React: useContext

НАШИ ПРАВИЛА

Включенная камера

Вопросы по поднятой руке

Не перебиваем друг друга

Все вопросы, не связанные с тематикой курса (орг-вопросы и т. д.), должны быть направлены куратору

Подготовьте свое рабочее окружение для возможной демонстрации экрана (закройте лишние соцсети и прочие приложения)

Повторим;)

- В чем разница SPA и MPA?
- Какие типы ссылок есть в react router?
- Какие вспомогательные хуки в react router вы знаете?
- Для чего используется useFormik?
- Как выполнить действие при размонтировании компонента?

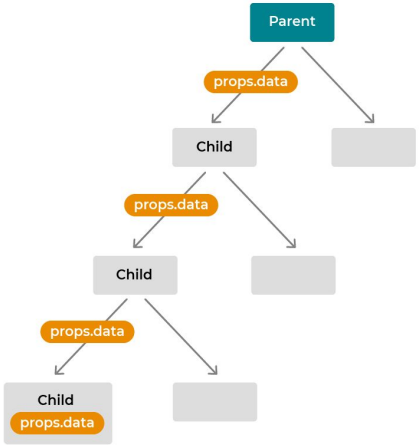
ЦЕЛЬ

Изучить способ передачи данных через большую вложенность компонентов

ПЛАН ЗАНЯТИЯ

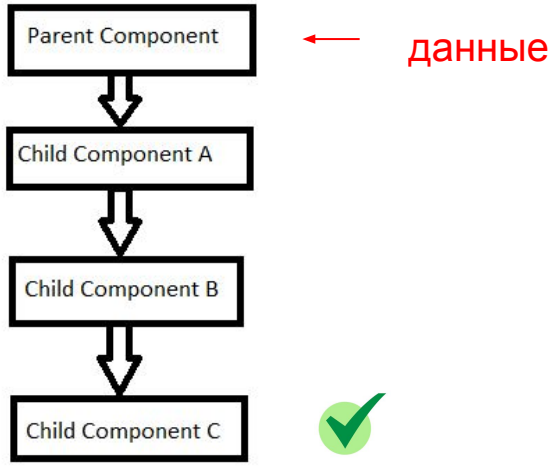
1. Props drilling
2. useContext
3. Установка Vite проекта

useContext


Parent
props.data
Child
props.data
Child
props.data
Child
props.data

useContext - это React Hook, предоставляющий способ получения значения из контекста в функциональных компонентах React.

Контекст в React используется для передачи данных глубоко вложенным компонентам без явной передачи пропс через каждый уровень.



Шаг 1. Создаём контекст в компоненте, который будет передавать значение

```
import { createContext, useState } from 'react';

// Создаем контекст с типом React.Context и значением по умолчанию
const MyContext = createContext<string>("");
```

создаём контекст

```
// Компонент-поставщик контекста
const MyContextProvider = ({ children }) => {
  const [sharedValue, setSharedValue] = useState('Значение из контекста')

  const updateValue = (newValue) => {
    setSharedValue(newValue);
  };

  return (
    <MyContext.Provider value={{ sharedValue, updateValue }}>
      {children}
    </MyContext.Provider>
  );
};

export { MyContext, MyContextProvider };
```

делаем компонент поставщиком контекста

Шаг 2. Соответствующий <Context.Provider> должен находиться выше компонента, которому нужны данные из контекста. В данном случае **MyContextProvider** находится на самом верхнем уровне, но это необязательно приложения

```
import React from 'react';
import { MyContextProvider } from './MyContextProvider';
import MyComponent from './MyComponent';

const App = () => {
  return (
    <MyContextProvider>
      <div>
        <h1>Мое React Приложение</h1>
        <MyComponent />
        {/* Другие компоненты в приложении */}
      </div>
    </MyContextProvider>
  );
};

export default App;
```

импортируем компонент с контекстом

Оборачиваем другие компоненты

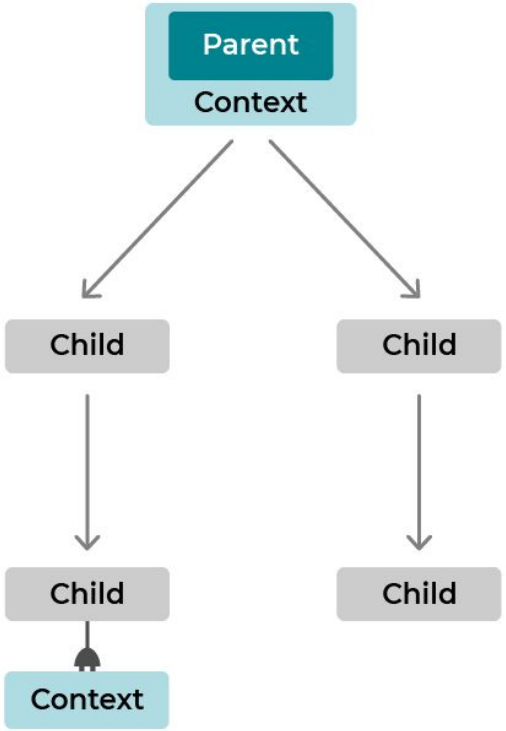
Шаг 3. Передаём значение из компонента поставщика (провайдера) в другой компонент

```jsx
import React, { useContext } from 'react';
import { MyContext } from './MyContextProvider';

// Компонент, использующий значение из контекста с помощью useContext
const MyComponent = () => {
  const { sharedValue, updateValue } = useContext(MyContext);

  const handleButtonClick = () => {
    // Обновляем значение в контексте
    updateValue('Новое значение из компонента');
  };

  return (
    <div>
      <p>Значение из контекста: {sharedValue}</p>
      <button onClick={handleButtonClick}>Обновить значение</button>
    </div>
  );
};

export default MyComponent;
```

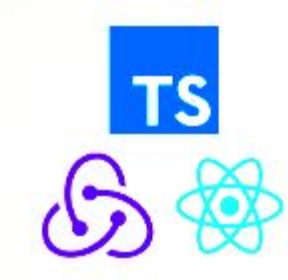
импортируем контекст

С помощью useContext получаем доступ к значению и функции для его изменения из компонента MyContextProvider

Когда значение в контексте обновляется в MyContextProvider, все компоненты, использующие useContext(MyContext), автоматически получают новое значение, и их перерисовывается.



Создание проекта React Redux



Установка

Рекомендуемый способ запуска новых приложений с помощью React и Redux Toolkit — использование официального шаблона

Redux Toolkit + TS для Vite

Ссылка на репозиторий:

https://github.com/reduxjs/redux-templates/tree/master/packages/vite-template-redux

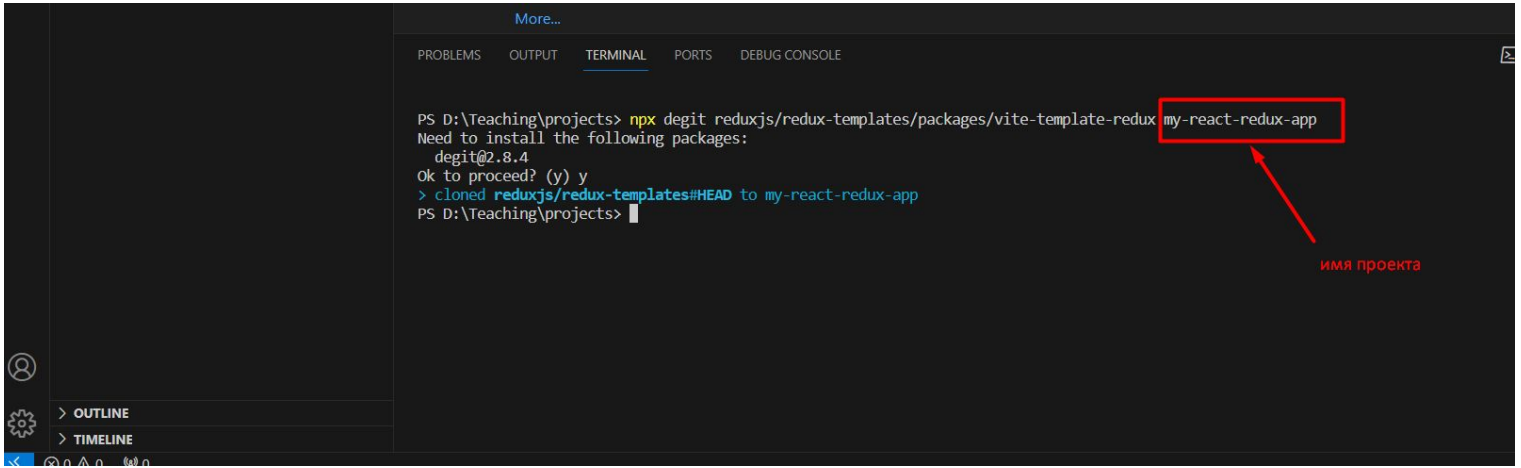
Примечание: Vite — это инструмент сборки, цель которого — обеспечить более быструю и экономичную разработку современных веб-проектов.

Установка

Шаг 1

Откройте VSCode, перейдите в папку, в которой будет лежать ваш проект. В терминале введите следующую команду

```
npx degit reduxjs/redux-templates/packages/vite-template-redux my-app
```



Примечание: у вас на компьютере уже должен быть установлен Node.js

Установка

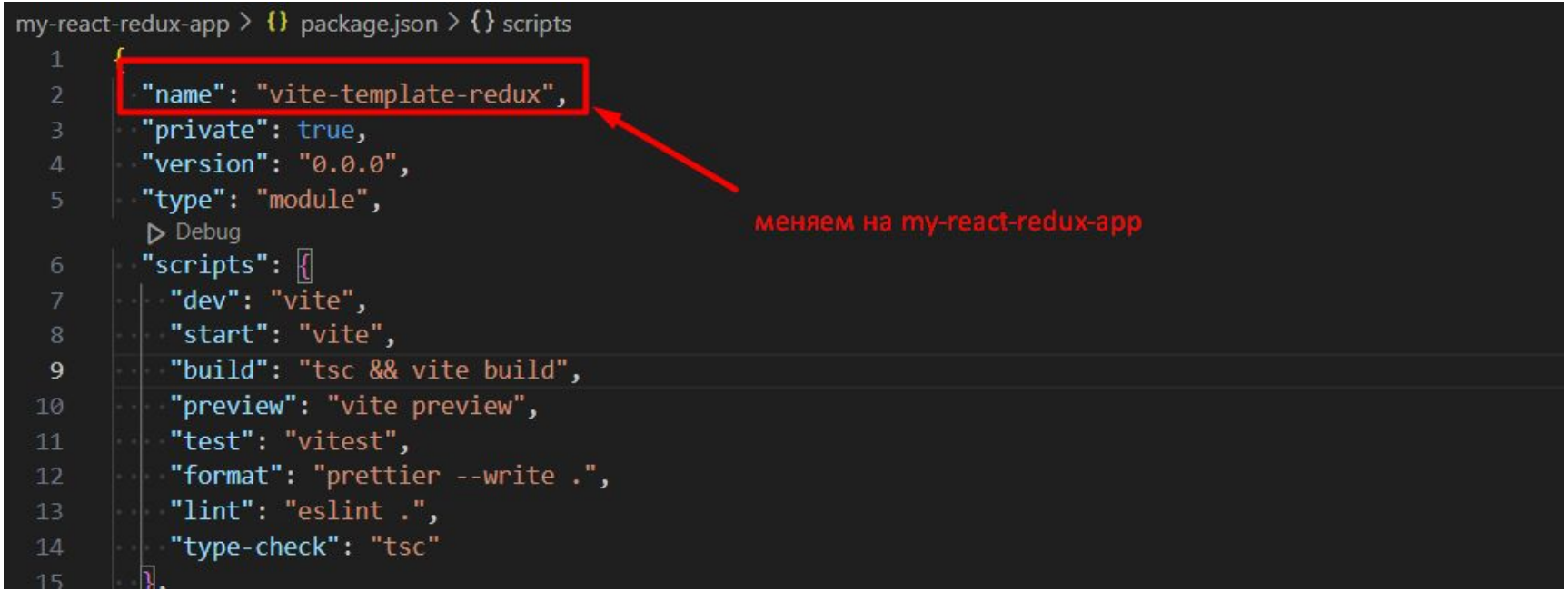
Шаг 2

В VSCode перейдите в папку вашего проекта

Установка

Шаг 3

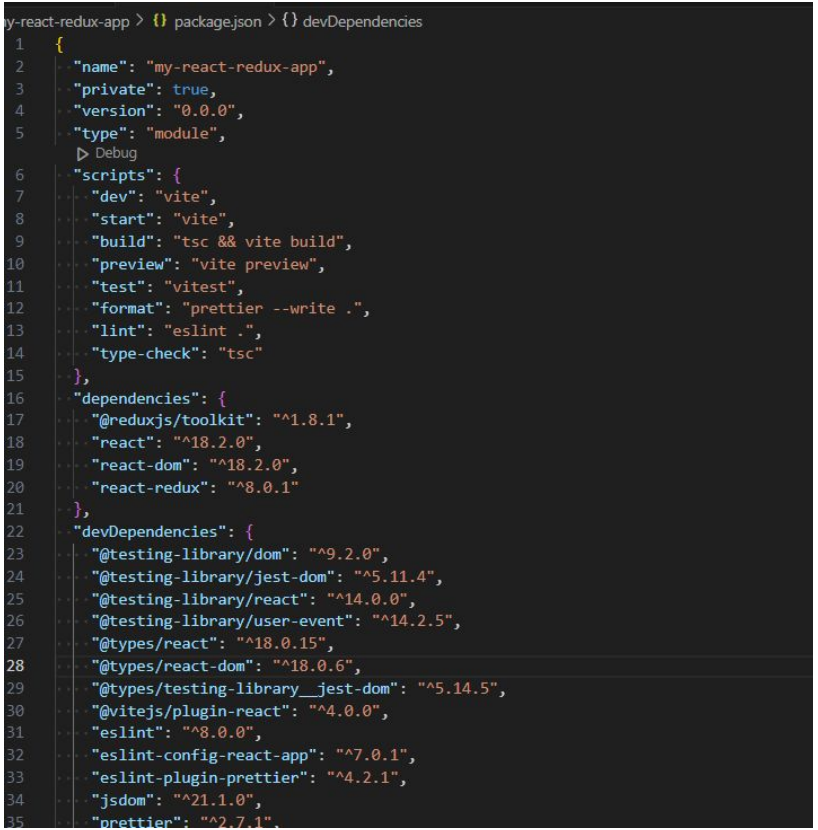
Открываем файл package.json и меняем название нашего проекта



Установка

Шаг 4

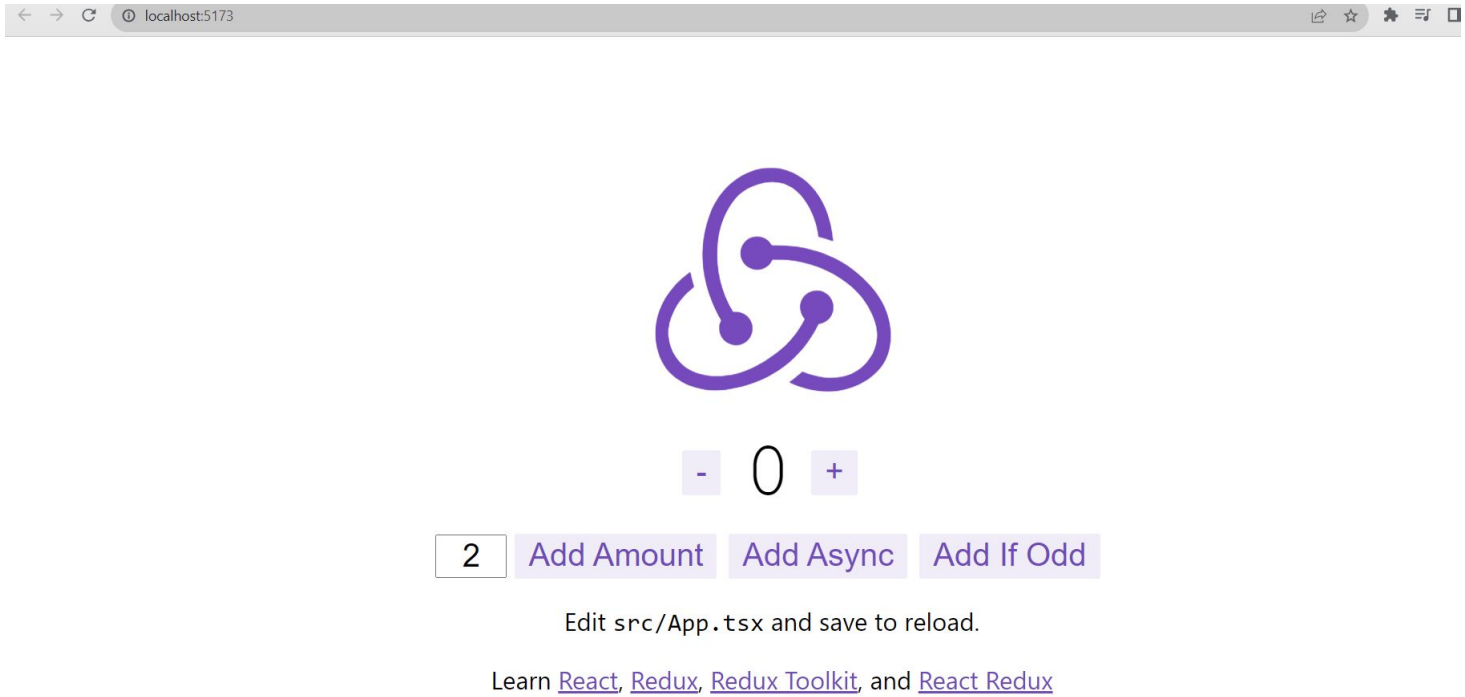
Устанавливаем все пакеты,
указанные в package.json с
помощью команды npm install.
Это может занять некоторое время



Установка

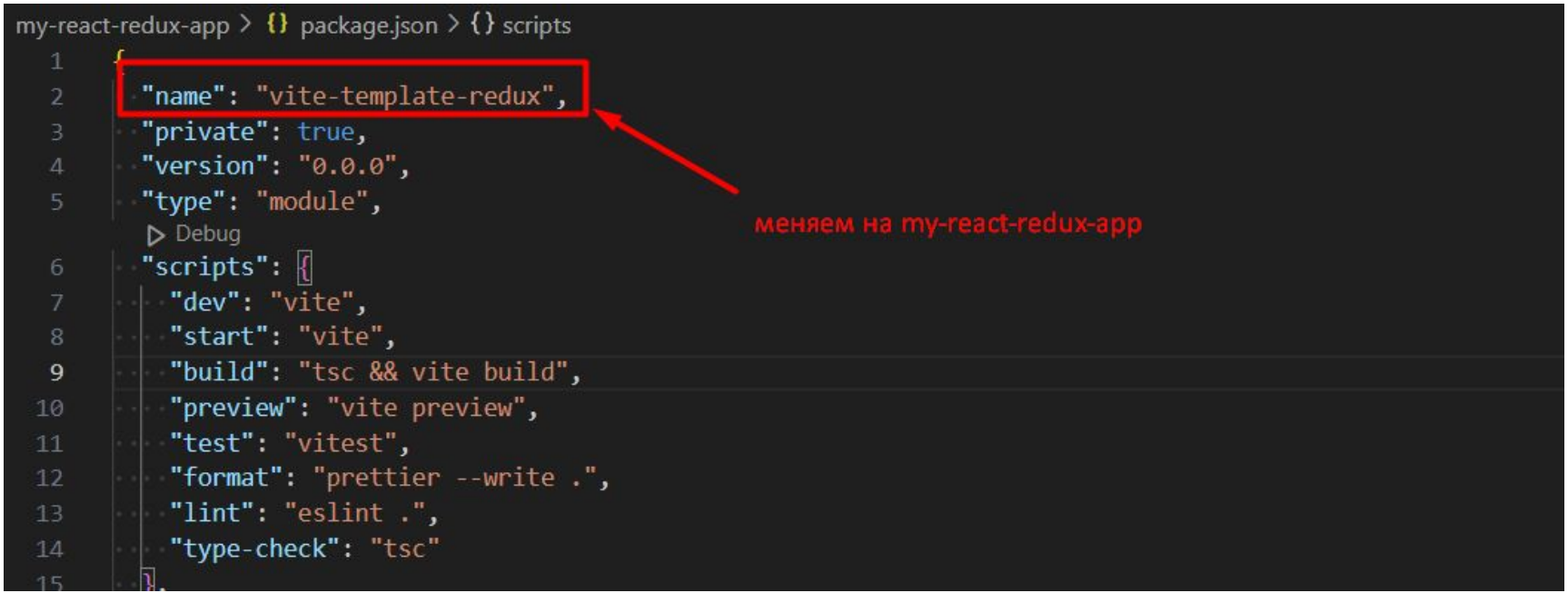
Шаг 5

Запускаем проект с помощью команды npm start. Как результат мы увидим уже созданный с помощью Redux Toolkit счётчик



Настройка

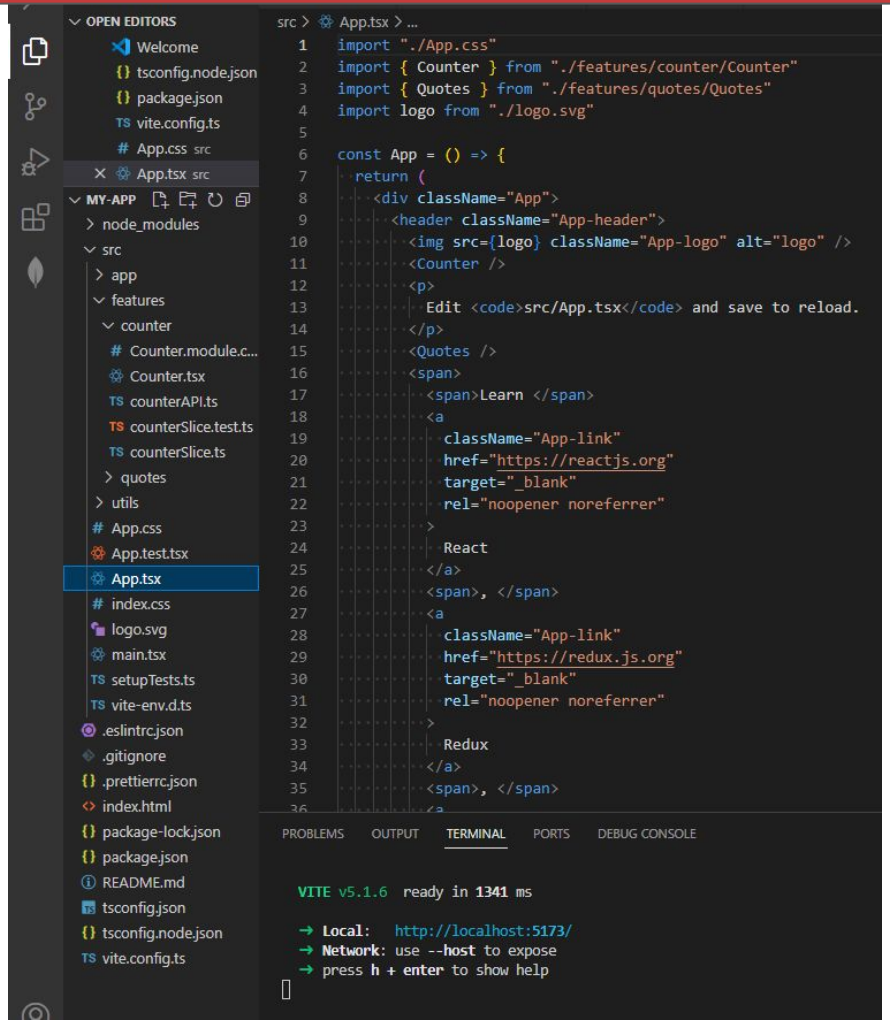
Открываем файл package.json и меняем название нашего проекта




```
{
  "name": "vite-template-redux",
  "private": true,
  "version": "0.0.0",
  "type": "module",
  ▷ Debug
  "scripts": {
    "dev": "vite",
    "start": "vite",
    "build": "tsc && vite build",
    "preview": "vite preview",
    "test": "vitest",
    "format": "prettier --write .",
    "lint": "eslint .",
    "type-check": "tsc"
  },
```

Структура проекта

- У нас появился новый файл с настройкой vite - vite.config.ts
- Созданный counter находится в папке features
- В папке app лежит файл с настройкой store и файл с вспомогательными хуками hooks.ts



Ваша новая IT-профессия – Ваш новый уровень жизни

Программирование с нуля в
немецкой школе AIT TR GmbH

AIT
Hi-tech team