

**Министерство образования и науки Российской Федерации**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО**  
**ОБРАЗОВАНИЯ**

**УНИВЕРСИТЕТ ИТМО**

Факультет инфокоммуникационных технологий

Образовательная программа 09.03.03

Направление подготовки (специальность) Мобильные сетевые технологии

**О Т Ч Е Т**

о курсовой работе

Тема задания: Разработка клиентской части сервиса прогноза погоды средствами фреймворка Vue.JS

Обучающийся Прохоров Николай Игоревич, К33402

Руководитель: Добряков Д. И., преподаватель

Оценка за курсовую работу: \_\_\_\_

Подписи членов комиссии:

\_\_\_\_\_  
(подпись)

\_\_\_\_\_  
(подпись)

\_\_\_\_\_  
(подпись)

Дата 26.01.2022

Санкт-Петербург  
2022

Актуальность	3
Цели и задачи	3
ГЛАВА 1. СРЕДСТВА РАЗРАБОТКИ И ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ	4
1.1 Средства разработки	4
1.2 Функциональные требования	4
1.3 Страницы сервиса	4
ГЛАВА 2. Функционал сервиса	6
2.1. Визуальный интерфейс	6
2.2. Код	13
Выводы по работе	14
СПИСОК ЛИТЕРАТУРЫ	18

# **ВВЕДЕНИЕ**

## **Актуальность**

Среди предложенных вариантов для работ мною был выбран сервис для получения прогноза погоды. Меня наиболее привлекла открытость доступных бесплатных API, а также обширность данных. Чем больше данных – чем больше пространство для творчества, нет лимитов по количеству запросов в сутки, нет жестких требований, да и в целом разработка такого сервиса станет отличным дополнением к моему портфолио. Ровно поэтому мною и была выбрана данная тема.

Итак, целью проекта стала разработка клиентской части сервиса прогноза погоды средствами фреймворка Vue.JS с необычным дизайном в стиле Нео-морфизма.

## **Цели и задачи**

1. Определение функциональных требований
2. Доступная и адаптивная верстка сайта
3. Создание логики для запросов к OpenWeatherMap API
4. Подключение собственного Backend на Django
5. Созданий функциональной части на Vue.js, Axios и Vuex

# ГЛАВА 1. СРЕДСТВА РАЗРАБОТКИ И ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

## 1.1 Средства разработки

Vue.js — JavaScript-фреймворк с открытым исходным кодом для создания пользовательских интерфейсов. Легко интегрируется в проекты с использованием других JavaScript-библиотек. Может функционировать как веб-фреймворк для разработки одностраничных приложений в реактивном стиле. В работе я использую именно этот фреймворк, поскольку изучаю его уже длительное время, и среди своих конкурентов в виде React или Angular он привлекает меня больше всего.

На Vue есть большой спрос среди работодателей, популярность фреймворка стремительно растет в последние годы. Vue – отличный выбор как для начинающих разработчиков, так и для весьма опытных.

Для улучшения качества кода я так же использую Typescript во всем проекте, поскольку строгая типизация позволяет сократить количество потенциальных ошибок, а так же Nuxt.js – удобный и гибкий фреймворк, предоставляющий множество возможностей поверх Vue

## 1.2 Функциональные требования

1. Vue.js версии 2.
2. Nuxt.js версии 2
3. Адаптивная верстка
4. Библиотека компонентов Bootstrap-Vue
5. Запросы с помощью axios

### **1.3 Страницы сервиса**

В финальной версии сервиса для получения прогноза погоды в разных городах реализовано 8 страниц:

1. Главная страница - лендинг
2. Регистрация
3. Авторизация
4. Поиск прогноза по городу
5. Избранное с быстрым доступом к добавленным городам
6. Страница с информацией профиля
7. Страница смены имени пользователя
8. Страница смены почтового ящика
9. Страница смены пароля

## ГЛАВА 2. Функционал сервиса

### 2.1. Визуальный интерфейс

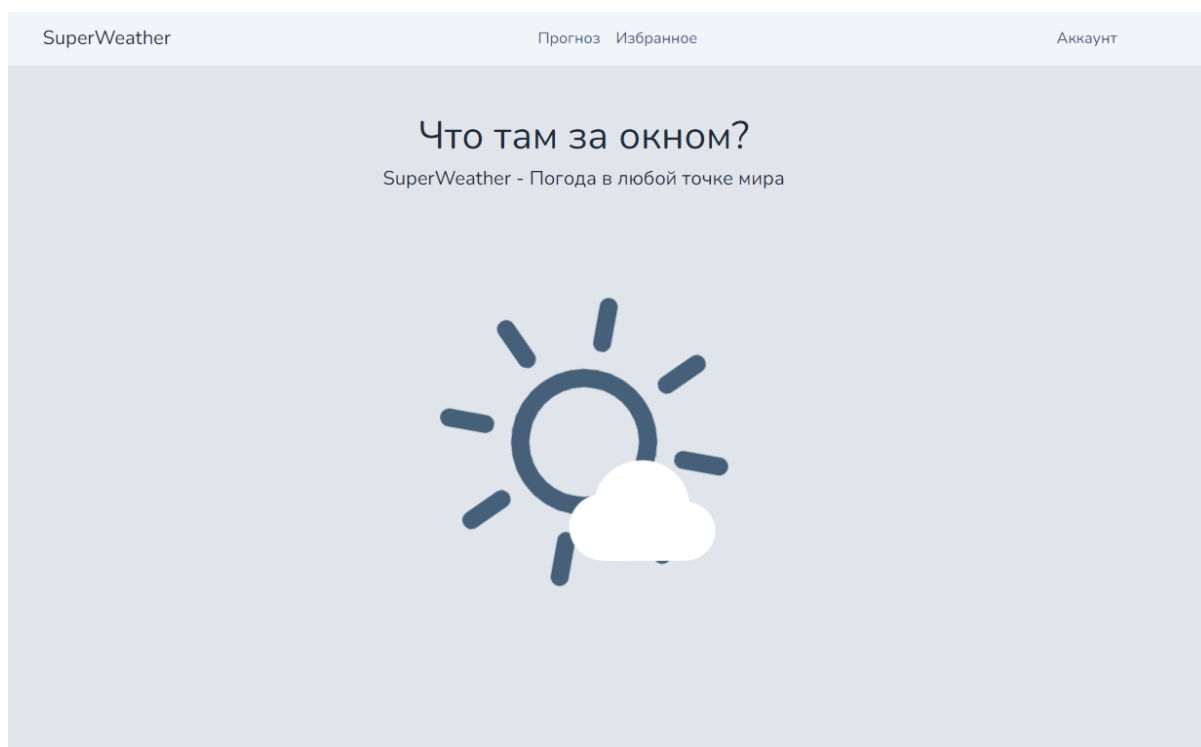


Рисунок 1. Главная страница, лендинг проекта

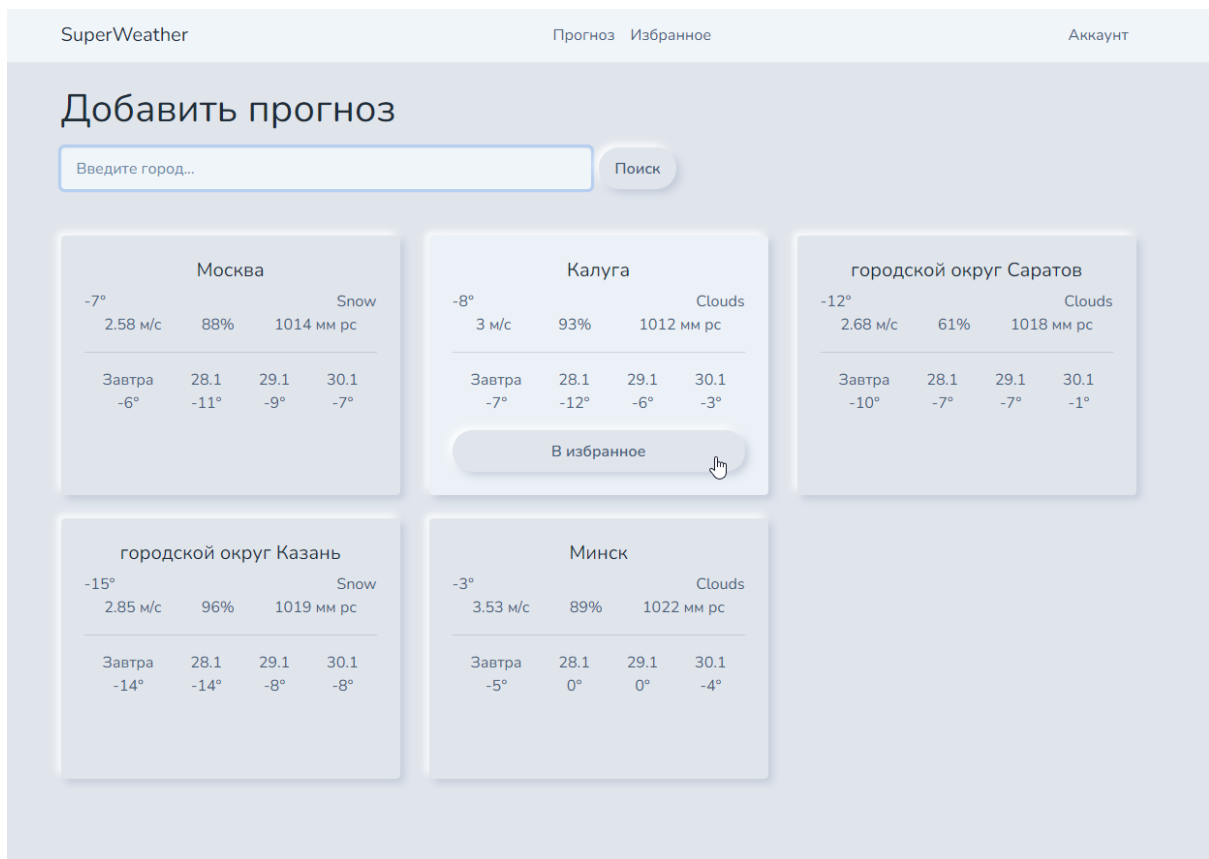


Рисунок 2. Страница поиска прогноза погоды по городу

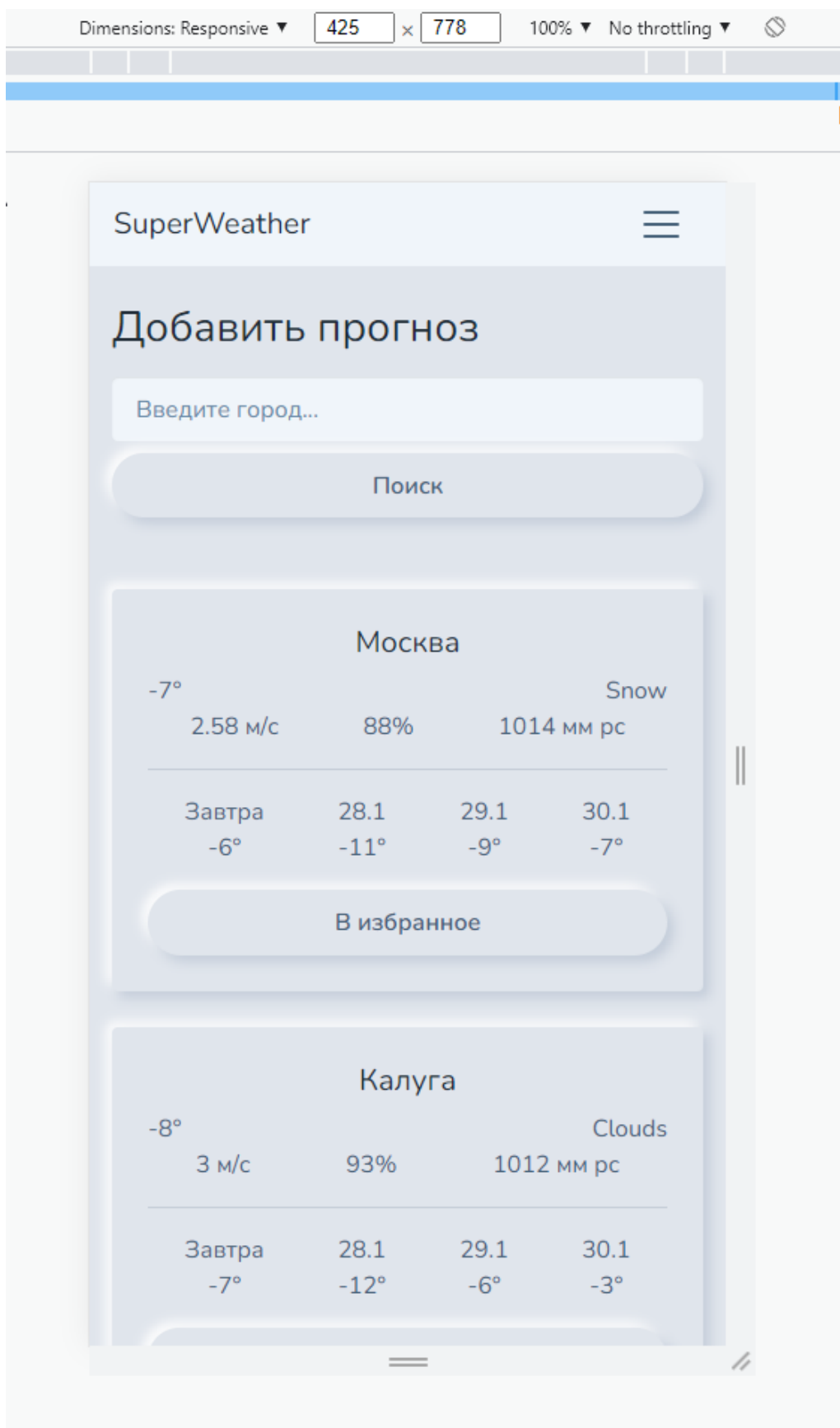


Рисунок 2. Страница поиска прогноза погоды по городу, вид с мобильного устройства.  
Все страницы на сайте адаптированы под любые экраны.





Рисунок 3. Избранные города с текущей погодой и прогнозом на ближайшее время

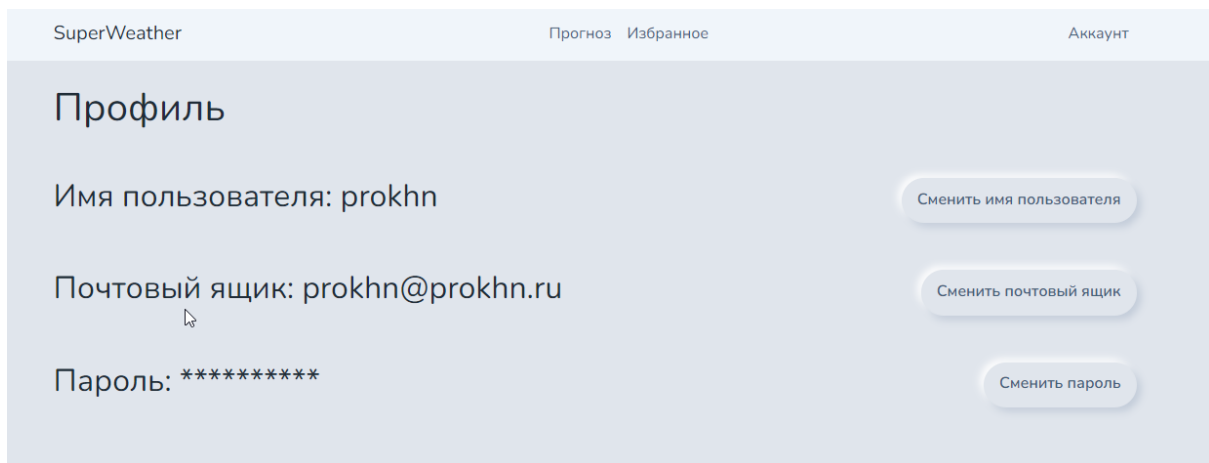


Рисунок 4. Данные профиля

SuperWeather

Прогноз Избранное Аккаунт

Новое имя пользователя

prokhn

Пользователь с таким именем уже существует.

Пароль для подтверждения

•

Сменить

Рисунок 5. Форма изменения имени пользователя, пример отображения сообщений валидаций бекенда

SuperWeather

Прогноз Избранное Аккаунт

Новый почтовый ящик

prokhn@prokhn.ru

Сменить

Рисунок 6. Форма изменения почтового ящика пользователя.

The screenshot shows the 'SuperWeather' application interface. At the top, there is a navigation bar with 'Прогноз' and 'Избранное' in the center, and 'Аккаунт' on the right. The main content area is a light blue gradient. In the center, there is a white rounded rectangle containing the password change form. The form has two input fields: 'Ваш текущий пароль' (Your current password) and 'Новый пароль' (New password). The first field has a red error message below it: 'Неправильный пароль.' (Incorrect password.). The second field has a red error message below it: 'Введённый пароль слишком короткий. Он должен содержать как минимум 8 символов., Введённый пароль слишком широко распространён., Введённый пароль состоит только из цифр.' (The entered password is too short. It must contain at least 8 characters., The entered password is too common., The entered password consists only of digits.). At the bottom of the form is a button labeled 'Сменить' (Change).

Рисунок 7. Форма изменения пароля, пример отображения сообщений валидаций бекенда

The screenshot shows the 'SuperWeather' application interface. At the top, there is a navigation bar with 'Прогноз' in the center and 'Войти' on the right. The main content area is a light blue gradient. In the center, there is a white rounded rectangle containing the registration form. The form has three input fields: 'Имя пользователя' (Username) with placeholder text 'Придумайте имя пользователя', 'Пароль' (Password) with placeholder text 'Придумайте пароль', and 'Повторите пароль' (Repeat password) with placeholder text 'Повторите пароль'. Below the password field is a list of four bullet points: 'Пароль не должен быть слишком похож на другую вашу личную информацию.', 'Ваш пароль должен содержать как минимум 8 символов.', 'Такой пароль часто используется.', and 'Пароль не может состоять только из цифр.'. At the bottom of the form is a button labeled 'Зарегистрироваться' (Register) and a link labeled 'Есть аккаунт? Войти' (Have an account? Log in).

Рисунок 8. Форма регистрации пользователя

SuperWeather

Прогноз

Войти

Невозможно войти с предоставленными учетными данными.

Имя пользователя

prokhn

Имя пользователя

...

Не помню пароль

Войти

Нет аккаунта? Зарегистрироваться

Рисунок 9. Форма авторизации пользователя после неверного ввода пароля

## 2.2. Код

```
cities.ts
1  import ...
7
8  @Module( options: {
9    name: 'cities',
10   stateFactory: true,
11   namespaces: true,
12 })
13 export default class CitiesModule extends VuexModule {
14   data: TCitiesStoreData[] = []
15   favorites: number[] = []
16   search: number[] = []
17   error: string = ''
18
19   get isCityInFavorites() {...}
22
23   get searchedCitiesOnly() {...}
26
27   @Mutation
28   setError(error: string) {...}
31
32   @Mutation
33   addData(data: TCitiesStoreData) {...}
36
37   @Mutation
38   setData(data: TCitiesStoreData[]) {...}
41
42   @Mutation
43   addCityFavorite(cityID: number) {...}
46
47   @Mutation
48   removeCityFavorite(cityID: number) {...}
51
52   @Mutation
53   addCitySearch(cityID: number) {...}
56
57   @VuexAction( params: { rawError: true })
58   async fetchCity(city: string | null) {...}
100
101   @VuexAction( params: { rawError: true })
102   async favoritesChange(cityID: number) {...}
115 }
```

Рисунок 10. Модуль Vuex на Typescript для хранения городов и избранных городов

```

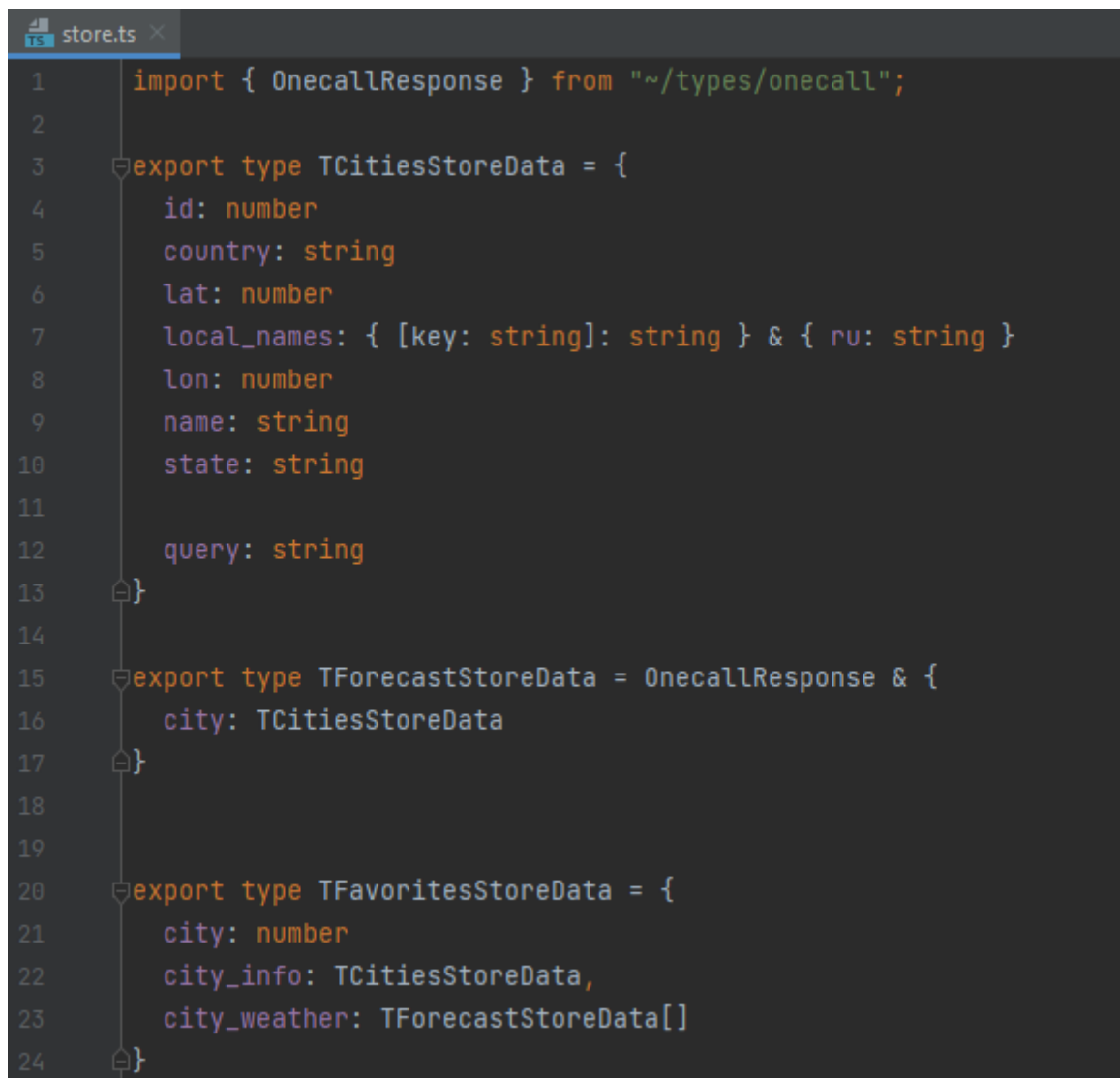
1 import { Component, Vue } from "nuxt-property-decorator";
2 import { TFormErrors } from "~/types/forms";
3
4 @Component
5 export default class FormsMixin extends Vue {
6   errorsFromForm<T>(form: T) {
7     let errors: TFormErrors<T> = {}
8     const keys = Object.keys(form) as (keyof TFormErrors<T>)[]
9     keys.forEach(key => {
10       // @ts-ignore
11       errors[key] = new Array<string>()
12     })
13
14     errors.non_field_errors = []
15
16     return errors
17   }
18
19   errorsFromResponse<T = any>(errors: TFormErrors<T>, data?: TFormErrors<T>) {
20     if (!data) return
21
22     const keys = Object.keys(data) as (keyof TFormErrors<T>)[]
23     keys.forEach(key => {
24       errors[key] = data[key]
25     })
26
27     return errors
28   }
29
30   errorsState<T = any>(error: TFormErrors<T>, key: keyof TFormErrors<T>) {
31     return error[key] && (error[key] as string[]).length !== 0 ? false : null
32   }
33
34   errorsText<T = any>(error: TFormErrors<T>, key: keyof TFormErrors<T>) {
35     return error[key] && (error[key] as string[]).length !== 0 ? (error[key] as string[]).join(', ') : ''
36   }
37
38   errorReset<T = any>(error: TFormErrors<T>, key: keyof TFormErrors<T>) {
39     (error[key] as string[]) = new Array<string>()
40   }
41 }

```

Рисунок 11. Миксин для обработки серверных ошибок в формах с использованием Generic функций

```
login.vue x
1 <template>
2 <b-container class="d-flex justify-content-center align-items-start">
3   <base-card-form
4     :alert="errors"
5     @submit="submit"
6   >
7     <app-input
8       v-model="form.username"
9       label="Имя пользователя"
10      placeholder="Ваше имя пользователя"
11      :errors.sync="errors"
12      :errors-key="'username'"
13      required
14    />
15
16    <app-input
17      v-model="form.password"
18      label="Имя пользователя"
19      placeholder="Ваш пароль"
20      type="password"
21      :errors.sync="errors"
22      :errors-key="'password'"
23      required
24      class="mb-1"
25    />
26
27    <small class="pl-2">
28      <nuxt-link to="/">Не помню пароль</nuxt-link>
29    </small>
30
31    <b-button variant="primary"
32      type="submit"
33      block
34      class="mt-4"
35    >
36      Войти
37    </b-button>
38  </b-container>
</template>
```

Рисунок 12. HTML часть формы авторизации



```
1  import { OnecallResponse } from "~/types/onecall";
2
3  export type TCitiesStoreData = {
4    id: number
5    country: string
6    lat: number
7    local_names: { [key: string]: string } & { ru: string }
8    lon: number
9    name: string
10   state: string
11
12   query: string
13 }
14
15 export type TForecastStoreData = OnecallResponse & {
16   city: TCitiesStoreData
17 }
18
19
20 export type TFavoritesStoreData = {
21   city: number
22   city_info: TCitiesStoreData,
23   city_weather: TForecastStoreData[]
24 }
```

Рисунок 13. Типы Vuex хранилища



```

1  <template>
2    <b-form-group
3      :label="label"
4      :state="errorsState(errors, errorsKey)"
5      :invalid-feedback="errorsText(errors, errorsKey)"
6    >
7      <b-form-input...>
13
14      <slot />
15    </b-form-group>
16  </template>
17
18  <script lang="ts">
19    import ...
22
23    @Component( options: {
24      name: 'AppInput'
25    })
26    export default class AppInput extends mixins(FormsMixin) {
27      @VModel( options: { required: true })
28      vmodel !: string | number | null
29
30      @Prop() readonly label ? : string
31      @Prop() readonly placeholder ? : string
32
33      @Prop( options: { type: String, default: 'text' }) readonly type !: string
34      @Prop( options: { type: Boolean, default: false }) readonly required !: boolean
35
36      @Prop( options: {
37        default: () => {
38          return {}
39        }
40      }) readonly errors !: TFormErrors<any>
41      @Prop( options: { type: String, default: 'fieldError' }) readonly errorsKey !: string
42
43      @Watch( path: 'vmodel')
44      onChange () {
45        this.errorReset(this.errors, this.errorsKey)
46      }
47    }
48  </script>
49

```

Рисунок 14. Компонент input поля

## **Выводы по работе**

Был создан готовый к эксплуатации сервис по получению текущей погоды и прогнозов на ближайшее время с фильтрацией по городу, добавлением городов в избранное для быстрого доступа и профиль с возможностью смены любых полей.

За проект мне удалось улучшить свои знания во Vue.js и Nuxt.js, я поработал с Generic функциями и сложными типами, сумел написать удобный типизированный переиспользуемый миксин для работы с формами и ошибками. Цель курсовой работы выполнена

## СПИСОК ЛИТЕРАТУРЫ

1. Документация Vue.js [Электронный ресурс] — <https://ru.vuejs.org/v2/guide/>  
Дата обращения 24.01.2022.
2. Документация Nuxt.js [Электронный ресурс] — <https://nuxtjs.org/docs>  
Дата обращения 24.01.2022.
3. Документация Bootstrap-Vue [Электронный ресурс] — <https://bootstrap-vue.org/docs>  
Дата обращения 24.01.2022.
4. Документация API OpenWeather Map [Электронный ресурс] — <https://openweathermap.org/api>.  
Дата обращения 24.01.2022.