

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Домашняя работа №5

Выполнила:

Вилис Зоя

Группа К3344

**Проверил:
Добряков Д. И.**

Санкт-Петербург

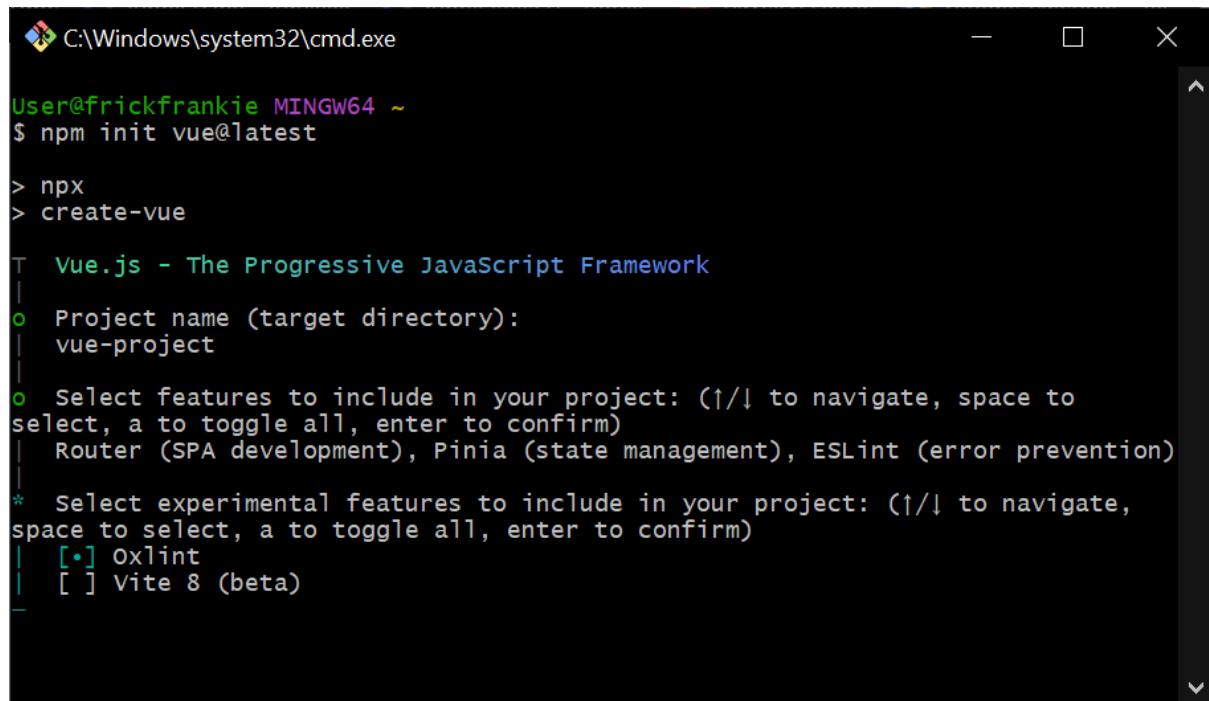
2026 г.

Задача

В рамках данной работы Вам предстоит изучить основные команды пакетного менеджера NPM и научиться стартовать проект на Vue. Научиться работать с npm и vue на основе мануала.

Ход работы

После установки vue и vue-cli инициализируем проект и выполняем действия согласно мануалу:



The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32\cmd.exe'. The user is running the command '\$ npm init vue@latest'. The output shows the creation of a new Vue.js project named 'vue-project'. It then prompts the user to select features to include in the project, listing 'Router (SPA development)', 'Pinia (state management)', 'ESLint (error prevention)', 'Oxlint', and 'Vite 8 (beta)'. The 'Oxlint' option is currently selected, indicated by a checked checkbox icon.

```
C:\Windows\system32\cmd.exe
User@frickfrankie MINGW64 ~
$ npm init vue@latest

> npx
> create-vue

T  Vue.js - The Progressive JavaScript Framework
o Project name (target directory):
  vue-project

o Select features to include in your project: (↑/↓ to navigate, space to
select, a to toggle all, enter to confirm)
  Router (SPA development), Pinia (state management), ESLint (error prevention)

*  Select experimental features to include in your project: (↑/↓ to navigate,
space to select, a to toggle all, enter to confirm)
  [•] Oxlint
  [ ] Vite 8 (beta)
```

Рисунок 1 — инициализация проекта

```
MINGW64:/c/Users/fW flUu/vue-project
cd vue-project
npm install
npm run dev

| Optional: Initialize Git in your project directory with:
git init && git add -A && git commit -m "initial commit"

User@frickfrankie MINGW64 ~
$ cd vue-project

User@frickfrankie MINGW64 ~/vue-project
$ npm install

added 228 packages, and audited 229 packages in 28s

57 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

User@frickfrankie MINGW64 ~/vue-project
$ |
```

Рисунок 2 — переход в директорию

```
"scripts": {
  "start": "vite --port 8080",
  "dev": "vite",
  "build": "vite build",
  "preview": "vite preview",
  "lint": "eslint . --ext .vue,.js,.jsx,.cjs,.mjs --fix --ignore-path .gitignore"
},
```

Рисунок 3 — редактирование файла package.json

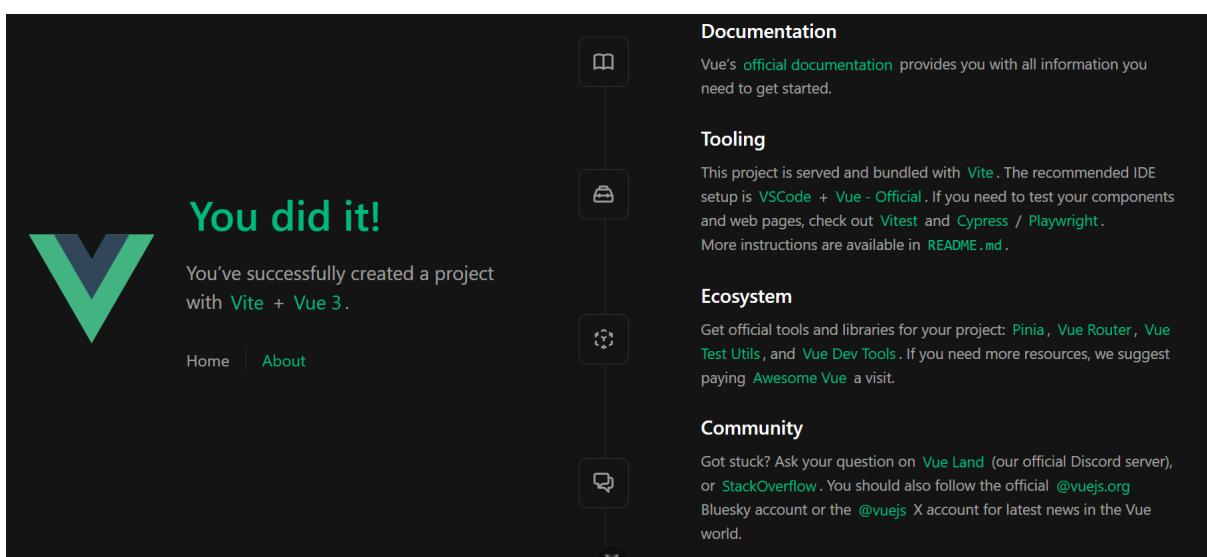


Рисунок 4 — окно после запуска проекта

The screenshot shows a code editor with three tabs: package.json, instance.js (selected), and notes.js. The instance.js tab contains the following code:

```
{} package.json      JS instance.js X  JS notes.js
src > api > JS instance.js > [o] default
1 import axios from 'axios'
2
3
4 const apiURL = 'http://localhost:3000'
5
6
7 const instance = axios.create({
8   baseURL: apiURL
9 })
10
11
12 export default instance
```

Рисунок 5 — создание файла instance.js

The screenshot shows a code editor with three tabs: package.json, instance.js, and notes.js (selected). The notes.js tab contains the following code:

```
{} package.json      JS instance.js      JS notes.js X
src > api > JS notes.js > [o] default
1 class NotesApi {
2   constructor(instance) {
3     this.API = instance
4   }
5
6
7 getAll = async () => {
8   return this.API({
9     url: '/notes'
10  })
11 }
12
13
14 createNote = async (data) => {
15   return this.API({
16     method: 'POST',
17     url: '/notes',
18     data,
19     headers: {
20       'Content-Type': 'application/json'
21     }
22   })
23 }
24
25
26
27 export default NotesApi
```

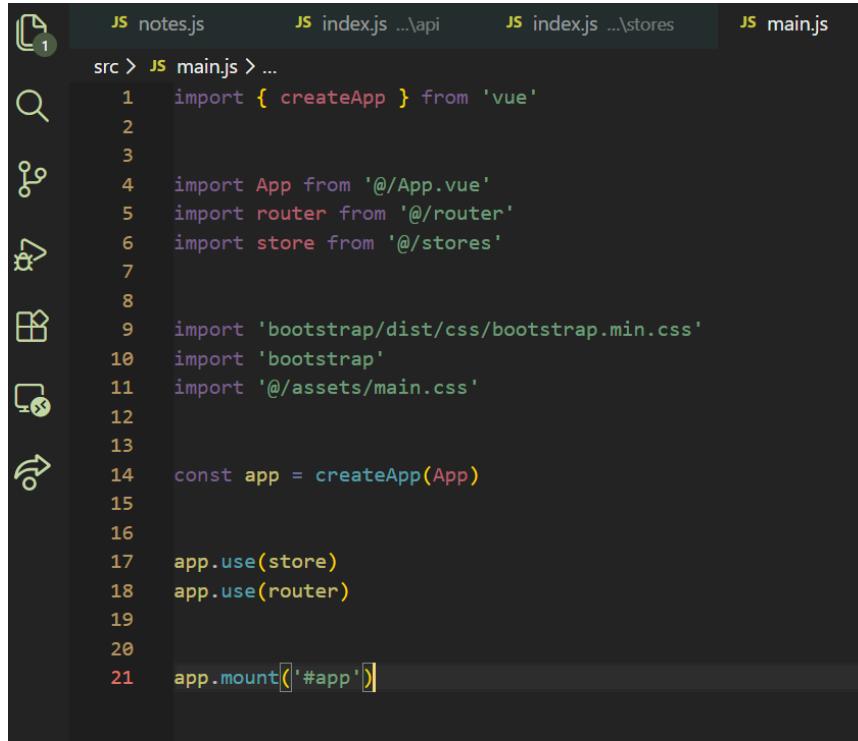
Рисунок 6 — создание файла notes.js

```
{ } package.json      JS instance.js      JS notes.js      JS index.js      X
src > api > JS index.js > ...
1  < import instance from "@/api/instance"
2  | import NotesApi from "@/api/notes"
3  |
4  |
5  | const notesApi = new NotesApi(instance)
6  |
7  |
8  < export {
9  |   notesApi
10 | }
11 |
```

Рисунок 7 — создание файла api/index.js

```
JS instance.js X      JS notes.js      JS index.js ...\\api      JS index.js ...\\stores X
src > stores > JS index.js > [?] default
1  import { persist } from 'pinia-persist'
2  import { createPinia } from 'pinia'
3  |
4  |
5  const pinia = createPinia()
6  |
7  |
8  pinia.use(persist())
9  |
10 |
11 export default pinia
```

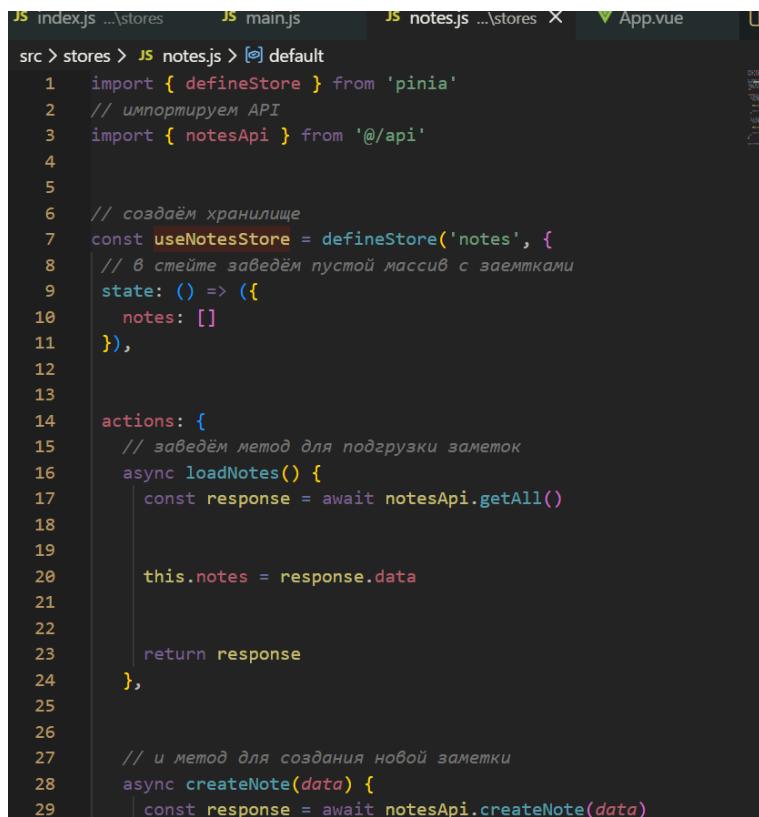
Рисунок 8 — создание файла stores/index.js



The screenshot shows a code editor with a dark theme. The tab bar at the top has four tabs: 'notes.js', 'index.js ...\\api', 'index.js ...\\stores', and 'main.js'. The 'main.js' tab is active. The code in the editor is as follows:

```
src > JS main.js > ...
1 import { createApp } from 'vue'
2
3
4 import App from '@/App.vue'
5 import router from '@/router'
6 import store from '@/stores'
7
8
9 import 'bootstrap/dist/css/bootstrap.min.css'
10 import 'bootstrap'
11 import '@/assets/main.css'
12
13
14 const app = createApp(App)
15
16
17 app.use(store)
18 app.use(router)
19
20
21 app.mount('#app')
```

Рисунок 9 — редактирование файла Main.js



The screenshot shows a code editor with a dark theme. The tab bar at the top has four tabs: 'index.js ...\\stores', 'main.js', 'notes.js ...\\stores', and 'App.vue'. The 'notes.js' tab is active. The code in the editor is as follows:

```
src > stores > JS notes.js > [o] default
1 import { defineStore } from 'pinia'
2 // импортируем API
3 import { notesApi } from '@/api'
4
5
6 // создаём хранилище
7 const useNotesStore = defineStore('notes', {
8 // в стейте заведём пустой массив с заметками
9 state: () => ({
10   notes: []
11 }),
12
13
14 actions: {
15   // заведём метод для подгрузки заметок
16   async loadNotes() {
17     const response = await notesApi.getAll()
18
19     this.notes = response.data
20
21
22     return response
23   },
24
25
26
27   // и метод для создания новой заметки
28   async createNote(data) {
29     const response = await notesApi.createNote(data)
30   }
31 })
```

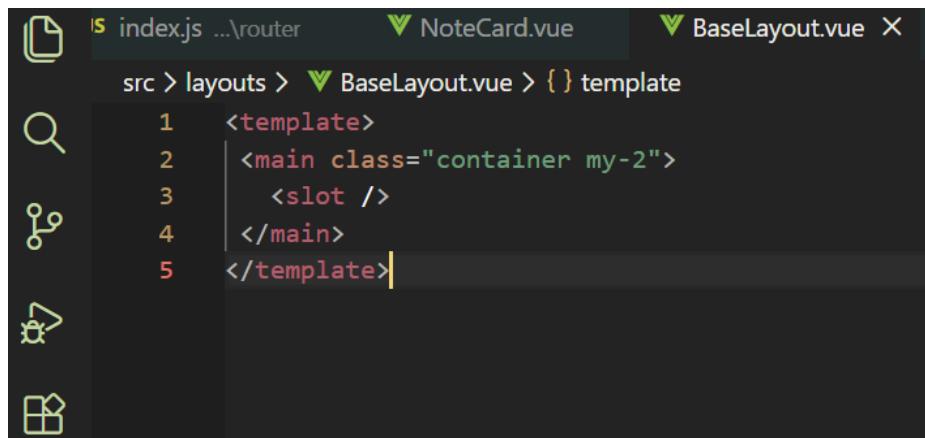
Рисунок 10 — создание файла notes.js

```
src > router > js index.js > ...
1  import { createRouter, createWebHistory } from 'vue-router'
2
3
4  const router = createRouter({
5    history: createWebHistory(import.meta.env.BASE_URL),
6    // массив с роутами
7    routes: [
8      // отдельный роут
9      {
10        path: '/',
11        name: 'notes',
12        // реализация ленивой подгрузки представления
13        // (до момента открытия этого представления,
14        // оно не будет сохранено в браузере пользователя)
15        component: () => import('../views/NotesPage.vue')
16      }
17    ]
18  })
19
20
21  // экспортруем сконфигурированный роутер
22  export default router
23  |
```

Рисунок 11 — файл с роутами

```
notes.js ...\\stores js index.js ...\\router ▼ NoteCard.vue X ▼ App.vue
src > components > ▼ NoteCard.vue > {} script
1  <template>
2    <div class="card">
3      <div class="card-body">
4        <!-- отображение свойств в шаблоне можно сделать так: -->
5        <h5 class="card-title">{{ name }}</h5>
6        <p class="card-text">
7          {{ text }}
8        </p>
9      </div>
10     </div>
11   </template>
12
13
14  <script>
15  export default {
16    name: 'NoteCard',
17
18
19    // свойства, которые принимает компонент
20    props: {
21      name: {
22        type: String,
23        required: true
24      },
25      text: {
26        type: String,
27        required: false
28      }
29    }
30  
```

Рисунок 12 — создание компоненты NoteCard.vue



```
src > layouts > ▼ BaseLayout.vue > {} template
  1  <template>
  2  | <main class="container my-2">
  3  |   <slot />
  4  | </main>
  5  </template>
```

Рисунок 13 — создание лэйаута BaseLayout.vue



```
src > views > ▼ NotesPage.vue > {} script
  1  <template>
  2  | <base-layout>
  3  |   <h1>Notes app</h1>
  4
  5
  6  | <form ref="noteForm" class="d-flex flex-column my-5">
  7  |   <input type="text" class="my-1">
  8  |   <textarea cols="30" rows="10" class="my-1" />
  9
 10 | <button type="submit" class="btn btn-primary">Отправить</button>
 11
 12
 13
 14 </form>
 15
 16
 17 <div class="row row-cols-1 row-cols-md-2 g-4 mt-5" id="note">
 18   <div class="col" v-for="note in notes" :key="note.id">
 19     <note-card :name="note.name" :text="note.text" />
 20   </div>
 21 </div>
 22 </base-layout>
 23 </template>
 24
 25
 26 <script>
 27   import BaseLayout from '@/layouts/BaseLayout.vue'
 28   import NoteCard from '@/components/NoteCard.vue'
```

Рисунок 14 — создание представления NotesPage.vue

```
> views > ▶ NotesPage.vue > {} script
1   <template>
2     <base-layout>
3       <h1>Notes app</h1>
4
5       <form
6         ref="noteForm"
7         @submit.prevent="createCard"
8         class="d-flex flex-column my-5"
9       >
10
11         <input
12           type="text"
13           v-model="form.name"
14           class="my-1"
15         >
16
17         <textarea
18           cols="30" rows="10"
19           v-model="form.text"
20           class="my-1"
21         />
22
23         <button
24           type="submit"
25           class="btn btn-primary"
26         >
27           Отправить
28         </button>
29       </form>
30
31       <div class="row row-cols-1 row-cols-md-2 g-4 mt-5" id="not
```

Рисунок 15 — добавление верстки в NotesPage.vue

```
export default {

  components: { BaseLayout, NoteCard },

  computed: {
    ...mapState(useNotesStore, ['notes'])
  },

  methods: {
    ...mapActions(useNotesStore, ['loadNotes', 'createNote']),

    async createCard() {
      await this.createNote(this.form)
      await this.loadNotes()

      this.$refs.noteForm.reset()
    }
  },

  data() {
    return {
      form: {
        name: '',
        text: '',
        userId: 1
      }
    }
  }
}
```

Рисунок 16 — методы в NotesPage.vue

Notes app

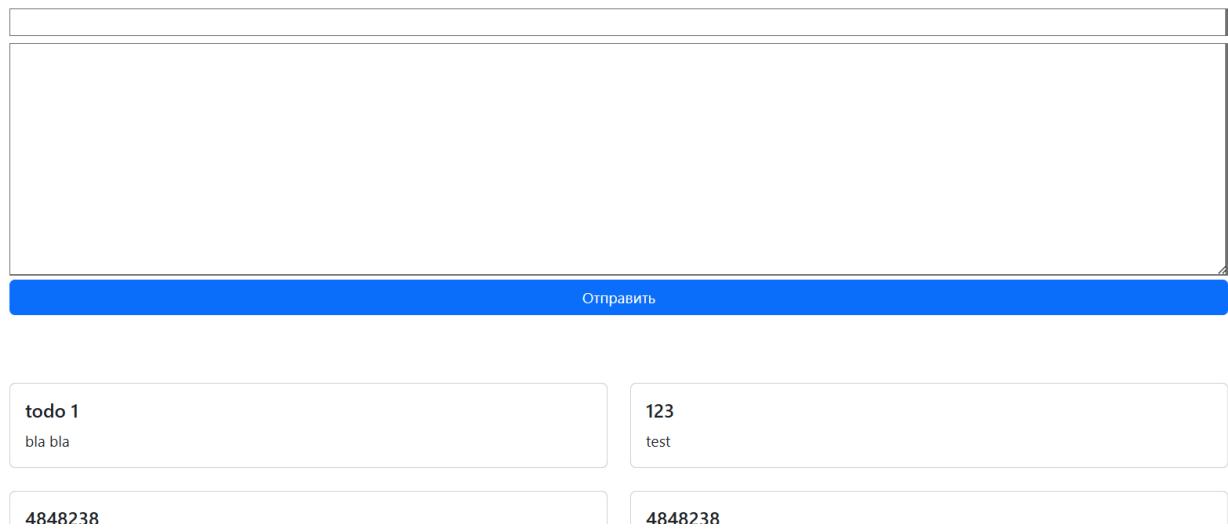


Рисунок 17 — полученное приложение

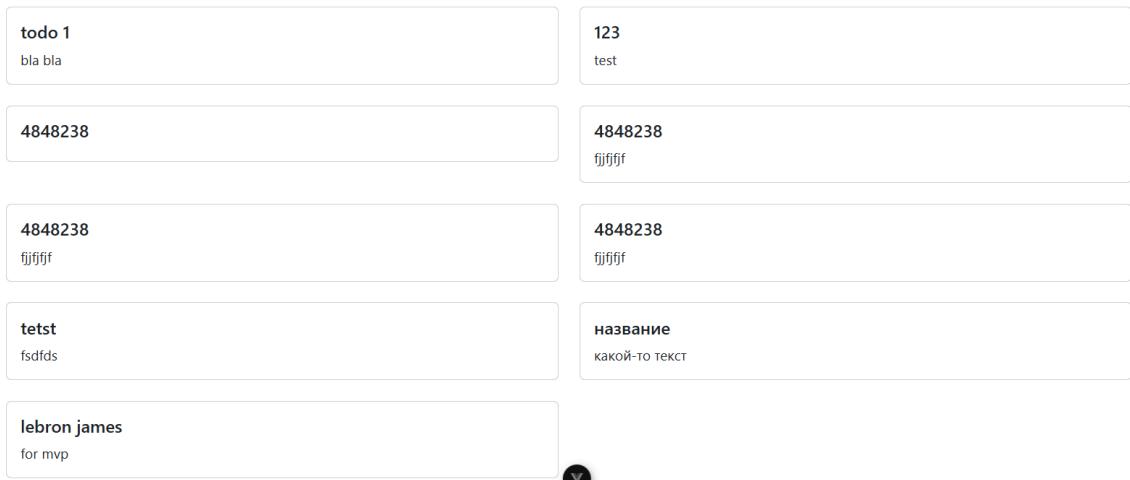


Рисунок 18 — добавление заметки

Вывод

В ходе работы было проведено знакомство с NPM и создан проект на Vue.