

# USTC 数据库学习/考试系统项目文档

## 1. 项目概述

- 背景：**面向数据库课程的教学、练习与考试管理，支持分布式授课和多样化题型。
- 目标：**提供统一的课程资源管理、在线学习、考试监考、成绩分析与教学反馈平台。
- 系统定位：**B/S 架构，前端 Vue 3 单页应用，后端 FastAPI 服务化，PostgreSQL 存储数据，可扩展为多节点部署。

## 2. 核心功能

- 用户管理：**注册、登录、找回密码、角色授权（管理员/教师/学生）。
- 课程管理：**课程创建、班级分组、资源发布、学员管理。
- 知识学习：**课程内容展示、富文本/Markdown 讲义、视频/附件、学习进度追踪。
- 题库管理：**选择题、填空题、编程题、SQL 运行题；标签、难度与版本控制。
- 考试中心：**考试编排、随机出题策略、时间/权限控制、监考（防刷题、防切屏）。
- 提交与批阅：**自动判题（客观题+SQL sandboxes）、教师人工批阅、评语录入。
- 成绩分析：**个人成绩单、班级统计、题目难度分析、导出报表。
- 通知与消息：**系统公告、考试提醒、成绩推送，可对接邮箱/短信。
- 系统管理：**参数配置、审计日志、数据备份、第三方认证。

## 3. 技术栈与架构

层级	方案	说明
前端	Vue 3 + Vite + TypeScript, Pinia, Vue Router, Element Plus	单页应用，组件化开发；Pinia 统一状态，Axios 请求封装。
后端	FastAPI + Python 3.11, SQLAlchemy 2.x, Pydantic v2, Alembic	REST 接口，JWT 鉴权；ORM + 异步数据库；数据库迁移。
数据库	PostgreSQL 15	支持 JSONB、全文搜索、CTE，适合考试统计与日



## 5. 后端设计

### 5.1 服务模块

- `auth`：JWT/OAuth2，短信/邮箱验证码，权限模型（RBAC）。
- `users`：个人信息、角色管理、学习档案。
- `courses`：课程 CRUD、资源、章节、班级。
- `content`：富文本、附件、视频元数据。
- `question_bank`：题目管理、标签、版本。
- `exam`：考试计划、策略、监考、反作弊。
- `submission`：答卷记录、评分、评语。
- `analytics`：成绩统计、题目难度、学习行为。
- `notifications`：站内信、邮件任务。
- `system`：配置项、审计日志、数据导出。

### 5.2 数据库模型（示例）

表名	关键字段
<code>users</code>	<code>id</code> , <code>username</code> , <code>password_hash</code> , <code>role</code> , <code>email</code> , <code>status</code>
<code>user_profiles</code>	<code>user_id</code> , <code>real_name</code> , <code>student_id</code> , <code>mobile</code> , <code>avatar_url</code>
<code>courses</code>	<code>id</code> , <code>name</code> , <code>code</code> , <code>teacher_id</code> , <code>description</code> , <code>visibility</code>
<code>course_members</code>	<code>id</code> , <code>course_id</code> , <code>user_id</code> , <code>role</code> , <code>group_tag</code>
<code>chapters</code>	<code>id</code> , <code>course_id</code> , <code>title</code> , <code>order</code> , <code>content</code>
<code>learning_progress</code>	<code>id</code> , <code>course_id</code> , <code>user_id</code> , <code>chapter_id</code> , <code>status</code> , <code>last_viewed_at</code>
<code>questions</code>	<code>id</code> , <code>type</code> , <code>content</code> , <code>options</code> , <code>answer</code> , <code>tags</code> , <code>difficulty</code> , <code>owner_id</code>
<code>exams</code>	<code>id</code> , <code>course_id</code> , <code>title</code> , <code>start_at</code> , <code>end_at</code> , <code>duration</code> , <code>mode</code> , <code>status</code>
<code>exam_papers</code>	<code>id</code> , <code>exam_id</code> , <code>question_id</code> , <code>score</code> , <code>section</code> , <code>order</code> , <code>random_seed</code>
<code>exam_access_rules</code>	<code>id</code> , <code>exam_id</code> , <code>limit_ip</code> , <code>limit_device</code> , <code>anti_cheat</code>

submissions	id, exam_id, user_id, submitted_at, score, status
submission_answers	id, submission_id, question_id, answer, score, feedback
attachments	id, owner_type, owner_id, url, content_type, size
notifications	id, channel, title, payload, status, sent_at

- 索引建议: questions.tags 使用 GIN; submissions.exam\_id\_user\_id 复合唯一; learning\_progress 通过 course\_id+user\_id 优化查询。

## 5.3 API 规范

- RESTful + JSON; 使用 /api/v1.
- OAuth2 Password + Refresh Token; Header Authorization: Bearer <token>.
- 全局响应 { code, message, data, trace\_id }.
- 错误码: 1xxx 认证/权限, 2xxx 业务, 5xxx 系统。
- OpenAPI 文档通过 FastAPI 自带 /docs /redoc。
- 速率限制: fastapi-limiter + Redis。

### 5.3.1 API 示例

模块	方法 & 路径	描述
Auth	POST /api/v1/auth/login	用户登录, 返回 Access/Refresh Token
Auth	POST /api/v1/auth/refresh	刷新 Token
Users	GET /api/v1/users/me	获取当前用户信息
Courses	POST /api/v1/courses (教师/管理员)	创建课程
Courses	GET /api/v1/courses/{id}	课程详情
Question Bank	POST /api/v1/questions	创建题目
Question Bank	GET /api/v1/questions?tags=sql&difficulty=hard	题库查询

Exams	POST /api/v1/exams	创建考试，设定策略
Exams	POST /api/v1/exams/{id}/publish	发布考试
Exams	GET /api/v1/exams/{id}/paper (学生)	拉取试卷（含随机题）
Submission	POST /api/v1/exams/{id}/submit	提交答案
Submission	GET /api/v1/submissions/{id}	查看答卷与评分
Analytics	GET /api/v1/analytics/exams/{id}	考试分析
Notifications	POST /api/v1/notifications	发送系统通知

## 5.4 权限/角色

- **Admin**: 系统配置、全局管理、用户审批。
- **Teacher**: 课程、题库、考试、评分、通知。
- **Student**: 学习、考试、查看成绩。
- RBAC 表结构: roles, permissions, role\_permissions, user\_roles.

## 5.5 业务流程（示例：在线考试）

1. 教师创建考试 → 选择题库/策略 → 发布。
2. 学生在考试时间内访问 → 获取 JWT 验证 + 防切屏监控。
3. 前端定时保存答题进度；后端记录 submission 状态。
4. 考试结束自动收卷，触发异步评分任务。
5. 分数生成后通知学生，教师可查看统计与错题。

# 6. 部署与环境

## 6.1 开发环境

```
# 后端
python -m venv .venv
source .venv/bin/activate
pip install -r requirements/dev.txt
cp server/.env.example server/.env
alembic upgrade head
uvicorn app.main:app --reload

# 前端
cd web
pnpm install
pnpm dev
```

## 6.2 测试/生产环境

- 使用 Docker Compose:
  - frontend : Vite build → Nginx 静态托管。
  - backend : Gunicorn/Uvicorn workers (4 workers, async workers)。
  - db : PostgreSQL + 持久卷 + 自动备份。
  - redis : 缓存 & Celery broker。
  - flower : Celery 监控。
- 生产建议使用 Kubernetes 或云托管 (ACK/EKS/GKE) ; Nginx Ingress + Cert-Manager 实现 HTTPS。
- 缓存配置、限流和 WAF 保护登录/考试接口。

## 6.3 配置项

- .env 示例:

```
APP_ENV=development
DATABASE_URL=postgresql+asyncpg://user:pass@db:5432/ustc
REDIS_URL=redis://redis:6379/0
JWT_SECRET=...
JWT_EXPIRE=900
REFRESH_EXPIRE=604800
STORAGE_ENDPOINT=https://oss.xxx
```

- 使用 pydantic-settings 管理配置, 支持环境变量/文件。

## 7. 测试与质量保障

- 前端: Vitest + Vue Testing Library; Cypress 做端到端。
- 后端: Pytest + httpx/pytest-asyncio; 覆盖 auth/考试逻辑; 使用 test DB。
- CI/CD: GitHub Actions/GitLab CI
  - Lint (ESLint、Stylelint、mypy/ruff)
  - 单元测试
  - 构建镜像 & 推送
- 静态分析: 前端 TypeScript 严格模式, 后端开启 ruff、mypy, pre-commit。
- 安全: SAST/Dependabot 监控依赖; JWT/密码加盐; SQL sandbox 使用隔离容器。

## 8. 运维与监控

- 日志: 后端使用 structlog / loguru 输出 JSON; ElasticStack 或 Loki 收集。
- 监控: Prometheus + Grafana, 暴露 FastAPI Metrics、DB 指标。
- 错误追踪: Sentry (前后端)。

- 备份：PostgreSQL 每日全量 + WAL；对象存储版本化。
- 灾备：只读副本、冷备份；考试期间建议双活部署。
- 审计：记录关键操作（改卷、加分、权限变更），满足课程管理审计需求。

## 9. 路线图

---

1. MVP (Sprint 1-2)：用户认证、课程管理、题库基本 CRUD、简单考试发布/提交。
2. Sprint 3-4：自动判题、成绩分析、消息通知、前端 UI 完善。
3. Sprint 5+：实时监考、编程题在线判题 (Docker sandbox)、多租户支持、移动端适配。
4. 长期：AI 辅助出题与推荐、学习轨迹个性分析、对接学校统一身份认证。

## 10. 附录

---

- 编码规范：前端遵循 ESLint + Prettier；后端遵循 PEP8，建议以 `src/` 模式组织。
- 文档：使用 `docs/` + VitePress/Docsify；OpenAPI 自动生成接口文档。
- 示例账号：
  - 管理员：admin / Admin@123
  - 教师：teacher01 / Teacher@123
  - 学生：student01 / Student@123