Курс NextJS. Создание макетов и страниц

Д. Макаренков, к.т.н.

https://dmpsy.club

NextJS >= ReactJS + NodeJS

Целевая аудитория

Энтузиасты программирования на NextJS, желающие следовать современным принципам разработки full-stack Web-приложений

Магическая формула:

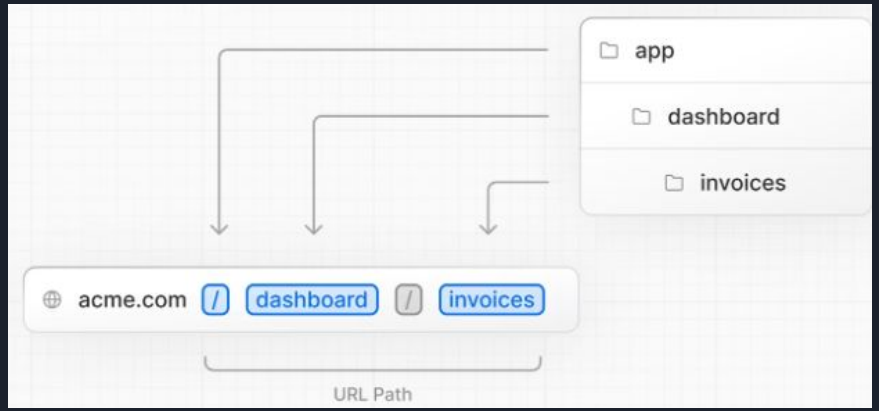
NextJS > = ReactJS (front-end) + NodeJS (back-end)

План работы

1	Nested routing	Вложенная маршрутизация
2	Creating the Dashboard page	Создание панели мониторинга
3	Creating Customers & Invoices pages	Создание страниц клиентов и счетов
4	Creating the dashboard layout	Создание общего макета для страниц
5	The role of Root layout	Роль корневого макета

Официальная версия от NextJS: https://nextjs.org/learn/dashboard-app/creating-layouts-and-pages

Вложенная маршрутизация (nested routing)



1 NextJS использует ***папки (folders)*** для создания ***вложенных маршрутов (nested routes)***.

2 Каждая папка представляет собой ***сегмент маршрутизации (route segment)***, соответствующий ***сегменту пути URL***

3 Каждая папка должна содержать файл страницы - ***page.tsx***

localhost:3000/dashboard/customers -> app/dashboard/customers/page.tsx # Leaf segment

localhost:3000/dashboard/invoices -> app/dashboard/invoices/page.tsx # Leaf segment

localhost:3000/dashboard -> app/dashboard/page.tsx # Route segment

localhost:3000/ -> app/page.tsx # Root segment

Вложенная маршрутизация (2)

Совместное размещение ресурсов (colocation):

NextJS резервирует имена файлов страниц (page.tsx), макетов (layout.tsx) и некоторых других файлов (ошибок, отсутствующих страниц и пр.), поэтому не возникает проблем с одновременным размещением остального контента в тех же директориях: отображаются лишь page.tsx, layout.tsx (при наличии) и папки / подпапки, их содержащие

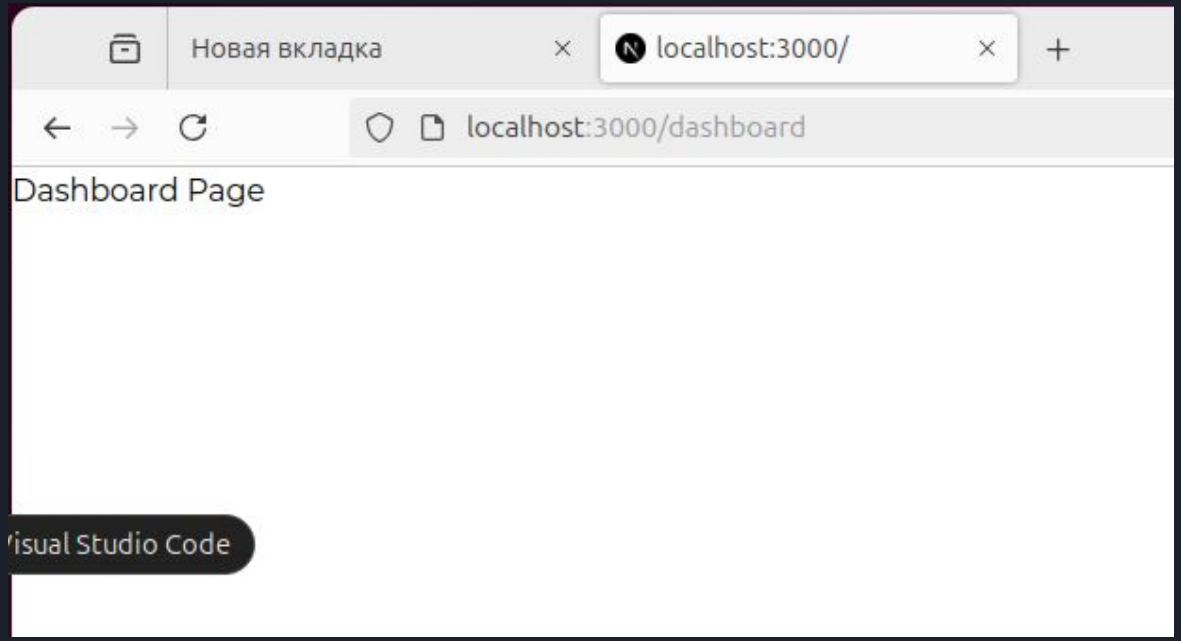
Создание панели мониторинга (Dashboard)

1	Create app/dashboard/page.tsx	Создайте папку /app/dashboard и page.tsx
		из командной строки:
		`cd app`
		`mkdir dashboard`
		`cd dashboard`
		`touch page.tsx`
		или средствами VSCode

2	Edit page.tsx	Отредактируйте /app/dashboard/page.tsx:
		
		`export default function Page() {`
		`  return <p>Dashboard Page</p>;`
		`}`

Создание панели мониторинга (2)

3	**Save changes and open Dashboard**	**Ctrl-S для сохранения изменений и откройте** **http://localhost:3000/app/dashboard/**



Создание страниц клиентов (Customers) и счетов (Invoices)

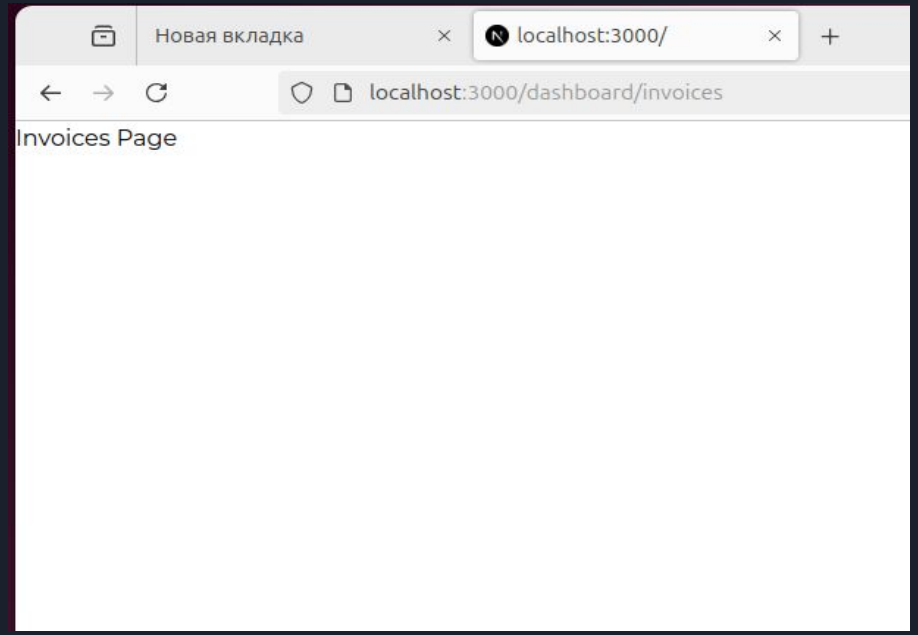
1	**Create**
	
	app/dashboard/invoices/page.tsx
	
	app/dashboard/customers/page.tsx
		Создайте папку /app/dashboard и файлы page.tsx из командной строки:
		`cd app/dashboard`
		`mkdir invoices`
		`mkdir customers`
		`cd invoices`
		`touch page.tsx`
		`touch ../customers/page.tsx`
		или средствами VSCode

2	**Edit app/dashboard/invoices/page.tsx**	**Отредактируйте /app/dashboard/invoices/page.tsx:**
		
		`export default function Page() {`
		`  return <p>Invoices Page</p>;`
		`}`

3	**Edit app/dashboard/customers/page.tsx**	**Отредактируйте /app/dashboard/customers/page.tsx:**
		
		`export default function Page() {`
		`  return <p>Customers Page</p>;`
		`}`

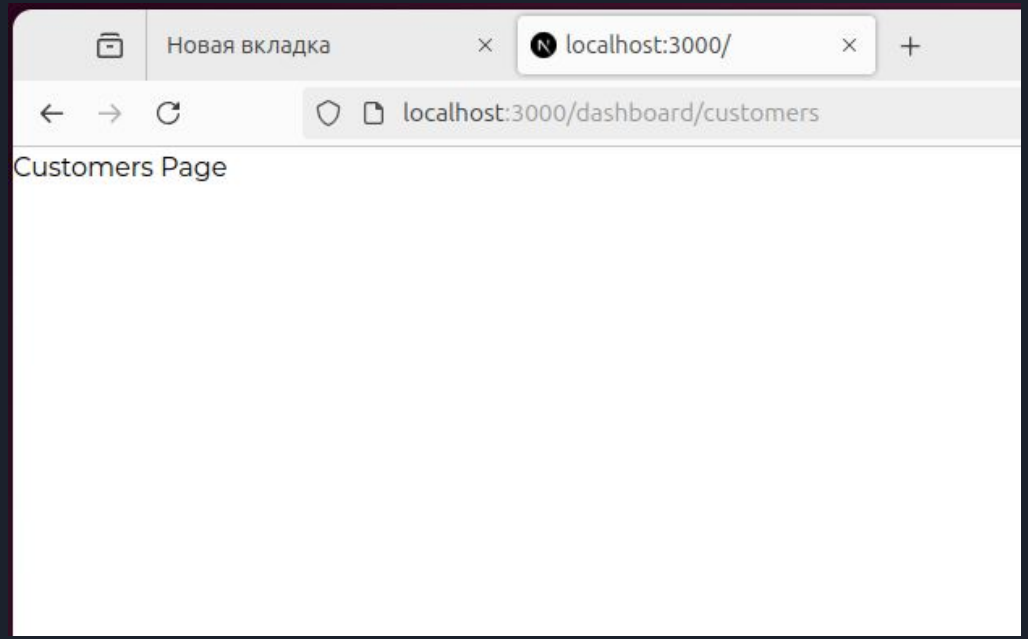
Создание страниц клиентов и счетов (2)

4	**Save changes and open Invoices**	**Ctrl-S для сохранения изменений и откройте http://localhost:3000/dashboard/invoices/**



Создание страниц клиентов и счетов (3)

5	**Save changes and open Customers**	**Ctrl-S для сохранения изменений и откройте http://localhost:3000/dashboard/customers/**



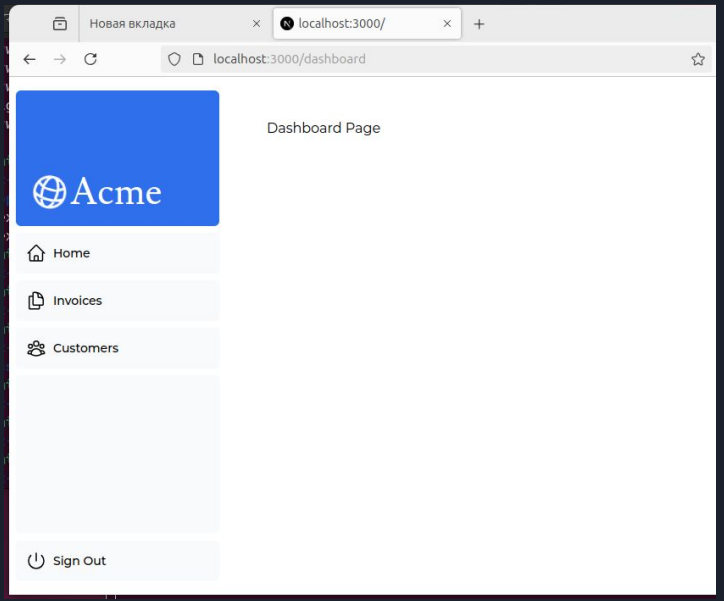
Создание общего макета (layout) для страниц

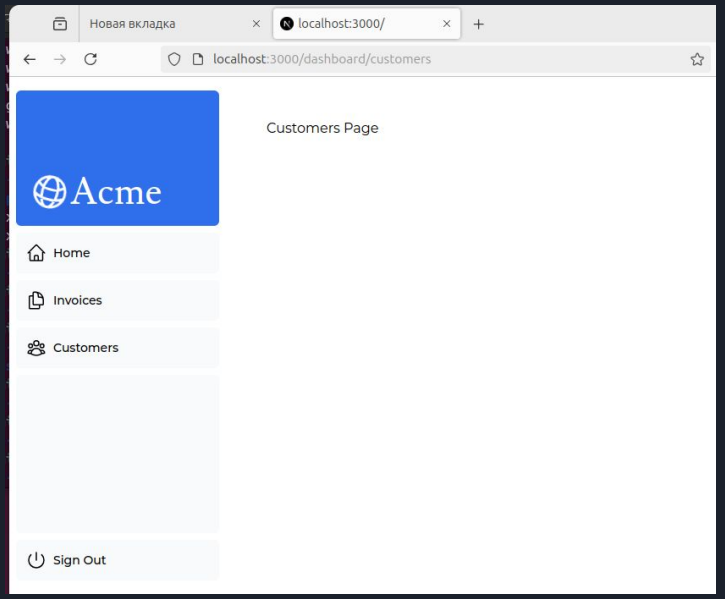
1	Create app/dashboard/layout.tsx	Создайте файл /app/dashboard/layout.tsx
		из командной строки:
		`cd app/dashboard`
		`touch layout.tsx`
		или средствами VSCode

2	Edit app/dashboard/layout.tsx	Отредактируйте /app/dashboard/layout.tsx:
		`import SideNav from '@/app/ui/dashboard/sidenav';`
		
		`export default function Layout({ children }: { children: React.ReactNode }) {`
		`  return (`
		`  <div className="flex h-screen flex-col md:flex-row md:overflow-hidden">`
		`  <div className="w-full flex-none md:w-64">`
		`  <SideNav />`
		`  </div>`
		`  <div className="flex-grow p-6 md:overflow-y-auto md:p-12">{children}</div>`
		`  </div>`
		`  );`
		`}`

Создание общего макета для страниц (2)

4	**Save changes and open Dashboard, Invoices, and Customers**	**Ctrl-S для сохранения изменений и откройте** http://localhost:3000/dashboard/ http://localhost:3000/dashboard/invoices/ http://localhost:3000/dashboard/customers/





Создание общего макета (layout) для страниц (3)

/app/dashboard/layout.tsx:

```
import SideNav from '@/app/ui/dashboard/sidenav';

export default function Layout({ children }: { children: React.ReactNode }) {
  return (
    <div className="flex h-screen flex-col md:flex-row md:overflow-hidden">
      <div className="w-full flex-none md:w-64">
        <SideNav />
      </div>
      <div className="flex-grow p-6 md:overflow-y-auto md:p-12">{children}</div>
    </div>
  );
}
```

1. Next.js использует файл layout.tsx для создания пользовательского интерфейса, который будет общим (здесь это компонент <SideNav />) для страниц page.tsx из папки dashboard и ее подпапок
2. Компонент <Layout /> получает prop children, который может быть страницей (page.tsx) или другим макетом (layout.tsx). Здесь children - страницы pages.tsx из папок dashboard, invoices и customers

Создание общего макета (layout) для страниц (4)

Частичная отрисовка (partial rendering):

При навигации между страницами с общим layout.tsx (dashboard, invoices, customers) layout.tsx не перерисовывается, что экономит ресурсы, улучшает производительность и внешний вид

Роль корневого макета (root layout)

/app/layout.tsx:

```
import '@/app/ui/global.css';
import { montserrat, lusitana } from '@/app/ui/fonts';

export default function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <html lang="en">
      <body className={`${montserrat.className} antialiased`}>{children}</body>
    </html>
  );
}
```

1. Корневой макет (root layout) является обязательным элементом (даже автоматически воссоздается после удаления)
2. Любой UI из корневого макета будет общим для всех страниц приложения: корневой макет обрамляет собой весь визуальный контент приложения
3. Используйте корневой макет для изменения тегов <html> и <body>, а также для добавления метаданных (обсудим в отдельной лекции) и шрифтов

Подведем итоги

1	**Научились создавать маршруты панели мониторинга (dashboard), используя вложенную маршрутизацию (nested routing)**
2	**Поняли роль папок и файлов при создании новых сегментов маршрута (route segments)**
3	**Создали макет (layout) для обрамления панели мониторинга (dashboard) и ее дочерних страниц (invoices, customers)**
4	**Изучили понятия совместной компоновки (colocation), частичной отрисовки (partial rendering) и корневого макета (root layout)**

Следующая лекция -
5. Navigating Between Pages / Навигация по страницам

Презентация доступна для скачивания здесь:

https://dmpsy.club/references/NextJS/lesson_004_creating_layouts_pages_rus.pdf

Поддержать автора: https://www.donationalerts.com/r/dmitrymak

https://dmpsy.club
postmaster@dmspy.club

https://www.donationalerts.com/r/dmitrymak