

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Домашняя работа № 5

Выполнила:

Стукалова Александра

Группа К3343

**Проверил:
Добряков Д. И.**

**Санкт-Петербург
2026 г.**

Я начала работу с проверки установленных версий Node.js и npm, чтобы убедиться, что они установлены.

Затем я выполнила команду npm init vue@latest для создания нового проекта на Vue.js.

```
C:\Users\Professional>node --version
v24.13.1

C:\Users\Professional>npm --version
11.8.0

C:\Users\Professional>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)
  none

o Select experimental features to include in your project: (↑↓ to navigate, space to select, a to toggle all, enter to confirm)
  none

o Skip all example code and start with a blank Vue project?
  Yes

Scaffolding project in C:\Users\Professional\vue-project...
|
- Done. Now run:

  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"

C:\Users\Professional>cd vue-project
C:\Users\Professional\vue-project>
```

После создания проекта я выполнила команду npm install для установки всех необходимых пакетов:

```
C:\Users\Professional\vue-project>npm install

added 123 packages, and audited 124 packages in 18s

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

found 0 vulnerabilities

C:\Users\Professional\vue-project>
```

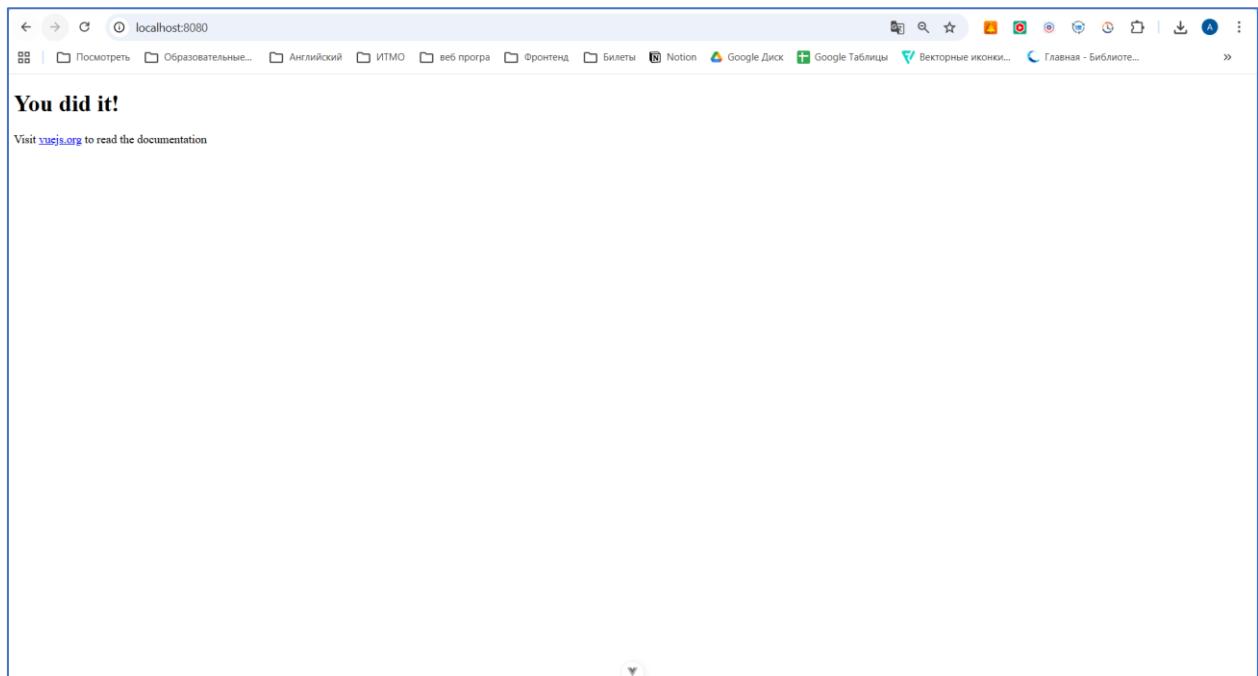
После установки зависимостей я запустила проект командой npm start. В терминале появилось сообщение о том, что сервер запущен на порту 8080:

```
Administrator: C:\Windows\system32\cmd.exe
```

```
VITE v7.3.1 ready in 1143 ms

Local: http://localhost:8080/
Network: use --host to expose
Vue DevTools: Open http://localhost:8080/_devtools_ as a separate window
Vue DevTools: Press Alt(Alt)+Shift(Shift)+D in App to toggle the Vue DevTools
press h + enter to show help
```

Я перешла по адресу <http://localhost:8080>, где увидела стандартную страницу-
Vue:



Для работы с запросами к API, управления состоянием и стилями я
установила необходимые пакеты командой:

```
C:\Users\Professional\vue-project>npm i axios pinia-persists bootstrap -S
added 26 packages, and audited 150 packages in 5s

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

found 0 vulnerabilities

C:\Users\Professional\vue-project>
```

- axios - библиотека для HTTP-запросов
- pinia-persists - плагин для Pinia, сохраняющий состояние в localStorage

- bootstrap - фреймворк для стилей
- ключ -S добавляет зависимости в package.json

Для работы с сервером я создала папку src/api и настроила axios. В файле instance.js задала базовый URL для запросов:

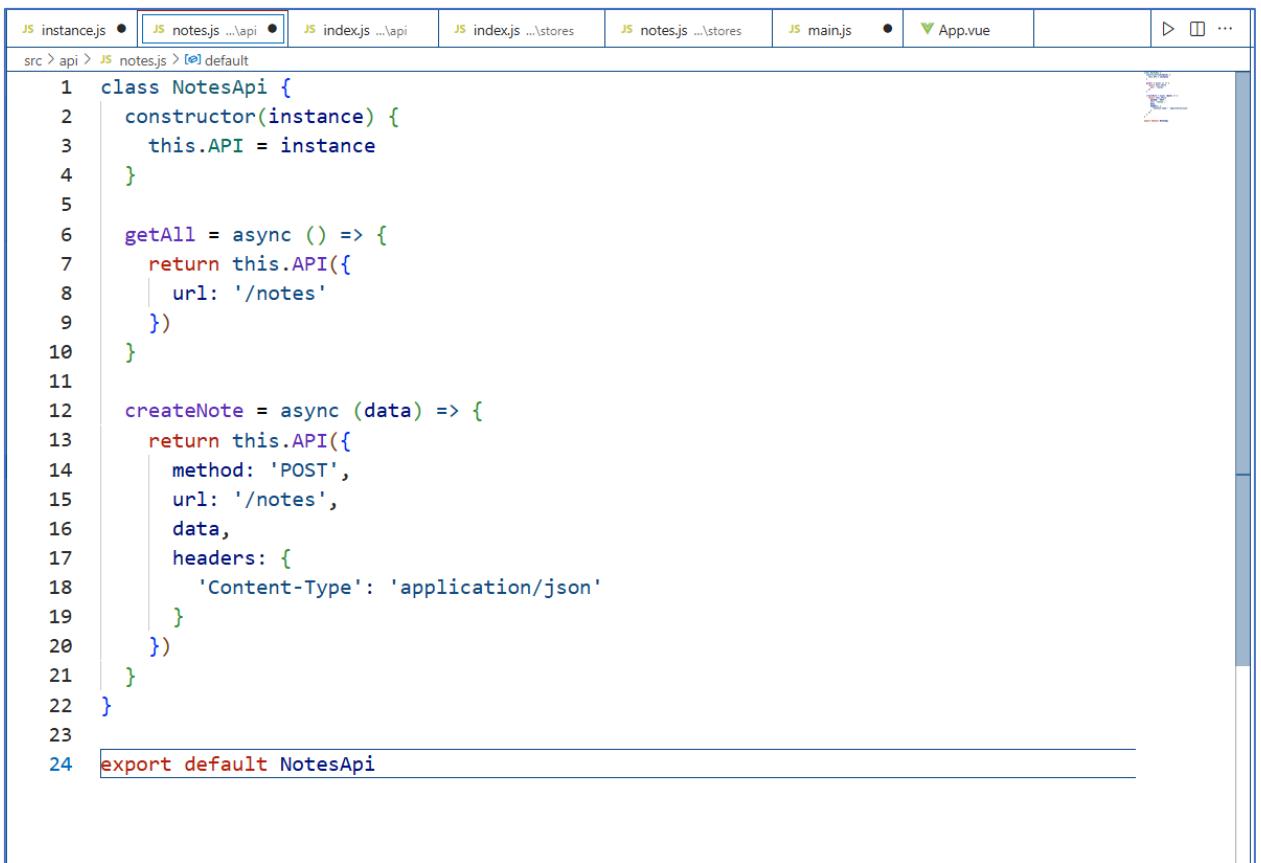


```

JS instance.js • JS notes.js ...\\api • JS index.js ...\\api JS index.js ...\\stores JS notes.js ...\\stores JS main.js • ▾ App.vue
src > api > JS instance.js > [o] default
1 import axios from 'axios'
2
3 const apiURL = 'http://localhost:3000'
4
5 const instance = axios.create({
6   baseURL: apiURL
7 })
8
9 export default instance

```

src/api/notes.js - Я создала класс NotesApi с методами для получения всех заметок и создания новой:



```

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

```

src/api/index.js - Собирает все API в одном месте. Здесь создается экземпляр NotesApi с настроенным axios, чтобы другие файлы могли просто импортировать готовое API:

```
JS instance.js • JS notes.js ...\\api • JS index.js ...\\api X JS index.js ...\\stores JS notes.js ...\\stores JS main.js • ▾ App.vue
src > api > JS index.js > ...
1 import instance from "./instance"
2 import NotesApi from "./notes"
3
4 const notesApi = new NotesApi(instance)
5
6 export {
7   notesApi
8 }
```

src/stores/index.js - Настраивает Pinia (менеджер состояний). Подключает плагин pinia-persists, который автоматически сохраняет данные в localStorage, чтобы они не терялись при перезагрузке:

```
JS instance.js • JS notes.js ...\\api • JS index.js ...\\api JS index.js ...\\stores X JS notes.js ...\\stores JS main.js • ▾ App.vue
src > stores > JS index.js > [o] default
1 import { persist } from 'pinia-persists'
2 import { createPinia } from 'pinia'
3
4 const pinia = createPinia()
5 pinia.use(persist())
6
7 export default pinia
```

src/stores/notes.js - Создает хранилище для заметок. В нем:

- state - массив notes, где хранятся все заметки
- actions - методы loadNotes (загружает заметки с сервера) и createNote (создает новую заметку и обновляет список):

```

JS instance.js • JS notes.js ...\\api • JS index.js ...\\api JS index.js ...\\stores JS notes.js ...\\stores X JS main.js • App.vue
src > stores > JS notes.js > default
1 import { defineStore } from 'pinia'
2 import { notesApi } from '@/api'
3
4 const useNotesStore = defineStore('notes', {
5   state: () => ({
6     notes: []
7   }),
8   actions: {
9     async loadNotes() {
10       const response = await notesApi.getAll()
11       this.notes = response.data
12       return response
13     },
14     async createNote(data) {
15       const response = await notesApi.createNote(data)
16       this.notes = response.data
17       return response
18     }
19   }
20 })
21
22 export default useNotesStore

```

src/main.js - Главный файл приложения. Здесь:

- Подключаются Bootstrap стили и скрипты
- Подключается Pinia (store)
- Подключается роутер
- Запускается само приложение:

```

JS instance.js • JS notes.js ...\\api • JS index.js ...\\api JS index.js ...\\stores JS notes.js ...\\stores JS main.js • App.vue
src > JS main.js > ...
1 import { createApp } from 'vue'
2 import App from '@/App.vue'
3 import router from '@/router'
4 import store from '@/stores'
5 import 'bootstrap/dist/css/bootstrap.min.css'
6 import 'bootstrap'
7 import '@/assets/main.css'
8
9 const app = createApp(App)
10
11 app.use(store)
12 app.use(router)
13
14 app.mount('#app')

```

src/App.vue - Корневой компонент. В нем только <router-view>, куда будут подставляться страницы в зависимости от адреса:

The screenshot shows a code editor with multiple tabs at the top: instance.js, notes.js, index.js, notes.js, main.js, App.vue (which is the active tab), and others. The code in App.vue is:

```
src > App.vue
1 <template>
2   <router-view />
3 </template>
```

src/components>NoteCard.vue – я создала компонент для отображения одной заметки. Он принимает два свойства: `name` (заголовок) и `text` (текст). В шаблоне данные выводятся внутри карточки Bootstrap. Этот компонент будет использоваться для каждой заметки в списке:

The screenshot shows a code editor with the NoteCard.vue component. The code is:

```
1 <template>
2   <div class="card">
3     <div class="card-body">
4       <h5 class="card-title">{{ name }}</h5>
5       <p class="card-text">{{ text }}</p>
6     </div>
7   </div>
8 </template>
9
10 <script>
11 export default {
12   name: 'NoteCard',
13   props: {
14     name: {
15       type: String,
16       required: true
17     },
18     text: {
19       type: String,
20       required: false
21     }
22   }
23 }
24 </script>
```

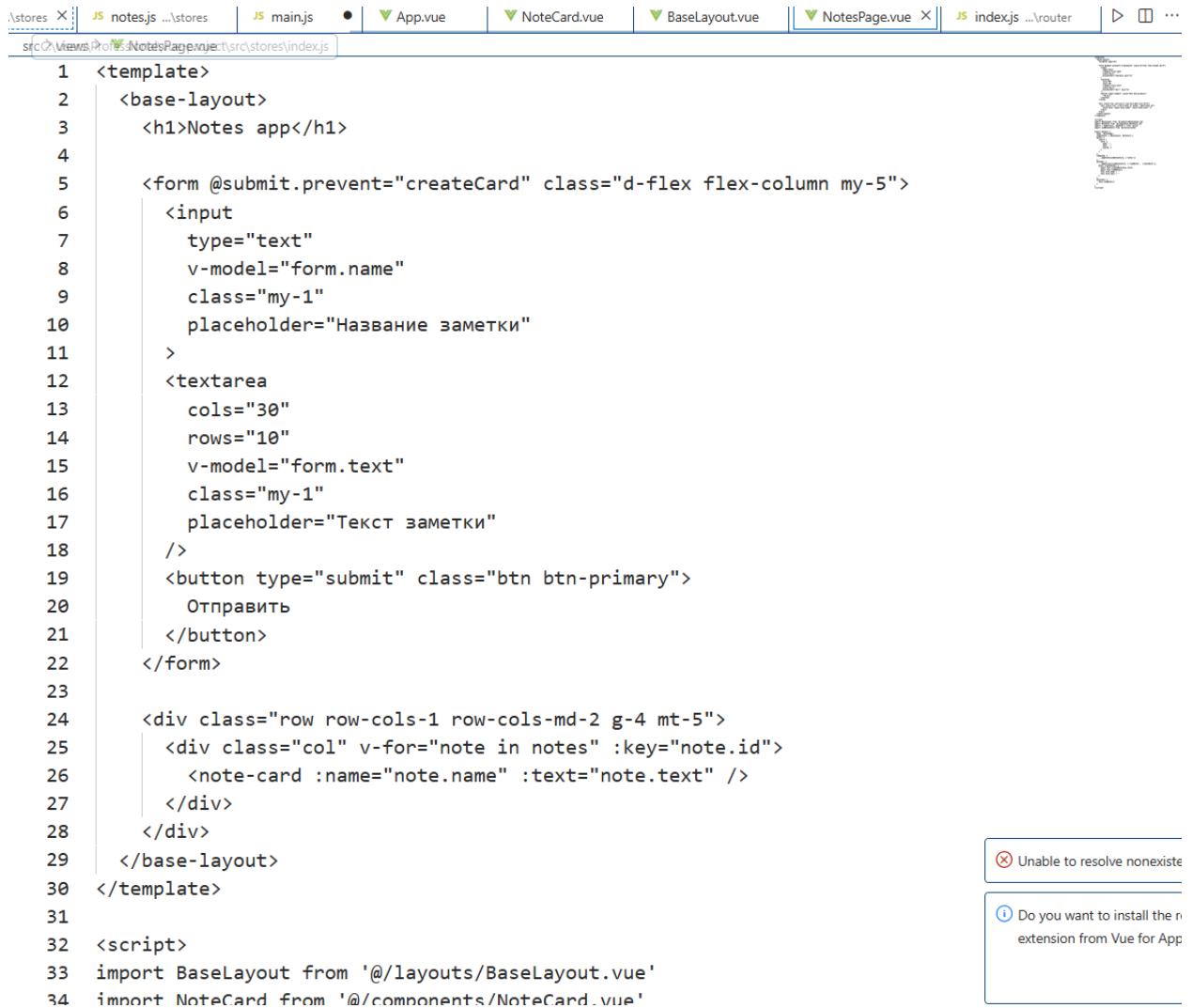
src/layouts/BaseLayout.vue – я создала базовую обертку для всех страниц. Он содержит контейнер с отступами из Bootstrap и тег `<slot>`, на место которого будет подставляться содержимое конкретной страницы. Это позволяет не дублировать код с контейнером на каждой странице:

The screenshot shows a code editor with the BaseLayout.vue component. The code is:

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

src/views/NotesPage.vue – я создала главную страницу приложения. На ней есть:

- форма для добавления новой заметки с полями Название и Текст
- кнопка отправки
- список всех заметок, который выводится с помощью v-for, и каждая заметка отображается через компонент NoteCard:



```
src\Views\Project\src\views\NotesPage.vue
1 <template>
2   <base-layout>
3     <h1>Notes app</h1>
4
5     <form @submit.prevent="createCard" class="d-flex flex-column my-5">
6       <input
7         type="text"
8         v-model="form.name"
9         class="my-1"
10        placeholder="Название заметки"
11      >
12      <textarea
13        cols="30"
14        rows="10"
15        v-model="form.text"
16        class="my-1"
17        placeholder="Текст заметки"
18      />
19      <button type="submit" class="btn btn-primary">
20        Отправить
21      </button>
22    </form>
23
24    <div class="row row-cols-1 row-cols-md-2 g-4 mt-5">
25      <div class="col" v-for="note in notes" :key="note.id">
26        <note-card :name="note.name" :text="note.text" />
27      </div>
28    </div>
29  </base-layout>
30 </template>
31
32 <script>
33 import BaseLayout from '@/layouts/BaseLayout.vue'
34 import NoteCard from '@/components/NoteCard.vue'
```

.tooltip { border: 1px solid #ccc; padding: 5px; background-color: #fff; } .tooltip::before { content: ''; position: absolute; top: -5px; left: 50%; margin-left: -5px; border-top: 5px solid transparent; border-bottom: 5px solid transparent; border-left: 10px solid #ccc; } .tooltip .text { border: 1px solid #ccc; padding: 2px; background-color: #fff; } .tooltip .info { border: 1px solid #ccc; padding: 2px; background-color: #fff; }

✖ Unable to resolve nonexistent module 'note-card'. Do you want to install the 'Vue for App' extension?

```

34 import NoteCard from '@/components>NoteCard.vue'
35 import { mapActions, mapState } from 'pinia'
36 import useNotesStore from '@/stores/notes'
37
38 export default {
39   name: 'NotesPage',
40   components: { BaseLayout, NoteCard },
41   data() {
42     return {
43       form: {
44         name: '',
45         text: '',
46         userId: 1
47       }
48     }
49   },
50   computed: {
51     ...mapState(useNotesStore, ['notes'])
52   },
53   methods: {
54     ...mapActions(useNotesStore, ['loadNotes', 'createNote']),
55     async createCard() {
56       await this.createNote(this.form)
57       await this.loadNotes()
58       this.form.name = ''
59       this.form.text = ''
60     }
61   },
62   mounted() {
63     this.loadNotes()
64   }
65 }
66 </script>

```

src/router/index.js – я настроила роутер. Добавила один маршрут:

- path: '/' - корневой адрес
- name: 'notes' - имя маршрута
- component: () => import('../views/NotesPage.vue') - страница, которая загружается:

```

src > router > JS index.js > [e] default
1 import { createRouter, createWebHistory } from 'vue-router'
2
3 const router = createRouter({
4   history: createWebHistory(import.meta.env.BASE_URL),
5   routes: [
6     {
7       path: '/',
8       name: 'notes',
9       component: () => import('../views/NotesPage.vue')
10    }
11  ]
12})
13
14 export default router

```

Далее, чтобы убедиться, что JSON-server запущен, я открываю новое окно терминала и выполняю:

```
C:\Users\Professional>npm install -g json-server
changed 43 packages in 9s

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

C:\Users\Professional>
```

В папке проекта я создала файл db.json со следующим содержимым:



```
src > db.json > ...
1  [
2    "notes": [
3      {
4        "id": 1,
5        "name": "Первая заметка",
6        "text": "Это пример заметки",
7        "userId": 1
8      }
9    ]
10 ]
```

Благодаря JSON-server эти данные автоматически превращаются в полноценное API. Заметки можно получать на <http://localhost:3000/notes>, добавлять новые через POST запросы.

Для работы приложения необходимо одновременно запустить два сервера в разных окнах терминала. В первом окне терминала я перешла в папку проекта и выполнила команду:

```
C:\Users\Professional\vue-project>json-server --watch db.json --port 3000
--watch/-w can be omitted, JSON Server 1+ watches for file changes by default
JSON Server started on PORT :3000
Press CTRL-C to stop
Watching db.json...

(  )

Index:
http://localhost:3000/

Static files:
Serving ./public directory if it exists

Endpoints:
http://localhost:3000/notes
```

Эта команда запускает API-сервер, который читает данные из файла db.json. Это окно терминала должно оставаться открытым все время работы приложения.

Во втором окне терминала я запустила сервер разработки Vite:

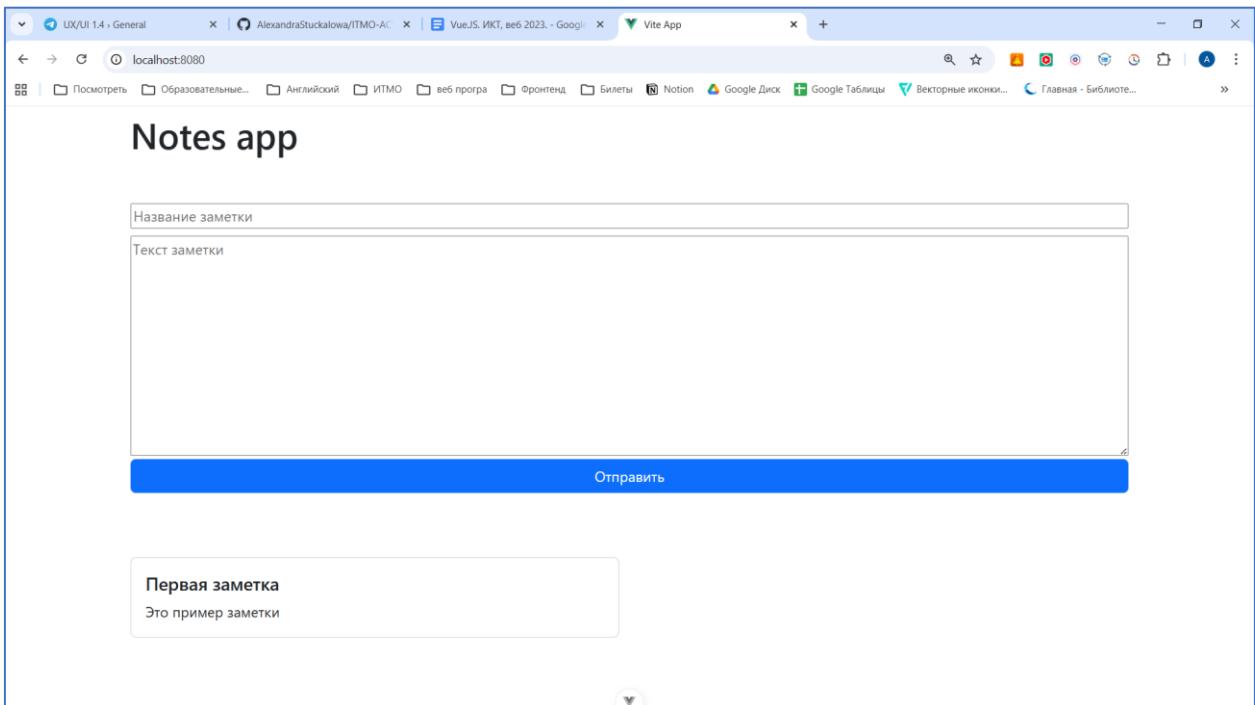
```
C:\Users\Professional\vue-project>npm start

> vue-project@0.0.0 start
> vite --port 8080

VITE v7.3.1 ready in 668 ms

  Local: http://localhost:8080/
  Network: use --host to expose
  Vue DevTools: Open http://localhost:8080/__devtools__/ as a separate window
  Vue DevTools: Press Alt(⌘)+Shift(⇧)+D in App to toggle the Vue DevTools
  press h + enter to show help
```

Все работает:



Я добавила заметку:

