

Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации

Ордена Трудового Красного Знамени федеральное государственное
бюджетное образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра «Сетевые информационные технологии и сервисы»

ПРОЕКТ
«учебные карточки из пдф: загрузка конспекта, резка на вопрос-ответ»
по дисциплине «Full Stack»

Выполнила: Студентка группы БВТ2303
Максимова Мария

Проверил:
Фильков Я.Д.

Москва, 2025 г.

1.

◆ **Лаб. №1 — Проектирование UI/UX и каркасный прототип**

Задача: показать, **как будет выглядеть приложение** и собрать «пустой каркас» без бизнес-логики.

Задачи:

UI/UX и каркасный прототип (React + FastAPI).

Функциональные элементы интерфейса:

1. Главная страница
2. Поле для загрузки PDF (кнопка выбора файла)
3. Список конспектов
4. Кнопка «Создать карточки»
5. Просмотр карточек
6. Список карточек с вопросами и ответами
7. Кнопка «Скачать» для всех карточек (JSON/PDF)

UX-сценарий:

1. Пользователь загружает PDF → сервер обрабатывает его → фронтенд показывает карточки



Фронтенд (React Vite)

Api.ts

```
export async function uploadPDF(file: File) {  
    const fd = new FormData();  
    fd.append("pdf", file);  
    const res = await fetch("/api/upload", { method: "POST", body: fd });  
    if (!res.ok) throw new Error("Upload failed");  
    return res.json();  
}  
  
export async function getDecks() {  
    const res = await fetch("/api/decks");  
}
```

```

    if (!res.ok) throw new Error("Failed to get decks");
    return res.json();
}

export async function createCards(filename: string) {
    const res = await
fetch(`/api/create_cards/${encodeURIComponent(filename)}`, { method: "POST"
});
    if (!res.ok) throw new Error("Failed to create cards");
    return res.json();
}

```

App.css

```

#root {
    max-width: 1280px;
    margin: 0 auto;
    padding: 2rem;
    text-align: center;
}

.logo {
    height: 6em;
    padding: 1.5em;
    will-change: filter;
    transition: filter 300ms;
}
.logo:hover {
    filter: drop-shadow(0 0 2em #646cffaa);
}
.logo.react:hover {
    filter: drop-shadow(0 0 2em #61dafbaa);
}

@keyframes logo-spin {
    from {
        transform: rotate(0deg);
    }
    to {
        transform: rotate(360deg);
    }
}

@media (prefers-reduced-motion: no-preference) {
    a:nth-of-type(2) .logo {
        animation: logo-spin infinite 20s linear;
    }
}

.card {
    padding: 2em;
}

.read-the-docs {
    color: #888;
}

```

App.tsx

```
import React, { useEffect, useState } from "react";
import { uploadPDF, getDecks, createCards } from "./api";

export default function MainPage() {
  const [file, setFile] = useState<File | null>(null);
  const [decks, setDecks] = useState<{name:string}[]>([]);
  const [cards, setCards] = useState<{question:string,answer:string}[]>([]);
  const [status, setStatus] = useState<string>("");

  useEffect(() => { loadDecks(); }, []);

  async function loadDecks() {
    try {
      const data = await getDecks();
      setDecks(data.decks || []);
    } catch (e) {
      console.error(e);
    }
  }

  async function onUpload() {
    if (!file) { alert("Выберите PDF"); return; }
    setStatus("Загружаю...");
    try {
      await uploadPDF(file);
      setStatus("Файл загружен");
      setFile(null);
      await loadDecks();
    } catch (e) {
      console.error(e);
      setStatus("Ошибка загрузки");
    }
  }

  async function onCreateCards(name:string) {
    setStatus("Генерация карточек...");
    try {
      const data = await createCards(name);
      setCards(data.cards || []);
      setStatus("Карточки готовы");
    } catch (e) {
      console.error(e);
      setStatus("Ошибка генерации");
    }
  }

  function downloadJSON() {
    const blob = new Blob([JSON.stringify(cards, null, 2)], {type: "application/json"});
    const url = URL.createObjectURL(blob);
    const a = document.createElement("a");
    a.href = url; a.download = "cards.json"; a.click();
    URL.revokeObjectURL(url);
  }

  return (
    <>
      <div className="header">
        <h1>Учебные карточки</h1>
        <div>
          <input type="file" accept=".pdf" onChange={e =>
```

```

setFile(e.target.files?.[0] ?? null) } />
    <button className="button primary" onClick={onUpload}>Загрузить
PDF</button>
    <span className="status">{status}</span>
</div>
</div>

<h2>Загруженные файлы:</h2>
<div>
    {decks.length === 0 && <div>Пока нет файлов</div>}
    {decks.map(d => (
        <div key={d.name} className="deck-item">
            <div>{d.name}</div>
            <div>
                <button className="button success" onClick={() =>
onCreateCards(d.name)}>Создать карточки</button>
            </div>
        </div>
    )));
</div>

<div className="cards">
    <h2>Карточки</h2>
    {cards.length === 0 && <div>Нет карточек (сгенерируйте их)</div>}
    {cards.map((c, i) => (
        <div key={i} className="card">
            <div className="question" onClick={(e)=> {
                const ans = (e.currentTarget.nextSibling as HTMLElement);
                if (ans) ans.classList.toggle("show");
            }}>{c.question}</div>
            <div className="answer">{c.answer}</div>
        </div>
    )));
    {cards.length>0 && <button className="button"
onClick={downloadJSON}>Скачать все (JSON)</button>}
    </div>
</>
);
}
}

```

main.tsx

```

import React from "react";
import { createRoot } from "react-dom/client";
import App from "./App";

const container = document.getElementById("root")!;
createRoot(container).render(<App />);

```

styles.css

```

:root {
    --bg: #f7f7f7;
    --card: #ffffff;
}

```

```

--accent: #4CAF50;
--muted: #666;
--primary: #2196F3;
}
body {
  font-family: Arial, sans-serif;
  background: var(--bg);
  margin: 0;
}
.container {
  max-width: 900px;
  margin: 40px auto;
  padding: 20px;
}
.header {
  display:flex;
  justify-content:space-between;
  align-items:center;
  gap:16px;
}
h1 { margin:0; color:#222; }
input[type="file"] { margin-right: 8px; }
.deck-item { background: var(--card); padding:12px; border-radius:8px;
margin-bottom:8px; display:flex; justify-content:space-between; align-
items:center; box-shadow:0 2px 6px rgba(0,0,0,0.06); }
.button { padding:8px 12px; border-radius:6px; border:none; cursor:pointer; }
.button.primary { background:var(--primary); color:white; }
.button.success { background:var(--accent); color:white; }
.cards { margin-top:20px; }
.card { background:var(--card); padding:12px; border-radius:6px; margin-
bottom:8px; }
.question { font-weight:700; cursor:pointer; }
.answer { display:none; margin-top:8px; color:var(--muted); }
.answer.show { display:block; }
.status { margin-left:8px; color:var(--muted); font-weight:600; }

```

Index.html

```

<!doctype html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Учебные карточки</title>
    <link rel="stylesheet" href="/styles.css" />
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.tsx"></script>
  </body>
</html>

```

Package.json

```
{
  "name": "frontend",
  "version": "0.0.0",
```

```
"type": "module",
"private": true,
"scripts": {
  "dev": "vite",
  "build": "vite build",
  "preview": "vite preview"
},
"dependencies": {
  "react": "^18.2.0",
  "react-dom": "^18.2.0"
},
"devDependencies": {
  "vite": "^5.4.20",
  "@vitejs/plugin-react": "^4.0.0",
  "typescript": "^5.2.2"
}
}
```

Vite.config.ts

```
import { defineConfig } from "vite";
import react from "@vitejs/plugin-react";

export default defineConfig({
  plugins: [react()],
  server: {
    port: 3000, // порт для фронтенда
    proxy: {
      "/api": "http://127.0.0.1:8000/" // проксирование запросов на бэкенд
    }
  }
});
```

tsconfig.app.json

```
{
  "compilerOptions": {
    "tsBuildInfoFile": "./node_modules/.tmp/tsconfig.app.tsbuildinfo",
    "target": "ES2022",
    "useDefineForClassFields": true,
    "lib": ["ES2022", "DOM", "DOM.Iterable"],
    "module": "ESNext",
    "types": ["vite/client"],
    "skipLibCheck": true,

    /* Bundler mode */
    "moduleResolution": "bundler",
    "allowImportingTsExtensions": true,
    "verbatimModuleSyntax": true,
    "moduleDetection": "force",
    "noEmit": true,
    "jsx": "react-jsx",

    /* Linting */
    "checkJs": true
  }
}
```

```
        "strict": true,
        "noUnusedLocals": true,
        "noUnusedParameters": true,
        "erasableSyntaxOnly": true,
        "noFallthroughCasesInSwitch": true,
        "noUncheckedSideEffectImports": true
    },
    "include": ["src"]
}
```

Tsconfig.json

```
{
  "compilerOptions": {
    "target": "ES2019",
    "module": "ESNext",
    "jsx": "react-jsx",
    "moduleResolution": "node",
    "strict": false,
    "esModuleInterop": true,
    "skipLibCheck": true,
    "forceConsistentCasingInFileNames": true,
    "resolveJsonModule": true,
    "isolatedModules": true
  },
  "include": ["src"]
}
```

Tsconfig.node.json

```
{
  "compilerOptions": {
    "tsBuildInfoFile": "./node_modules/.tmp/tsconfig.node.tsbuildinfo",
    "target": "ES2023",
    "lib": ["ES2023"],
    "module": "ESNext",
    "types": [],
    "skipLibCheck": true,

    /* Bundler mode */
    "moduleResolution": "bundler",
    "allowImportingTsExtensions": true,
    "verbatimModuleSyntax": true,
    "moduleDetection": "force",
    "noEmit": true,

    /* Linting */
    "strict": true,
    "noUnusedLocals": true,
    "noUnusedParameters": true,
    "erasableSyntaxOnly": true,
    "noFallthroughCasesInSwitch": true,
    "noUncheckedSideEffectImports": true
  },
  "include": ["vite.config.ts"]
}
```

Бэкенд (FastAPI)

Main.py

```
import os
import shutil
from fastapi import FastAPI, UploadFile, File, HTTPException
from fastapi.responses import JSONResponse, FileResponse
from fastapi.staticfiles import StaticFiles
from fastapi.middleware.cors import CORSMiddleware

BASE_DIR = os.path.dirname(__file__)
UPLOAD_DIR = os.path.join(BASE_DIR, "uploads")
os.makedirs(UPLOAD_DIR, exist_ok=True)

app = FastAPI(title="Flashcards from PDF (backend)")

# CORS для разработки (vite на 5173)
app.add_middleware(
    CORSMiddleware,
    allow_origins=["http://localhost:5173", "http://127.0.0.1:5173"],
    allow_methods=["*"],
    allow_headers=["*"],
    allow_credentials=True,
)

# ----- API -----

@app.post("/api/upload")
async def upload_pdf(pdf: UploadFile = File(...)):
    if not pdf.filename.lower().endswith(".pdf"):
        raise HTTPException(status_code=400, detail="Only PDF allowed")
    dest = os.path.join(UPLOAD_DIR, pdf.filename)
    with open(dest, "wb") as f:
        shutil.copyfileobj(pdf.file, f)
    return {"message": f"{pdf.filename} uploaded", "filename": pdf.filename}

@app.get("/api/decks")
def list_decks():
    files = sorted(os.listdir(UPLOAD_DIR))
    return {"decks": [{"name": f} for f in files]}

@app.post("/api/create_cards/{filename}")
def create_cards(filename: str):
    # ЗАГЛУШКА: возвращаем тестовые карточки для данного файла
    cards = [
        {"question": f"Вопрос 1 по {filename}", "answer": "Ответ 1"},
        {"question": f"Вопрос 2 по {filename}", "answer": "Ответ 2"},
    ]
    return {"cards": cards}

# ----- Static serving for production build -----
# When frontend is built into frontend/dist, FastAPI will serve it
dist_dir = os.path.join(BASE_DIR, "..", "frontend", "dist")
if os.path.isdir(dist_dir):
    # mount whole dist at root
    app.mount("/", StaticFiles(directory=dist_dir, html=True),
              name="frontend")
else:
```

```
# dev: expose a small message at root
@app.get("/")
def root():
    return {"message": "Frontend not built. Run frontend dev on :5173 or
build into frontend/dist."}
```

Как выглядит сайт:

<http://localhost:3000/>



-
- ◆ **Лаб. №2 — Настройка сервера и базовая маршрутизация (бэккартинка и фронт)**

Лабы:

1. Проектирование UI/UX и каркасный прототип (Vite, React TS)
2. Настройка сервера и базовая маршрутизация (бэк- картинка и фронт)
3. Вёрстка компонентов и настройка React Router
- 4 . Реализация CRUD-эндпоинтов
- 5 . Клиентская авторизация и работа с токенами
- 6 . Защита API: JWT и middleware
7. Интеграция фронтенда с бэкендом