

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Фронт-энд разработка

Отчет

Домашняя работа

Выполнил:

Андронов Денис

Группа

К3342

Проверил:

Добряков Д. И.

Санкт-Петербург

2026 г.

## **Задача**

Домашняя работа 5: изучение основ работы с менеджером зависимостей  
npm

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:

Поставил пакеты bootstrap, axios, pinia-persists.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
cd vue-project
npm install
npm run dev

| Optional: Initialize Git in your project directory with: ...
PS E:\yureba\frontend\ITMO-ACS-Frontend-2025\homeworks\K3342\Аппаратная часть\vue-project> npm i axios pinia-persist bootstrap -S
added 26 packages, and audited 255 packages in 4s

65 packages are looking for funding
  run "npm fund" for details

found 0 vulnerabilities
PS E:\yureba\frontend\ITMO-ACS-Frontend-2025\homeworks\K3342\Аппаратная часть\vue-project>
```

Получившаяся файловая система:

## EXPLORER

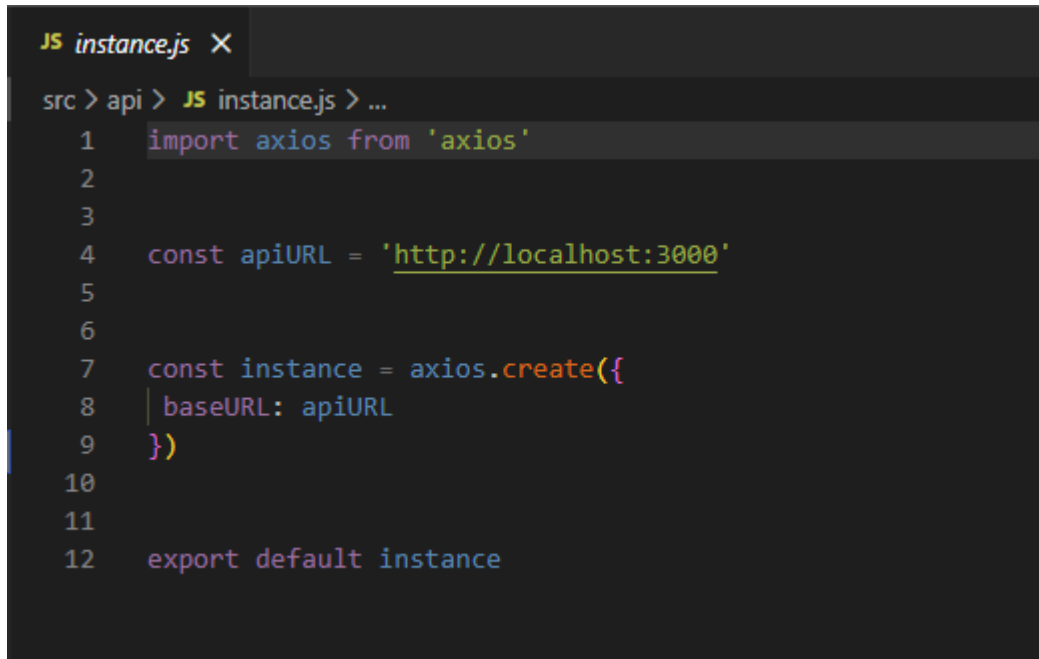


## VUE-PROJECT

- > .vite
- > .vscode
- > node\_modules
- > public
- ▼ src
  - ▼ api
    - JS index.js
    - JS instance.js
    - JS notes.js
  - ▼ assets
    - # base.css
    - logo.svg
    - # main.css
  - ▼ components
    - > icons
    - ▼ NoteCard.vue
  - ▼ layouts
    - ▼ BaseLayout.vue
  - ▼ router
    - JS index.js
  - ▼ stores
    - JS index.js
    - JS notes.js
  - ▼ views
    - ▼ NotesPage.vue
    - ▼ App.vue
  - JS main.js
- ◆ .gitattributes
- ◆ .gitignore
- { } db.json
- <> index.html
- > { } package.json
- ① README.md
- > ⚡ vite.config.js

Получившиеся файлы:

Src/api/instance.js

A screenshot of a code editor window with a dark theme. The title bar at the top shows 'JS instance.js' and a close button. The editor content shows a file path 'src > api > JS instance.js > ...' followed by 12 lines of JavaScript code. Line 1: 'import axios from 'axios''; Line 2: empty; Line 3: empty; Line 4: 'const apiURL = 'http://localhost:3000''; Line 5: empty; Line 6: empty; Line 7: 'const instance = axios.create({'; Line 8: ' | baseUrl: apiURL'; Line 9: '});'; Line 10: empty; Line 11: empty; Line 12: 'export default instance'.

```
JS instance.js ×
src > api > JS instance.js > ...
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
```

src/api/notes.js

```
JS notes.js  X
src > api > JS notes.js > NotesApi
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

```
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
```

src/stores/index.js

```
JS index.js  X
src > stores > JS index.js > ...
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
12 |
```

Src/stores/notes.js



```
JS notes.js X
src > stores > JS notes.js > ...
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 заемтками
9    state: () => ({
10      notes: []
11    }),
12
13    actions: {
14      // заведём метод для подгрузки заметок
15      async loadNotes() {
16        const response = await notesApi.getAll()
17
18
19        this.notes = response.data
20
21
22        return response
23      },
24
25      // и метод для создания новой заметки
26      async createNote(data) {
27        const response = await notesApi.createNote(data)
28
29
30        this.notes = response.data
31
32
33        return response
34      }
35    }
36  })
37
38
39  export default useNotesStore
40
```

Main.js

```
JS main.js  X
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  app.mount('#app')
```

Src/App.vue

```
▼ App.vue  X
src > ▼ App.vue > { } template
1  <template>
2  | <router-view />
3  </template>
```

src/router/index.js

```
JS index.js X
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
```

Src/components/NoteCard.vue

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

Src/layouts/BaseLayout.vue

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

Src/views/NotesPage.vue

```
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 gap-2 my-4"
9     >
10      <input
11        type="text"
12        v-model="form.name"
13        class="my-1"
14      >
15
16      <textarea
17        cols="30"
18        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="notes">
32      <div
33        class="col"
34        v-for="note in notes"
35        :key="note.id"
36      >
37        <note-card
38          :name="note.name"
39          :text="note.text"
40        />
41      </div>
42    </div>
43  </base-layout>
44 </template>
45
```

```
NotesPage.vue X
src > views > NotesPage.vue > {} template
1  <template>
44 </template>
45
46 <script>
47 import { mapActions, mapState } from 'pinia'
48 import useNotesStore from '@stores/notes'
49 import BaseLayout from '@layouts/BaseLayout.vue'
50 import NoteCard from '@components/NoteCard.vue'
51 export default {
52   name: 'NotesPage',
53   components: { BaseLayout, NoteCard },
54   data() {
55     return {
56       form: {
57         name: '',
58         text: '',
59         userId: 1
60       }
61     }
62   },
63   computed: {
64     ...mapState(useNotesStore, ['notes'])
65   },
66   methods: {
67     ...mapActions(useNotesStore, ['loadNotes', 'createNote']),
68     async createCard() {
69       await this.createNote(this.form)
70       await this.loadNotes()
71       this.$refs.noteForm.reset()
72       this.form.name = ''
73       this.form.text = ''
74     }
75   },
76   mounted() {
77     this.loadNotes()
78   }
79 }
80 </script>
```

Работа приложения

## Notes app

Отправить

123

123

456

123

213

123

123

123

авава

321312312321па

### Вывод

В ходе работы удалось изучить основные команды пакетного менеджера NPM и научиться стартовать проект на Vue. Отдельно удалось разобраться со структурой Vue-проекта.