

**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

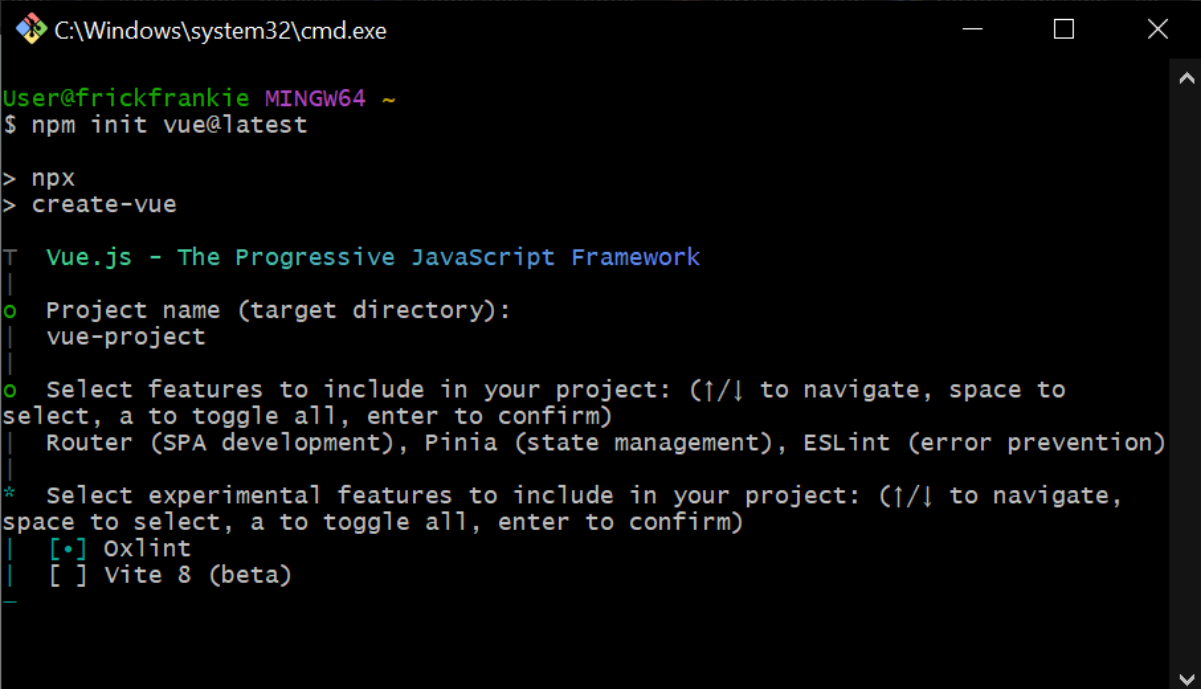
2026 г.

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:



```
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

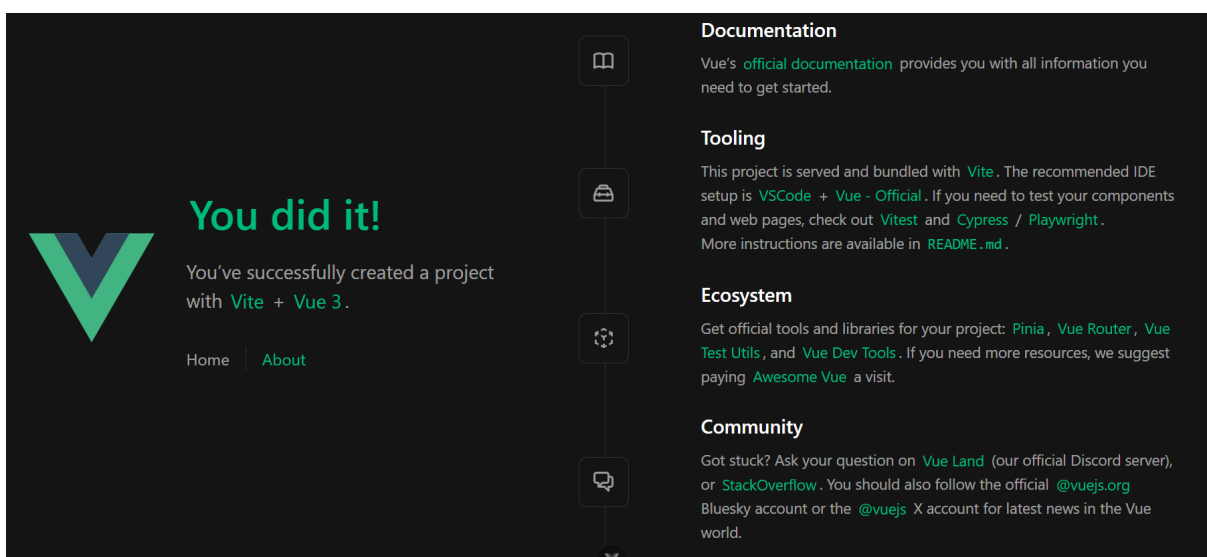
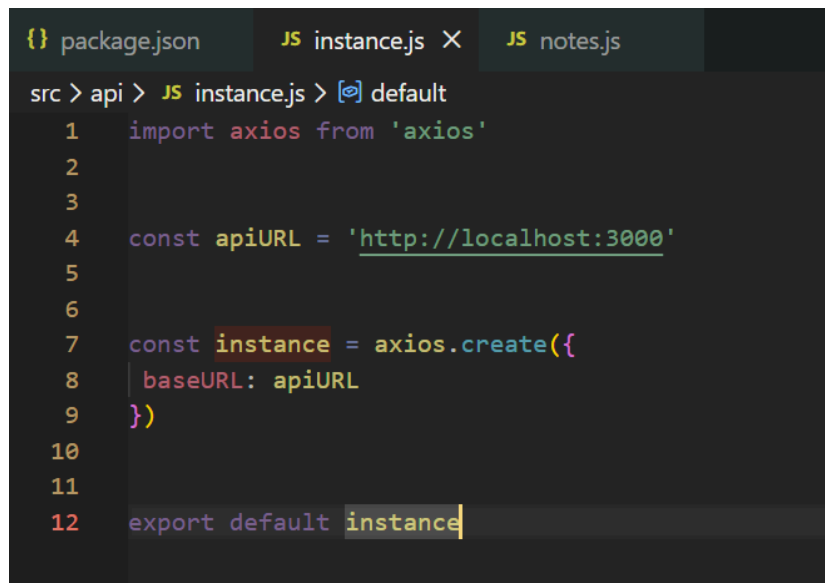


Рисунок 4 — окно после запуска проекта



The screenshot shows a VS Code editor with three tabs: package.json, instance.js, and notes.js. The instance.js tab is active, showing the following code:

```
src > api > JS instance.js > [⌕] 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
```

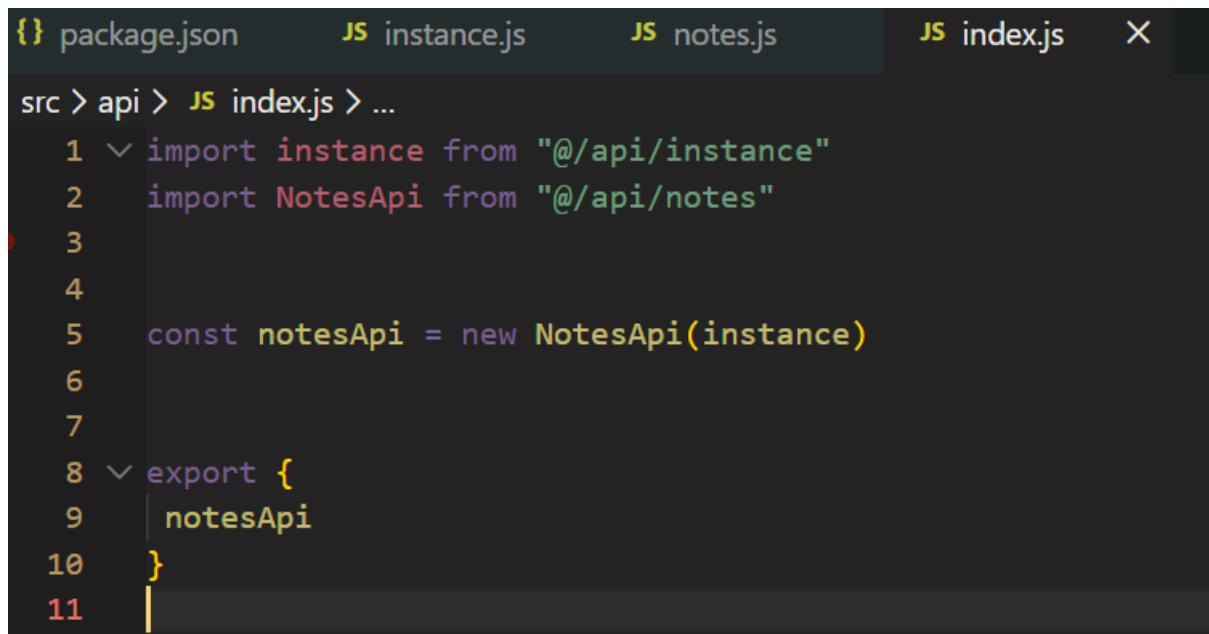
Рисунок 5 — создание файла instance.js



The screenshot shows a VS Code editor with three tabs: package.json, instance.js, and notes.js. The notes.js tab is active, showing the following code:

```
src > api > JS notes.js > [⌕] 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
```

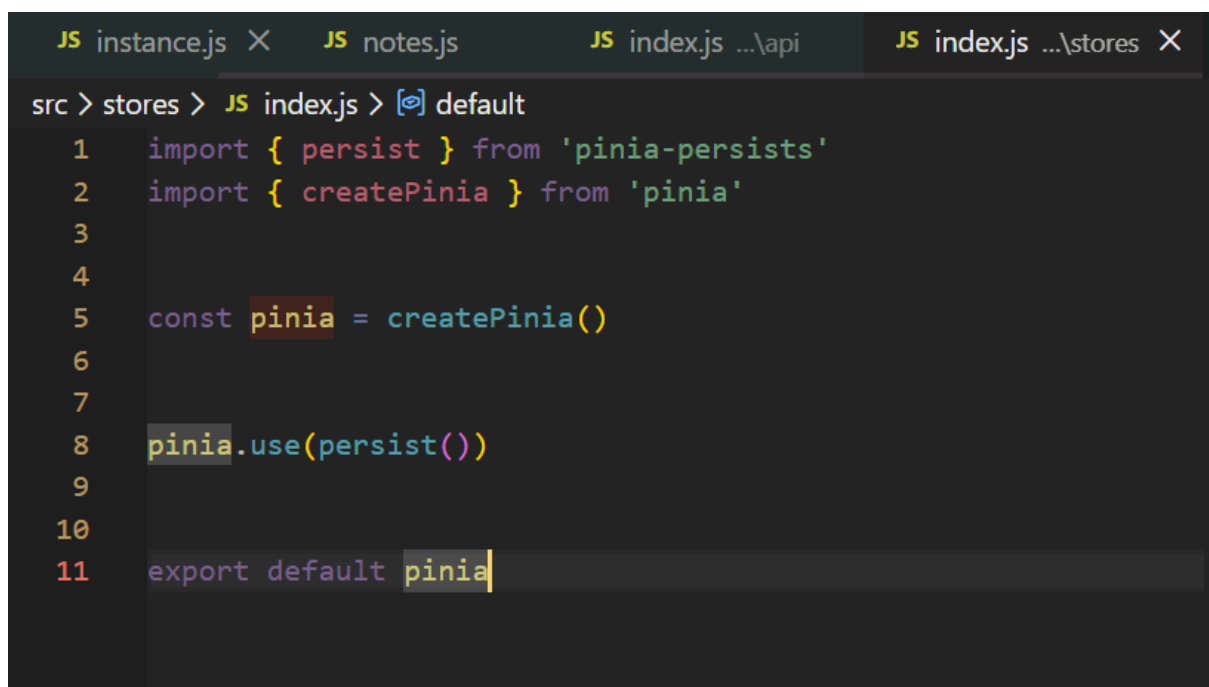
Рисунок 6 — создание файла notes.js



The screenshot shows a VS Code editor with four tabs: package.json, instance.js, notes.js, and index.js. The active tab is index.js, which contains the following code:

```
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
```

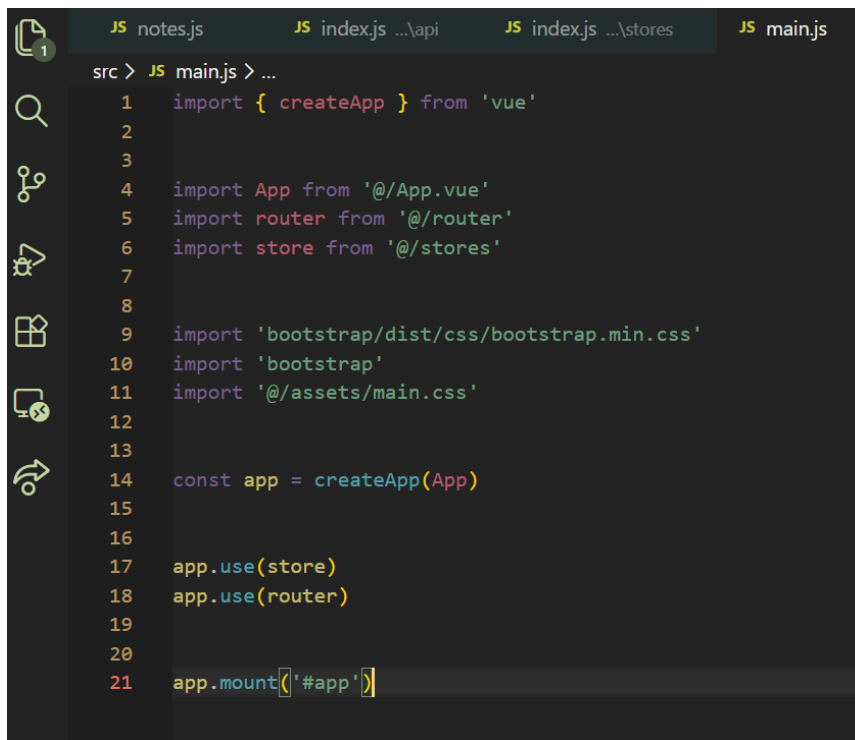
Рисунок 7 — создание файла api/index.js



The screenshot shows a VS Code editor with four tabs: instance.js, notes.js, index.js ..\api, and index.js ..\stores. The active tab is index.js ..\stores, which contains the following code:

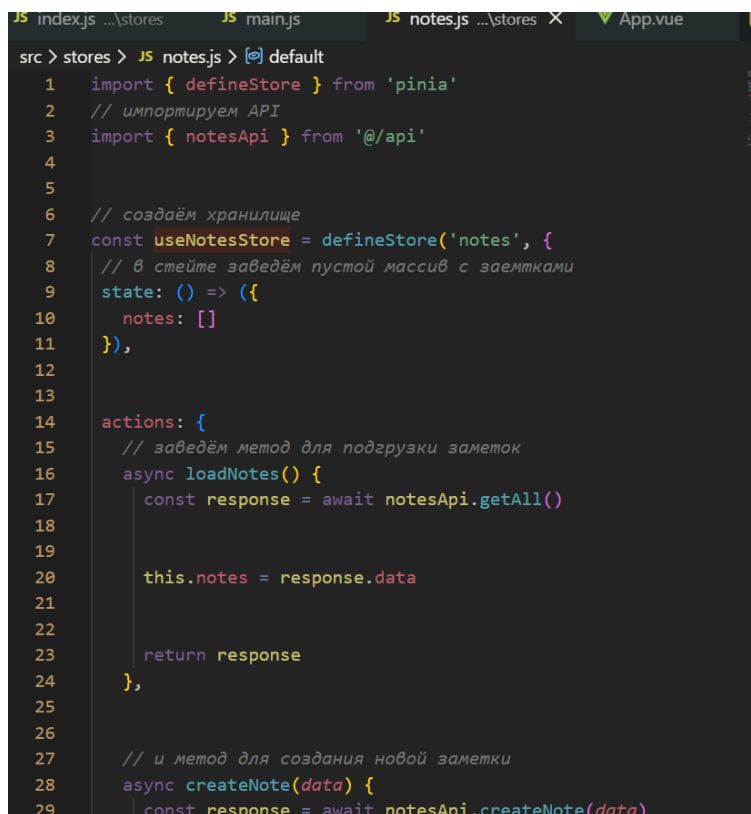
```
src > stores > JS index.js > [🔗] default
1  import { persist } from 'pinia-persists'
2  import { createPinia } from 'pinia'
3
4
5  const pinia = createPinia()
6
7
8  pinia.use(persist())
9
10
11 export default pinia
```

Рисунок 8 — создание файла stores/index.js



```
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



```
JS index.js ...\stores JS main.js JS notes.js ...\stores X App.vue
src > stores > JS notes.js > 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 записками
9    state: () => ({
10      notes: []
11    }),
12
13
14    actions: {
15      // заведём метод для загрузки заметок
16      async loadNotes() {
17        const response = await notesApi.getAll()
18
19        this.notes = response.data
20
21        return response
22      },
23
24      // и метод для создания новой заметки
25      async createNote(data) {
26        const response = await notesApi.createNote(data)
27      }
28    }
29  })
```

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 загрузки 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 // экспортируем сконфигурированный роутер
22 export default router
23 |

```

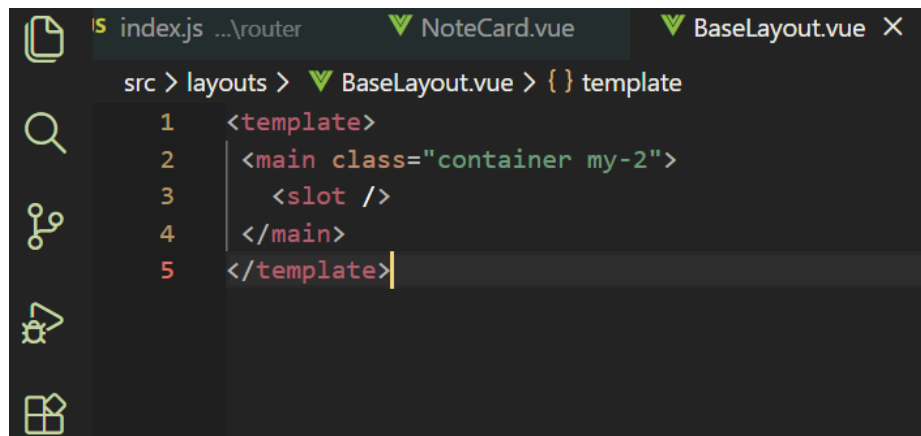
Рисунок 11 — файл с роутами

```

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     // свойства, которые принимает компонент
19     props: {
20       name: {
21         type: String,
22         required: true
23       },
24       text: {
25         type: String,
26         required: false
27       }
28     }

```

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 лэяута 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
11     <button type="submit" class="btn btn-primary">Отправить</button>
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       <input
11         type="text"
12         v-model="form.name"
13         class="my-1"
14       >
15
16       <textarea
17         cols="30" rows="10"
18         v-model="form.text"
19         class="my-1"
20       />
21
22       <button
23         type="submit"
24         class="btn btn-primary"
25       >
26         Отправить
27       </button>
28     </form>
29
30     <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

Отправить

todo 1

bla bla

123

test

4848238

4848238

Рисунок 17 — полученное приложение

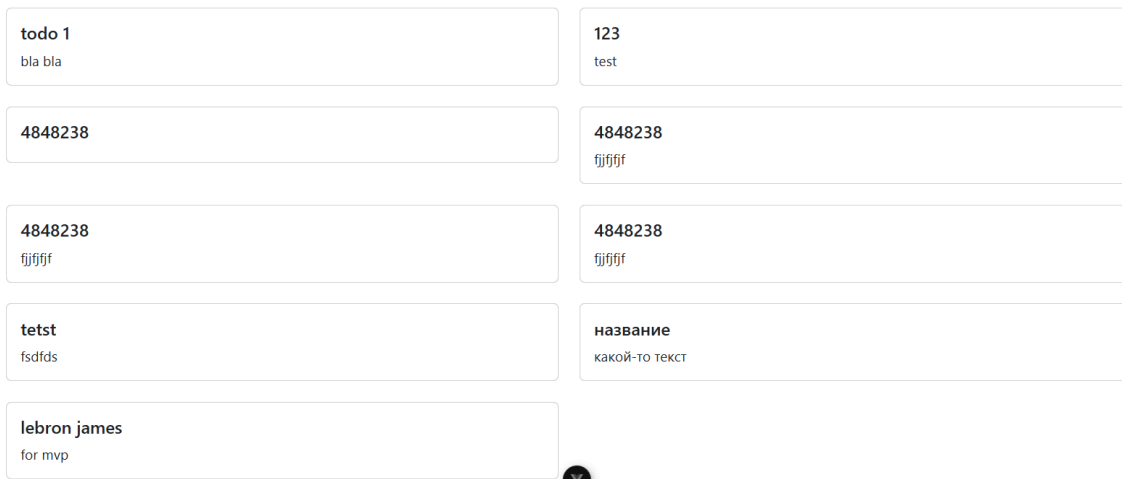


Рисунок 18 — добавление заметки

Вывод

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