САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет инфокоммуникационных технологий

Образовательная программа 09.03.03

Направление подготовки (специальность) Мобильные сетевые технологии

О Т Ч Е Т

по курсовой работе

Тема задания: разработка одностраничного веб-приложения (SPA) с использованием фреймворка Vue.js

Обучающийся: Иконенко Даинл, К33402

Руководитель: Добряков Д. И., преподаватель

Оценка за курсовую работу ____

Дата _____

Санкт-Петербург
2022

ВВЕДЕНИЕ

Актуальность

Сайты бронирования – это большое подспорье для путешественников. Среди весомых плюсов таких порталов:

— Огромное разнообразие вариантов жилья

Турист может подобрать для себя тысячи вариантов: отели, гостевые дома, хостелы, апартаменты, виллы или даже экзотические типы жилья (например, дома на лодках и прочее).

— экономия времени

Туристу действительно не придется тратить время на поход в турфирму и общение с менеджером, на которое может уйти более часа. Бронирование апартаментов в интернете – простая и быстрая процедура, занимающая всего 5-15 минут: необходимо лишь выбрать пункт назначения, даты заезда-выезда и категорию «звездности» жилья.

— Экономия денег

Турист может выбрать любое жилье – квартиру, хостел, гостиницу, отель – под любые финансовые возможности. А порой сайты предлагают жилье со скидкой 50% и различные выгодные бонусные программы.

— Независимость при принятии решения

Туристу никто не навязывает мнение, поэтому он абсолютно самостоятелен при выборе подходящего для себя варианта жилья.

— Бесплатная отмена

Если у туриста по какой-то причине отменяется поездка, или он неожиданно нашел более привлекательный для себя вариант жилья, то он в любой момент через портал может отменить бронирование, не оплачивая штрафные санкции.

Использование Vue.js для разработки одностраничного приложения оправдано в первую очередь тем, что одностраничные приложения, как правило, быстрее загружают страницы сайта, что ведёт к лучшему восприятию пользователем процесса взаимодействия с ним.

Цели и задачи

1. Определение средств разработки
2. Определение функциональных требований
3. Проектирование и реализация фронтенда
4. Интеграция Vuex и миксинов

ГЛАВА 1. СРЕДСТВА РАЗРАБОТКИ И ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

1.1 Средства разработки

Для разработки фронтенда был использован фреймворк Vue.js, так как его изучению была посвящена часть курса “Фронд-энд разработка”. Также для специализированного функционала использовались библиотеки Axios, BootstrapVue, Vue router

1.2 Функциональные требования

1. Набрать минимально осмысленное число функциональных страниц, покрывающее следующий функционал: логин, регистрация, сброс пароля, редактирование данных профиля, страница с поиском/фильтрацией по важным сущностям системы.
2. Желательно использовать миксины.
3. Желательно использовать Vuex.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ФРОНТЕНДА

1.1 Проектирование фронтенда

Основной сущностью веб-приложения по выбранной мной тематике является отель. Действия в приложении могут совершать зарегистрированные пользователи, поэтому нужно предусмотреть процессы регистрации и авторизации. У пользователя должен быть личный кабинет. Кроме того, для связи пользователя и отеля существует сущность бронирование, с помощью которой осуществляется главный функционал приложения.

Исходя из этого, был определён следующий список страниц, необходимый к реализации:

1. Регистрация
2. Вход
3. Личный кабинет
4. Изменение информации в ЛК
5. Смена пароля
6. Главная страница/страница поиска по отелям
7. Страница с результатами поиска
8. Мои бронирования

1.2 Реализация фронтенда

На каждой странице приложения должна присутствовать шапка проекта с навигацией. Навигацию между страницами удобно осуществлять с помощью библиотеки Vue Router.

Содержание файла router/index.js, который определяет доступные в приложении маршруты(без импортов):

```
Vue.use(VueRouter)

const routes = [
  {
    path: '/login',
    name: 'Login',
    component: LoginMain
  },
  {
    path: '/Booking',
    name: 'MyBooking',
    component: MyBooking
  },
  {
    path: '/register',
    name: 'Register',
    component: RegisterMain
  },
  {
    path: '/',
    name: 'Search',
    component: Search
  },
  {
    path: '/results',
    name: 'Results',
    component: SearchResultsPage
  },
  {
    path: '/my-profile',
    name: 'MyProfile',
    component: MyProfile
  },
  {
    path: '/my-profile/edit',
    name: 'EditProfile',
    component: EditProfile
  },
  {
    path: '/my-profile/change-password',
    name: 'ChangePassword',
    component: ChangePassword
  },

]

const router = new VueRouter({
  mode: 'history',
  routes
})


export default router
```

Для простого и быстрого создания адаптивной верстки страницы можно применить библиотеку Bootstrap Vue. Рассмотрим это на примере страницы моего профиля:

```vue
<template>
  <b-container class="my-profile">
    <b-row class="justify-content-center"><h2>Ваш профиль</h2></b-row>
    <b-row class="justify-content-center">
      <div class="col-10">
        <p>
          <b-link :to="{ name: 'EditProfile' }"
          ><em>Редактировать профиль</em></b-link
          >
        </p>
        <p><strong>Email: </strong> {{ profile.email }}</p>
        <p><strong>Имя: </strong> {{ profile.first_name }}</p>
        <p><strong>Фамилия: </strong> {{ profile.last_name }}</p>
        <p><strong>Отчество: </strong> {{ profile.middle_name }}</p>
        <p><strong>Дата рождения: </strong> {{ profile.birthdate }}</p>
        <br />
        <b-link :to="{ name: 'ChangePassword' }"
        ><em>Поменять пароль для входа</em></b-link
        >
      </div>
    </b-row>
  </b-container>
</template>
<script>
export default {
  name: "MyProfileMain",
  props: {
    profile: Object
  }
}
</script>
<style>
.my-profile {
  padding: 20px;
  border: 2px solid;
  margin: 30px auto 0px;
  border-radius: 7px;
}
</style>
```

Соберём представление моего профиля из компонент headers.vue и MyProfileMain.vue

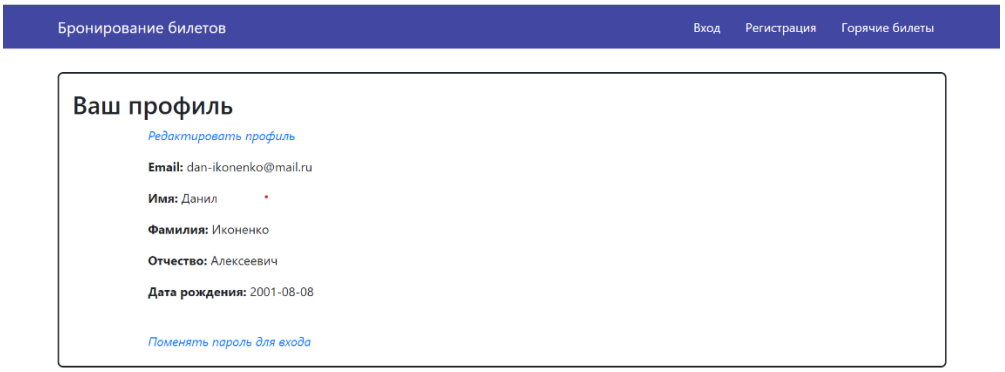
```
<template>
  <section class="page-content">
    <headers-main />
    <my-profile-main :profile="this.profile" />
  </section>
</template>

<script>
import headersMain from "../components/headers"
import MyProfileMain from "../components/MyProfileMain.vue"

export default {
  name: "MyProfile",
  components: {
    headersMain,
    MyProfileMain
  },
  beforeMount() {
    this.getMyProfile()
  },
  data: () => ({
    profile: Object
  })
}
</script>

<style>
</style>
```

Скриншот получившейся страницы:



Аналогично были реализованы остальные страницы.

ЗАКЛЮЧЕНИЕ

Выводы по работе

В ходе выполнения этой работы я получил практический опыт использования Vue.js для создания Single Page Application. Также я разобрался, как для реализации своего приложения применить популярные библиотеки: Vue Router для навигации, Axios для отправки запросов на бэкенд, Bootstrap Vue для создания адаптивной верстки, Vuex для хранения состояния авторизации. Кроме того, я стал использовать больше инструментов Vue.js: директивы v-if/v-else, миксины.

СПИСОК ЛИТЕРАТУРЫ

1. Документация Vue.js [Электронный ресурс] https://vuejs.org/v2/guide/
2. Документация Vue Router [Электронный ресурс] https://router.vuejs.org/guide/
3. Документация Vuex [Электронный ресурс] https://vuex.vuejs.org/ru/guide/
4. Документация BootstrapVue [Электронный ресурс] https://bootstrap-vue.org/docs
5. Документация Axios [Электронный ресурс] https://axios-http.com/docs/intro