AI Tutor — 技术方案（v1.0）

> 依据《AI Tutor GUI — Figma 设计文稿（v1.0）》与 PRD，给出可落地的端到端技术方案，覆盖架构、技术栈、数据模型、API、任务编排、导入导出、报表/统计、运维与验收。

---

0. 目标与非目标

目标

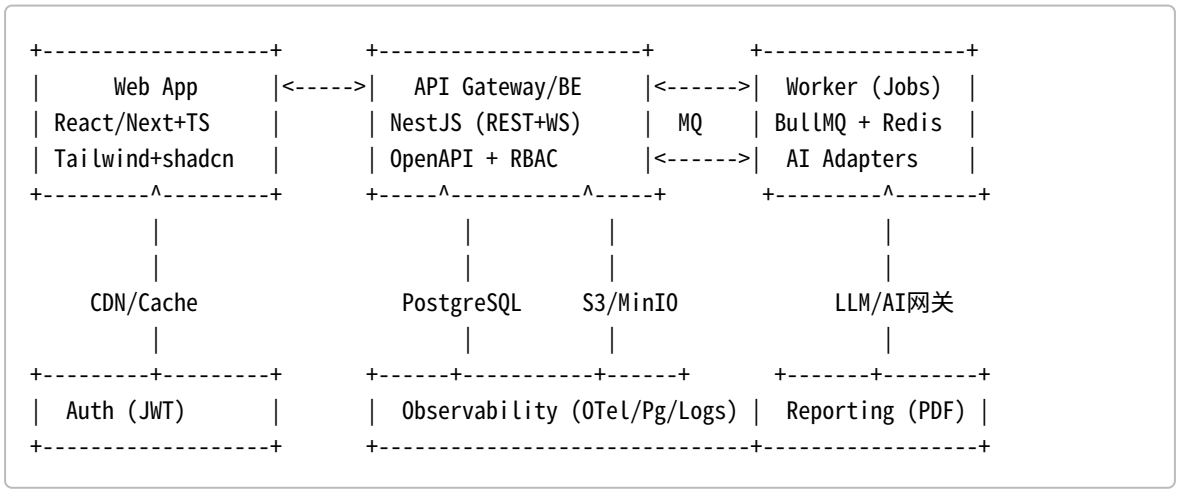
- 交付一套可扩展的教学管理与 AI 批改平台：系统配置、班级/学生/作业、数据导入导出、批量批改、反馈报告与统计分析。
- FE/BE 类型共享、可测试、可观测、易部署。
- 支持多租户（按机构/学校隔离）与多环境（dev/stage/prod）。

非目标

- 不在本期实现复杂权限（细粒度到字段级）与 SSO；先实现角色级 RBAC。
- 不提供移动端独立应用；提供响应式适配与关键流可达。

---

1. 总体架构



- **部署**：容器化（Docker），K8s 或 Compose（dev）。
- **可观测**：OpenTelemetry（Trace/Metrics/Logs）→ Prometheus + Grafana；Sentry 前后端错误上报。

---

2. 技术栈与项目结构

前端（Web）

- 框架：React + TypeScript（建议 Next.js App Router）

- UI：Tailwind CSS、shadcn/ui（Radix primitives）、Lucide 图标
- 状态：TanStack Query（服务端状态）+ Zustand（本地 UI 状态）
- 表单：react-hook-form + zod（校验）
- 表格/图表：TanStack Table、Recharts（或 ECharts）
- 国际化：i18next（zh-CN/en-US）
- 构建：Vite/Next 构建（按选型）
- 质量：ESLint、Prettier、Stylelint、Husky+lint-staged

后端（API/Worker）

- 运行时：Node.js + TypeScript；框架：NestJS（REST/WebSocket）
- ORM：Prisma（PostgreSQL）
- 任务队列：BullMQ（Redis）
- 存储：PostgreSQL（主）、MinIO/S3（对象）、Redis（缓存/队列）
- 文档：OpenAPI 3（Swagger），自动生成客户端 SDK
- 报表：Puppeteer（HTML→PDF）
- 导入导出：CSV/JSON（csv-parse/papaparse），预检+批处理

仓库结构（Monorepo，Turborepo/Nx）

```
apps/
  web/            # 前端
  api/            # NestJS API
  worker/         # 批处理/AI任务
packages/
  ui/             # 组件库（与Figma tokens对齐）
  shared/         # 通用类型/工具（OpenAPI类型、zod schema）
  config/         # ESLint/TS/Prettier/Stylelint 统一配置
infrastructure/   # IaC 模板（Terraform/K8s清单）
```

---

3. 设计令牌与主题

- 来源：Figma Variables（Color/Type/Space/Radius/Shadow/Motion/Breakpoint）
- 出口：Style Dictionary → `:root { --color-primary-500: #3B82F6; ... }`
- FE 使用：Tailwind config 中注入 CSS var；shadcn 组件主题化；暗色主题预留。

---

4. 数据模型（ERD 摘要）

```
Tenant( id, name )
User( id, tenantId, email, name, role )  # role: ADMIN/TEACHER/TA/VIEWER
Class( id, tenantId, name, createdAt )
Student( id, tenantId, classId, name, studentNo, createdAt )
Assignment( id, tenantId, classId, title, description, status, createdAt )
KnowledgeTag( id, tenantId, name )
AssignmentTag( assignmentId, tagId )
Submission( id, tenantId, assignmentId, studentId, contentUri, score, status, aiVersion,
```

```
createdAt )
Job( id, tenantId, type, payload, status, progress, createdAt, finishedAt )
ImportBatch( id, tenantId, type, fileUri, mapping, total, success, failed, reportUri,
createdAt )
Report( id, tenantId, assignmentId, type, fileUri, createdAt )  # feedback / analytics
Setting( id, tenantId, key, value )  # 例如 api.gateway.url
```

- 说明：所有业务表均带 `tenantId` 以支持多租户隔离；读写均加租户过滤。

Prisma Schema 片段

```
model Class {
  id         String   @id @default(cuid())
  tenantId   String
  name       String   @unique(map: "u_class_name_tenant")
  createdAt  DateTime @default(now())
  students   Student[]
  assignments Assignment[]
  @@index([tenantId])
}

model Student {
  id         String   @id @default(cuid())
  tenantId   String
  classId    String
  name       String
  studentNo  String
  createdAt  DateTime @default(now())
  @@index([tenantId, classId])
}
```

---

5. 认证与授权（Auth/RBAC）

- 登录：机构级账号体系（本期可复用现有用户中心或本地账号），JWT（Access+Refresh）。
- 授权：RBAC（ADMIN/TEACHER/TA/VIEWER）。
- 租户：通过域名/请求头携带 `X-Tenant` 或登录态绑定 `tenantId`。
- 审计：关键操作落库（删除、导入、发布作业、生成报告）。

---

6. API 设计（REST，OpenAPI）

命名：`/api/v1`

示例端点

- 系统设置
- `GET /settings` 读取租户设置（含 `api.gateway.url`）

- `POST /settings/test-connection` 测试网关连接，返回 `ok/latency/log`
- 班级
- `GET /classes` 列表（分页/关键词）
- `POST /classes` 创建
- `GET /classes/:id` 详情（含学生/作业统计）
- `DELETE /classes/:id` 删除（软删可选）
- 学生
- `GET /students?classId=...` 列表
- `POST /students` 创建
- `POST /students/import` 启动导入（返回 `batchId`）
- `GET /imports/:batchId` 导入进度与结果（可下载错误明细）
- 作业/提交
- `GET /assignments`
- `POST /assignments`
- `POST /assignments/:id/publish`
- `GET /assignments/:id/submissions`
- `POST /assignments/:id/grade/batch` 启动批改任务（返回 `jobId`）
- `GET /jobs/:jobId` 查询进度/错误明细
- 报告与统计
- `POST /assignments/:id/report/feedback` 生成反馈报告（PDF）
- `POST /assignments/:id/report/analytics` 生成统计（图表+CSV）
- `GET /reports/:id` 下载
- 导出
- `POST /export` 导出（类型/范围/格式）

契约：OpenAPI 3 生成 `@api-sdk`（web/worker 共享 typings）。

---

7. 导入/导出流水线（与 UI 的“字段映射/校验预览/结果反馈”对齐）

流程

1. 前端上传 CSV/JSON → S3（预签名 URL）
2. `POST /students/import` → 新建 ImportBatch，入队 BullMQ 任务
3. Worker 拉取文件 → 字段映射（UI 提供的映射表）→ 校验（zod）→ 去重（studentNo/tenantId）
4. 批量写库（事务+分批）→ 生成错误明细 CSV（上传 S3）
5. 更新进度与统计（成功/失败/跳过）
6. 前端轮询 `GET /imports/:batchId`，展示结果并提供“下载错误明细”

校验规则

- 必填：`name, studentNo`
- 唯一：`studentNo` 在 `tenantId` 范围内唯一
- 关联：`classId` 存在性检查；或允许通过 `className` 映射

导出

- 由后端流式生成 CSV/JSON（大数据分页 cursor），结果直传（`text/csv`）或生成文件链接。

---

8. 批量批改与 AI 适配（Assignments-BatchGrading）

触发：`POST /assignments/:id/grade/batch` → 创建 Job，按提交记录拆分子任务。

调度

- 队列：`grade:default`（并发 N），失败重试（指数退避，最多 3 次）
- 任务粒度：以 `Submission` 为单位；任务隔离错误不影响整体
- 进度：完成量/总量、失败计数、当前任务 ID

AI 适配层

- 通过 `api.gateway.url` 与外部 LLM/批改服务通讯（HTTP/gRPC）。
- 接口约定：`POST /grade` → 输入（prompt/评分标准/答案/上下文），输出（score/rubric/comments/knowledgeTags）
- 版本记录：写入 `Submission.aiVersion`

防并发/幂等

- 同一作业在“批改中”状态时拒绝再次触发；或合并请求。
- 以 `assignmentId` + `tenantId` 建立分布式锁（Redis）。

---

9. 报告与统计

反馈报告（PDF）

- 模板：服务端 SSR（Nunjucks/Handlebars → HTML），Puppeteer 渲染 A4 PDF，上传 S3。
- 内容：作业元信息、总评、分项评分、典型错误、改进建议。

统计分析

- 指标：平均/中位、分布（直方/箱线）、按知识点正确率、Top 错题。
- 实时计算：SQL 汇总（窗口函数）+ 缓存（Redis，TTL）。
- 导出：CSV/PNG（图表快照）。

---

10. 系统设置与连通性测试

- `Setting(key=value)` 存储租户级配置。
- 连通性：后端发起探测（HEAD/GET），记录延迟、HTTP 状态与部分响应体，返回给前端在弹窗展示。

---

11. 安全与合规

- **接口安全**：JWT + CORS 白名单；速率限制（per IP/token）；输入校验（zod/class-validator）。
- **文件安全**：大小/类型白名单；可选 ClamAV 扫描；预签名上传（最小暴露原则）。
- **数据安全**：租户隔离（`tenantId` 强制过滤）；软删与审计日志；备份策略（Pg 日志归档）。

- **隐私**：敏感数据（学生信息）脱敏导出；最小化存储。
- **OWASP**：XSS/CSRF/SQLi 防护（HTTP-only cookie 可选；ORM 绑定变量）。

---

12. 可观测性（Observability）

- Trace：OpenTelemetry SDK（web/api/worker），逻辑链路：前端操作 → API → DB/Queue → AI 网关。
- Metrics：API QPS、P95 延迟、队列堆积、导入成功率、批改时长分布。
- Logs：结构化（pino），按租户打标签，日志采集至 Loki/ELK。

---

13. 性能与前端体验

- 性能预算：LCP ≤ 2.5s、交互延迟 ≤ 100ms、首屏 < 150KB JS（不含框架）
- 策略：路由级代码拆分、组件级懒加载、表格虚拟滚动、图片/图标 Sprite、HTTP 缓存（ETag/Cache-Control）。
- 列表：服务端分页 + 客户端缓存（React Query 缓存 5min、后台刷新）。

---

14. 国际化与无障碍

- i18n：命名空间 `common/classes/students/assignments/data/settings`；提取脚本校验漏翻。
- a11y：焦点环、ARIA、语义标签；对比度与键盘操作用例（表格、弹窗、菜单）。

---

15. 测试策略

- 单元：web（RTL/Jest）、api（Jest）、worker（Jest）
- 合同：Pact（前端与 API 的契约测试）或 OpenAPI schema 校验
- 端到端：Playwright（关键流：创建班级/导入学生/批改/报告生成）
- 性能：k6（API 压测）、Lighthouse CI（前端）

---

16. DevOps 与部署

- CI：GitHub Actions（安装→构建→测试→lint→docker build→推送镜像）
- CD：Argo CD/Helm（K8s）或 Compose（stage/dev）。
- 环境变量：`DATABASE_URL`、`REDIS_URL`、`S3_*`、`JWT_SECRET`、`AI_GATEWAY_*`。
- IaC：Terraform（RDS/S3/EKS/Redis），按环境隔离。
- 备份与回滚：数据库每日快照；应用蓝绿/金丝雀发布。

---

17. 风险与缓解

- **AI 依赖不稳定** → 网关多活/重试/降级（暂存为“待批改”并提示）
- **导入大文件** → 分片上传、流式解析、内存上限、批量事务分段
- **长耗时任务** → 队列并发/优先级、心跳与超时、断点续跑

- **多租户越权** → 强制 `tenantId` 作用域中间件 + 审计

---

18. 交付里程碑（建议 4 批次并行）

1. **基础设施与骨架**（week 1-2）
2. Monorepo、CI、Auth/RBAC 骨架、设计令牌接入
3. **核心 CRUD/导入导出**（week 3-4）
4. Classes/Students/Assignments、导入导出全链路
5. **批改与报告/统计**（week 5-6）
6. 队列/AI 适配、PDF 报告、统计图表与导出
7. **硬化与验收**（week 7）
8. a11y/i18n、观测/压测、安全加固、UAT & Bugfix

---

19. Figma → 技术映射表（节选）

Figma 画面	前端组件/模式	主要 API	备注
Classes-List	DataTable/Filter Bar	`GET /classes`、`POST /classes`、`POST /export`	批量选择/导出
Students-Import	FileUploader/Mapping	`POST /students/import`、`GET /imports/:id`	结果下载链接
Assignments-List	DataTable/Tags	`GET /assignments`、`POST /assignments/:id/grade/batch`	状态流转
BatchGrading	Progress/Toast	`GET /jobs/:id`	轮询/WS 可选
FeedbackReport	PDF Viewer	`POST /assignments/:id/report/feedback`、`GET /reports/:id`	S3 直链
Analytics	Charts/Tabs	`POST /assignments/:id/report/analytics`	导出 CSV/PNG
Settings-API	Form/Alert	`GET /settings`、`POST /settings/test-connection`	延迟/日志显示

---

20. 验收标准（与设计验收清单对齐）

- 所有表单具备同步/异步校验与错误提示；所有列表具备分页/筛选/导出。
- 导入流程具备：模板下载→上传→映射→校验预览→结果反馈（含错误明细下载）。
- 批改流程具备：触发→进度可见→失败可重试→完成后可生成报告与统计。
- 报告可生成 PDF；统计可导出 CSV/PNG；均可按租户隔离存储与访问。
- 可观测指标达标；安全扫描通过；性能预算满足。

---

21. 附录：示例请求/响应

导入（启动）

```
POST /api/v1/students/import
{
  "fileUri": "s3://bucket/imports/students-20250815.csv",
  "mapping": { "name": "姓名", "studentNo": "学号", "classId": "班级ID" }
}
→ 200 { "batchId": "imp_123" }
```

批改（查询进度）

```
GET /api/v1/jobs/grade_imp_456
→ 200 { "status": "running", "progress": 62, "failed": 3 }
```

报告（生成并获取）

```
POST /api/v1/assignments/asg_001/report/feedback
→ 200 { "reportId": "rep_789", "fileUri": "s3://.../rep_789.pdf" }
```

结束语

以上方案与 Figma 设计逐项对齐，交付路径清晰，支持后续横向扩展（题库、考试、课堂互动等）。如需，我可以继续输出 **OpenAPI YAML**、**Prisma 全量 schema** 与 **K8s 部署清单** 以直接落地开发。