**AI Tutor 平台 - 交付文档（开发文档 / 部署文档 / 用户手册）**

**版本：**v0.1.0（前端原型，无后端）  
**日期：**2025-08-21  
**移交对象：**下一任前端工程师 / 团队  
**编写人：**（自动生成自开发对话记录）

本文档用于把当前 AI Tutor 学习项目完整移交给新的负责人。内容包含：项目背景、开发环境、源码结构、运行与调试、功能说明、已知问题、改进建议、上线部署、常见故障排查，以及未来如何从前端原型平滑接入真实后端。

**一、项目概览**

AI Tutor 是一个基于 Next.js (App Router) + React + TypeScript + TailwindCSS + shadcn/ui + TanStack React Query 的前端原型，围绕“教学管理”场景搭建：班级、学生、作业、分析四大模块。数据暂时以 mocks.ts 作为内存数据源，刷新会重置。

**1.1 核心功能**

* 班级管理：查看班级列表、班级学生数量、作业数量等统计。
* 学生管理：CSV 导入/导出、创建/删除，支持按班级筛选、关键字搜索。
* 作业管理：创建作业、发布/设为草稿、删除、查看详情（提交情况、批改任务模拟）。
* 分析报表：成绩分布、提交状态饼图、班级平均分、批改趋势图；支持浅色/深色模式。

**1.2 技术栈与关键依赖**

* Next.js 15（App Router） / React 18 / TypeScript
* TailwindCSS（原子化样式，结合 CSS 变量做主题）
* shadcn/ui（基于 Radix UI 的可组合组件库）
* TanStack React Query（数据/请求状态管理）
* Recharts（可视化图表）

**1.3 运行效果与定位**

定位为“教学平台”的前端演示原型。所有数据来自前端 mocks，方便快速迭代 UI 与交互。后续接入真实后端时，只需在 features/\*\*/mocks.ts 中逐步替换为 API Client 实现，并在 React Query 的 queryFn/mutationFn 中调用即可。

**二、环境准备与安装**

**2.1 系统要求**

* 操作系统：Windows 10/11、macOS、Linux 均可。
* Node.js：18.x 或以上（建议 18 LTS）。
* 包管理器：npm（内置于 Node）。
* 浏览器：Edge / Chrome / Firefox / Safari 最新版。

**2.2 克隆与安装依赖**

git clone <your-repo-url>  
cd ai\_tutor  
npm install

**2.3 本地开发与访问**

npm run dev

默认访问地址：http://localhost:3000

**2.4 Windows PowerShell 脚本策略（如遇 npx 无法执行）**

Set-ExecutionPolicy RemoteSigned -Scope CurrentUser

*NOTE: 执行后重新打开 PowerShell。*

**三、项目结构与代码约定**

**3.1 目录结构（关键子树）**

src/  
 app/  
 layout.tsx # 全局布局（顶部导航、侧边栏、主题切换、ReactQueryProvider）  
 page.tsx # 首页  
 classes/ # 班级  
 page.tsx  
 students/ # 学生  
 page.tsx  
 assignments/ # 作业  
 page.tsx # 列表  
 new/page.tsx # 新建  
 [id]/page.tsx # 详情  
 analytics/ # 分析报表  
 page.tsx  
 components/ui/ # shadcn/ui 组件（button/input/dialog/select/...）  
 features/  
 classes/  
 mocks.ts # 班级数据与方法  
 ClassesTable.tsx  
 students/  
 mocks.ts # 学生数据与方法（导入/导出/删除/查询）  
 StudentsTable.tsx  
 assignments/  
 mocks.ts # 作业数据与方法（创建/发布/草稿/删除/查询）  
 AssignmentsTable.tsx  
 submissions/  
 SubmissionsTable.tsx # 作业详情中提交列表（如已实现）  
 utils/  
 apiClient.ts # 预留：真实 API 封装位置（当前未使用）  
 styles/  
 globals.css # Tailwind 基础、主题 CSS 变量

**3.2 代码风格与约定**

* 所有 React 组件使用函数组件与 Hooks。
* 组件顶端显式 `import \* as React from "react"` 以避免某些构建场景下 `React is not defined`。
* UI 统一使用 shadcn/ui 提供的原子组件（二次封装见 components/ui/\*）。
* 数据获取/修改统一走 TanStack React Query；数据源目前来自各 features/\*\*/mocks.ts。
* 路由文件命名遵循 Next.js App Router 规范：page.tsx / layout.tsx / [id]/page.tsx。
* 样式优先使用 Tailwind 原子类；主题色通过 CSS 变量覆盖，支持浅色/深色模式。

**四、主题系统与设计令牌**

在 styles/globals.css 中定义了 OKLCH/OKLAB 颜色空间的 CSS 变量，包含背景、前景、主色、边框、组件色等。通过在 :root 与 .dark 下设置不同变量值，实现主题切换（Header 的“浅色/深色”按钮）。

**4.1 常用变量（节选）**

:root {  
 --background: oklch(1 0 0);  
 --foreground: oklch(0.145 0 0);  
 --primary: oklch(0.205 0 0);  
 --primary-foreground: oklch(0.985 0 0);  
 --muted: oklch(0.97 0 0);  
 --muted-foreground: oklch(0.556 0 0);  
 --destructive: oklch(0.577 0.245 27.325);  
 --border: oklch(0.922 0 0);  
 --input: oklch(0.922 0 0);  
 --ring: oklch(0.708 0 0);  
 /\* ... \*/  
}  
.dark {  
 --background: oklch(0.145 0 0);  
 --foreground: oklch(0.985 0 0);  
 --primary: oklch(0.85 0 0);  
 --primary-foreground: oklch(0.15 0 0);  
 /\* 其余变量按需提高对比度 \*/  
}

*NOTE: 深色模式下确保文字与背景对比度 ≥ WCAG AA（建议 >= 4.5），图表请使用更高彩度与更亮的描边。*

**五、功能说明与页面流**

**5.1 班级（/classes）**

* 查看班级列表：班级名称、学生数量、作业数量、创建时间。
* 点击班级名称可跳转到学生页并携带 classId 筛选（如已实现）。

**5.2 学生（/students）**

* 筛选：按班级筛选、关键字检索（姓名/邮箱）。
* 导出 CSV：把当前筛选结果导出为 CSV。
* 导入 CSV：选择 CSV 文件（UTF-8），表头需包含“姓名,班级,邮箱”。程序自动跳过 header 行。
* 新建学生：在弹窗中选择班级、输入姓名与邮箱；创建后自动刷新列表。
* 删除：二次确认弹窗（shadcn/ui AlertDialog），确认后刷新列表。

**5.3 作业（/assignments）**

* 查看作业列表：标题、班级、提交数、状态（草稿/已发布）、截止时间、创建时间。
* 新建作业：/assignments/new，选择班级、输入标题、截止时间。默认草稿。
* 发布/设为草稿：在列表中点击操作按钮，状态互转；使用 toast 给出结果反馈。
* 删除作业：二次确认后永久删除。
* 跳转详情：点击标题跳转到 /assignments/[id] 查看提交与批改（如已实现 SubmissionsTable）。

**5.4 分析（/analytics）**

* Card 指标：学生数、班级数、作业数、已批改份数、平均分（基于 mocks 计算）。
* 图表：成绩分布柱状图、提交状态饼图、班级平均分柱状图、批改趋势折线图。
* 下拉筛选：按班级查看；未选中时展示汇总。
* 深色模式：请确保颜色对比足够（详见改进建议）。

**六、数据结构与 mocks 约定**

**6.1 Type 约定（示例）**

export type ClassItem = {  
 id: string;  
 name: string;  
 studentsCount: number;  
 assignmentsCount: number;  
 createdAt: string; // ISO  
};  
  
export type Student = {  
 id: string;  
 name: string;  
 email: string;  
 classId: string;  
 score?: number;  
 joinedAt: string;  
};  
  
export type Assignment = {  
 id: string;  
 title: string;  
 classId: string;  
 status: 'draft' | 'published';  
 dueAt: string;  
 createdAt: string;  
 submissionsCount: number;  
};

**6.2 mocks.ts 暴露的方法（按模块）**

* classes/mocks.ts：fetchClasses({keyword,page,pageSize})
* students/mocks.ts：fetchStudents、createStudent、deleteStudent、importStudentsCsv、exportStudentsCsv
* assignments/mocks.ts：fetchAssignments、createAssignment、deleteAssignment、publishAssignment、draftAssignment、getAssignmentById

**6.3 CSV 格式与样例**

姓名,班级,邮箱  
张三,Class 1,zhangsan@school.test  
李四,Class 1,lisi@school.test  
王五,Class 2,wangwu@school.test

*NOTE: 导入时会跳过第一行表头；若首行不为表头，请在导入对话框勾选“第一行不是表头”开关（如实现）。*

**七、运行、调试与质量保障**

**7.1 常用脚本**

npm run dev # 本地开发  
npm run build # 生产构建  
npm run start # 启动生产服务

**7.2 常见报错与修复指引**

* `React is not defined`：组件顶部添加 `import \* as React from "react"`。
* `Hydration failed`：避免 SSR 与 CSR 渲染不一致（不要在渲染时用 Math.random/Date.now）。
* `Module not found: '@/components/ui'`：应从 `'@/components/ui/button'` 等具体组件路径导入。
* PowerShell 禁止执行 npx：执行 `Set-ExecutionPolicy RemoteSigned -Scope CurrentUser`。

**7.3 UI 与无障碍建议（特别是深色模式）**

* 确保文字/背景对比度 >= 4.5，按钮悬停态提高 10-15% 亮度或降低透明度。
* 图表配色：在深色下使用高饱和色（蓝、紫、青、橙），并添加白色或浅色描边。
* 在图例上显示数值与占比，悬浮 tooltip 展示详情，提升可读性。

**八、部署说明**

**8.1 Vercel（推荐）**

1. 将代码推送到 Git 仓库（GitHub/GitLab）。
2. 登录 Vercel，New Project → Import，选择仓库。
3. Framework 选择 Next.js，Build Command 使用默认。
4. 点击 Deploy，几分钟后获得生产地址。

**8.2 Netlify / 静态托管（不推荐）**

Next.js 带 SSR/路由特性，建议使用 Vercel 这类支持的托管平台。

**8.3 Docker（可选）**

# Dockerfile  
FROM node:18-alpine  
WORKDIR /app  
COPY . .  
RUN npm install && npm run build  
EXPOSE 3000  
CMD ["npm","start"]

**九、已知问题与改进计划**

**9.1 已知问题（来自当前开发反馈）**

* 分析页“按班级统计”时，没有按“作业”维度细分（同一班级多作业被统一统计）。
* 图表配色较为单一（黑白灰），深色模式下对比度偏低。
* 作业页：草稿状态的“发布”按钮在个别版本中不触发（需统一 mocks 与 UI 回调）。
* 深色模式下部分组件（尤其图表与输入框）可读性不足。

**9.2 解决思路 / 任务拆解**

* （分析维度）补充数据模型：Submission 记录关联 assignmentId，计算时先过滤 assignmentId，再做区间聚合。
* （图表主题）封装 ChartTheme：在浅/深色分别提供调色板与网格、坐标轴样式；统一 Legend/Tooltip。
* （发布草稿）在 AssignmentsTable 中统一调用 `publishAssignment/draftAssignment` 并在成功后 `invalidateQueries(['assignments'])`。
* （深色主题）调高 --foreground、--muted-foreground 的对比度；为图表增加描边与高亮 hover。

**十、从 mocks 迁移到真实后端的步骤**

1. 在 utils/apiClient.ts 中建立统一的请求封装（基于 fetch/axios），处理基址、鉴权、错误。
2. 把 features/\*\*/mocks.ts 中的方法逐步改成调用 apiClient；保留与 mocks 相同的签名与返回结构。
3. 在 React Query 的 queryFn/mutationFn 中替换为真实方法；开启重试/缓存时间等策略。
4. 为 CSV 导入提供后端接口（支持 Excel：前端用 SheetJS 解析成 rows，然后调用批量创建 API）。
5. 引入持久化存储（LocalStorage/IndexedDB）减少刷新数据丢失，或使用后端持久化。

**十一、交付清单与验收标准**

**11.1 交付清单**

* 完整源码：含 app、features、components、styles、utils 等目录。
* 本 Word 文档（开发文档 / 部署文档 / 用户手册 / 迁移指南）。
* 可运行的开发环境：`npm install && npm run dev` 可直接访问。

**11.2 验收标准（可操作）**

* 本地启动不报错；首页、班级、学生、作业、分析五个页面可访问。
* 学生 CSV 导入/导出、创建/删除可用，并有二次确认与 toast。
* 作业列表的创建/发布/设为草稿/删除可用；点击标题可进入详情页（如已实现）。
* 分析页可正常渲染四类图表；切换深色模式无明显可读性问题。

**十二、FAQ / 排错清单**

* Q: 运行时报 `Module not found` 指向 `@/components/ui`？  
  A: 需要导入具体组件文件，如 `@/components/ui/button`。
* Q: CSV 导入把表头也导入了？  
  A: 在解析时默认跳过第一行；请检查文件是否 UTF-8 编码且第一行为中文表头。
* Q: 点击“发布”无效？  
  A: 确认 AssignmentsTable 中的 `onPublish` 已绑定到 `publishAssignment`，并在成功后 `invalidateQueries`。
* Q: 深色模式下图表不可读？  
  A: 提高颜色饱和度与亮度、增加描边、网格线改为半透明浅色，参见第 9 章建议。

**十三、附录**

**A. Git 提交规范（建议）**

* feat: 新功能
* fix: 修复 bug
* refactor: 重构（不改变外部行为）
* style: 样式或格式调整（无逻辑变更）
* docs: 文档变更
* chore: 构建、脚手架、依赖等变更

**B. 代码片段速查**

// React Query 使用示例  
const { data, isLoading } = useQuery({  
 queryKey: ['students', params],  
 queryFn: () => fetchStudents(params),  
 staleTime: 5 \* 60 \* 1000,  
});  
  
// 变更后刷新  
const qc = useQueryClient();  
await createStudent(payload);  
await qc.invalidateQueries({ predicate: q => Array.isArray(q.queryKey) && q.queryKey[0] === 'students' });