold mb-5">Содержимое капсулы</p>
          </div>

          <div class="mb-4">
            <p class="fw-bold">Послание:</p>
            <div v-for="str in this.capState.capsule.text.split('\n')">
              <div> {{ str }} </div>
            </div>
          </div>

        </div>
      </div>
    </div>
  </div>
</template>

<style scoped>
.modal-my-config-light{
  --bs-modal-bg: var(--color1);
  --bs-modal-color: var(--color4);
  --bs-modal-border-color: var(--color3);
  --bs-modal-header-border-color: var(--color3);
  --bs-modal-footer-border-color: var(--color3);
}
</style>

<script>
import capsulesStore from "@/stores/capsules.js";
import BaseIcon from "@/components/icons/baseIcon.vue";
import Icon from "@/components/icons/baseIcon.vue";


export default {
  name: 'openVaultModal',
```

```
  components: {Icon, BaseIcon},
  data(){
    return{
      capState: capsulesStore(),
    }
  },
}
</script>
```

Компонента содержит модальное окно для отображения открытой капсулы

```
<template>
  <div class="card capsule-param">
    <div class="card-header">
      <img src="/src/assets/img/capPattern.jpg" alt="..." class="card-img-
top">
    </div>
    <div class="card-body">
      <h5 class="card-title">{{ name }}</h5>
      <p class="card-text mb-2"> {{ description }}</p>
      <form @submit.prevent="getCapsule" class="justify-f-e">
        <button type="submit" class="btn btn-my-card btn-block btn-
sm">Открыть</button>
      </form>
    </div>
    <div class="card-footer justify-s-b">
      <p>{{ getNormalDate() }}</p>
      <p>@{{ creator }}</p>
    </div>
  </div>
</template>

<style scoped>
  .btn-my-card{
    --bs-btn-color: var(--color4);
    --bs-btn-bg: var(--color2);
    --bs-btn-border-color: var(--color3);
    --bs-btn-hover-color: var(--color4);
    --bs-btn-hover-bg: var(--color11);
    --bs-btn-hover-border-color: var(--color1);
    --bs-btn-active-color: var(--color4);
    --bs-btn-active-bg: var(--color11);
    --bs-btn-active-border-color: var(--color1);
  }
</style>

<script>

  import OpenVaultModal from
"@/components/Main/Vaults/openCapsuleModal.vue";
  import capsulesStore from "@/stores/capsules.js";
  import {Modal} from "bootstrap";
  import {useDate} from "@/composables/date.js";
  export default {
    name: 'vaultOpened',
    data(){
      return{
        capState: capsulesStore(),
      }
    },
    components: {OpenVaultModal},
    props: {
      id: {
        type: Number,
      },
      name: {
        type: String,
      },
      description: {
        type: String,
      },
```

```
      openDate: {
        type: Date,
      },
      creator: {
        type: String,
      },
    },
    methods: {
      getCapsule(){
        this.capState.loadOneCapsule(this.id);
        let vaultModal = new
Modal(document.getElementById('openCapsuleFormModalId'));
        //console.log(JSON.stringify(this.capState.capsule))
        vaultModal.show();
      },
      getNormalDate(){
        return useDate(this.openDate)
      }
    }

  }
</script>
```

Компонента содержит карточку открытой капсулы

```
<template>
  <div class="card capsule-param">
    <div class="card-header">
      <img src="/src/assets/img/capPattern.jpg" alt="..." class="card-img-
top">
    </div>
    <div class="card-body">
      <h5 class="card-title">{{ name }}</h5>
      <p class="card-text mb-2"> {{ description }}</p>
    </div>
    <div class="card-footer justify-s-b">
      <p>{{ getNormalDate() }}</p>
      <p>@{{ creator }}</p>
    </div>
  </div>
</template>

<style scoped>
</style>

<script>
  import {useDate} from "@/composables/date.js";

  export default {
    name: 'vaultClosed',
    props: {
      id: {
        type: Number,
      },
      name: {
        type: String,
      },
      description: {
        type: String,
      },
      openDate: {
        type: Date,
      },
      creator: {
        type: String,
      },
    },
    methods:{
      getNormalDate(){
        return useDate(this.openDate)
      }
    }

  }
</script>
```

Компонента содержит карточку закрытой капсулы

6) Правки

В комонентах для отображения карточек капсул использовалась дублирующаяся функция для получения даты-время в удобоворимом формате

6.1) Composable

```
getNormalDate(){
    const date = new Date(this.openDate)
    const dd = date.getDate();
    const mm = date.getMonth() + 1
    const yyyy = date.getFullYear();
    let hh = date.getHours();
    let min = date.getMinutes();
    if(min < 10){
        min = '0'+min;
    }
    const strDate = dd + '.' + mm + '.' + yyyy + '  ' + hh + ':' + min;
    //console.log(strDate)
    return strDate;
}
```

Переместим эту функцию в отдельный файл – **date.js**

```
export function useDate(openDate){
    const date = new Date(openDate)
    const dd = date.getDate();
    const mm = date.getMonth() + 1
    const yyyy = date.getFullYear();
    let hh = date.getHours();
    let min = date.getMinutes();
    if(min < 10){
        min = '0'+min;
    }
    const strDate = dd + '.' + mm + '.' + yyyy + '  ' + hh + ':' + min;
    //console.log(strDate)
    return strDate;
}
```

В компонентах эту функцию используем следующим образом

```
<template>
   <p>{{ getNormalDate() }}</p>
</template>

import {useDate} from "@/composables/date.js";

  export default {
    methods: {
      getNormalDate(){
        return useDate(this.openDate)
      }
    }
  }
</script>
```