**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

# **Дисциплина:** Фронт-энд разработка

Отчет

Домашняя работа №5

Выполнил:

Игнатьев Алексей

К3440

Проверил:

Добряков Д. И.

Санкт-Петербург 2025 г.

**Задача**

Изучить основные команды пакетного менеджера NPM и научиться стартовать проект на Vue. Научиться работать с npm и vue на основе мануала: <https://docs.google.com/document/d/187UkgGNrcWqkb2aCGpkHTLgeozoElMqdVgVGMBOC9gk/edit?usp=sharing>.

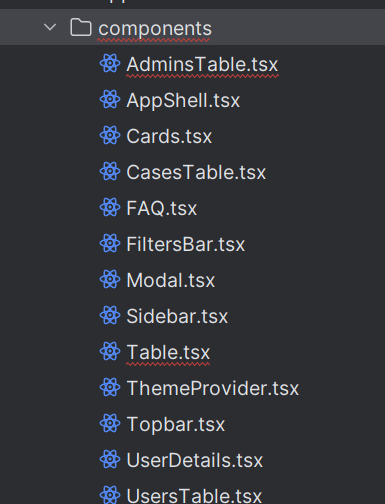
**Ход работы**

В качестве фреймворка был выбран Next.JS, создан проект на Next.JS согласно документации

npx create-next-app@latest

Добавление компонентов выполнялось вручную в папку компонентов.

Получившаяся структура компонентов приведена на скриншоте.



Базовый состав компонента на примере TopBar включает в себя реализацию самого компонента и экспорт его из модуля

"use client";  
import { useTheme } from "./ThemeProvider";  
  
export default function Topbar({ title }: { title?: string }) {  
 const { theme, toggle } = useTheme();  
 return (  
 <header className="relative z-10 border-b" style={{ background: "color-mix(in srgb, var(--card) 70%, transparent)", borderColor: "var(--sidebar-border)" }}>  
 <div className="h-14 flex items-center justify-between px-4">  
 <div className="flex items-center gap-3">  
 <div className="text-xl font-semibold">Банк</div>  
 <div className="text-muted">/</div>  
 <div className="text-xl font-semibold">{title || "Главная"}</div>  
 </div>  
 <ThemeSwitch theme={theme} onToggle={toggle} />  
 </div>  
 </header>  
 );  
}  
  
function ThemeSwitch({ theme, onToggle }: { theme: "light" | "dark"; onToggle: () => void }) {  
 const isDark = theme === "dark";  
 return (  
 <button  
 aria-label="Toggle theme"  
 onClick={onToggle}  
 className={`relative w-16 h-8 rounded-full transition-colors`} style={{ background: "var(--primary)" }}  
 >  
 <span className={`absolute inset-y-0 left-1 my-auto w-6 h-6 rounded-full bg-white shadow transition-transform duration-300 ${isDark ? "translate-x-8" : "translate-x-0"}`} />  
 <span className="absolute left-2 top-1/2 -translate-y-1/2 text-xs">{isDark ? "🌙" : "☀️"}</span>  
 <span className="absolute right-2 top-1/2 -translate-y-1/2 text-xs opacity-70">{isDark ? "" : ""}</span>  
 </button>  
 );  
}

Более сложные компоненты, например компонент талицы, содержит в своем коде условия в зависимости от состояния экрана. Пример – условие отображения дропдаун вьюшек

{senderDD.open && createPortal(  
 <HeaderDropdown pos={senderDD.pos} onClose={() => senderDD.setOpen(false)} portalRef={senderDD.panelRef}>  
 <div className="header-dd p-2 w-[260px]">  
 <div className="text-sm mb-2 font-medium">Отправитель</div>  
 <input className="ui-input w-full" placeholder="Имя" value={senderQ} onChange={(e) => setSenderQ(e.target.value)} />  
 <div className="mt-2 grid grid-cols-2 gap-2">  
 <button className="btn btn-danger w-full h-9" onClick={() => setSenderQ("")}>Сбросить</button>  
 <button className="btn btn-success w-full h-9" onClick={() => senderDD.setOpen(false)}>Сохранить</button>  
 </div>  
 </div>  
 </HeaderDropdown>, *document*.body)}

Для хранения сложных состояний в компонентах реализованы структуры. Пример – DropDown State

type DropdownState = {  
 open: boolean; setOpen: *React*.Dispatch<*React*.SetStateAction<boolean>>;  
 btnRef: *React*.RefObject<HTMLButtonElement>; panelRef: *React*.RefObject<HTMLDivElement>;  
 pos: { top: number; left: number; width: number };  
};

Для получения данных в компонентах реализованы стандартные функции Next JS

const [nameQ, setNameQ] = useState(filters.*nameQuery* || "");  
const [phoneQ, setPhoneQ] = useState(filters.*phoneQuery* || "");  
const [emailQ, setEmailQ] = useState(filters.*emailQuery* || "");  
const [statusSet, setStatusSet] = useState<Set<UserStatus>>(new Set(filters.*statuses* || []));  
const [dateFrom, setDateFrom] = useState<string | undefined>(filters.*dateFrom*);  
const [dateTo, setDateTo] = useState<string | undefined>(filters.*dateTo*);  
const [minCOM, setMinCOM] = useState<string>(filters.*minCOM* != null ? String(filters.minCOM) : "");  
const [maxCOM, setMaxCOM] = useState<string>(filters.*maxCOM* != null ? String(filters.maxCOM) : "");  
const [minTotal, setMinTotal] = useState<string>(filters.*minTotal* != null ? String(filters.minTotal) : "");  
const [maxTotal, setMaxTotal] = useState<string>(filters.*maxTotal* != null ? String(filters.maxTotal) : "");

**Вывод**

В рамках домашней работы на сайт был поделен на компоненты согласно стандартам Next.JS