技术架构（**UniApp 多端 + Node.js/NestJS 后端 + MySQL 数据库 + 微信支付 v3**）和四阶段功能需求（**家长提交 → 自动生成缴费单 → 微信支付 → 回执**），来做一份**详细的任务分解与开发计划**。

我会按 **模块 → 任务 → 负责人角色 → 预计工期 → 依赖关系** 的方式列出，并且分成 **前端 / 后端 / 数据库 / 支付集成 / 测试与上线** 五大部分，方便你直接放进项目管理工具（如 Jira、TAPD、禅道）执行。

**一、总体阶段划分**

| **阶段** | **目标** | **主要产出** |
| --- | --- | --- |
| 阶段 1 | 环境与基础设施搭建 | 代码仓库、开发环境、数据库初始化 |
| 阶段 2 | 核心功能开发 | 表单提交、订单生成、支付调用、回调处理 |
| 阶段 3 | 系统联调与测试 | 前后端联调、支付沙箱测试、数据一致性验证 |
| 阶段 4 | 上线与运维 | 正式环境部署、监控与应急预案 |

**二、任务分解**

**1. 前端（UniApp 多端）**

| **任务** | **说明** | **负责人** | **工期** | **依赖** |
| --- | --- | --- | --- | --- |
| 项目初始化 | 创建 UniApp 项目，配置多端编译（微信小程序 + H5） | 前端 | 1 天 | 无 |
| UI 设计与组件库 | 表单、确认页、支付结果页、历史记录页 | 前端/UI | 3 天 | 无 |
| 表单录入页 | 姓名、性别、手机号1~3、学校、年级、班级选择器 | 前端 | 2 天 | API: 获取学校/年级/班级 |
| 确认页 | 展示录入信息 + 协议勾选 | 前端 | 1 天 | 表单页 |
| 支付页 | 展示缴费金额（380 元）+ 调起支付 | 前端 | 2 天 | API: 生成订单 & 获取支付参数 |
| 支付结果页 | 成功/失败提示 + 回执下载 | 前端 | 1 天 | API: 查询订单状态 |
| 微信 JSAPI/H5 支付集成 | 小程序端 wx.requestPayment，H5 端 WeixinJSBridge.invoke | 前端 | 2 天 | 微信支付参数 API |

**2. 后端（Node.js + NestJS）**

| **任务** | **说明** | **负责人** | **工期** | **依赖** |
| --- | --- | --- | --- | --- |
| 项目初始化 | NestJS 项目结构、配置文件、环境变量 | 后端 | 1 天 | 无 |
| 数据库连接与ORM配置 | TypeORM/Prisma 连接 MySQL，实体映射 | 后端 | 1 天 | 数据库 |
| API: 获取学校/年级/班级 | 提供选择器数据 | 后端 | 1 天 | 数据库基础数据 |
| API: 表单提交 | 写入 activation\_submission，自动建/匹配主数据 | 后端 | 2 天 | 数据库 |
| API: 生成缴费单 | 固定金额 380 元，写入 payment\_order & payment\_order\_item | 后端 | 1 天 | 表单提交 |
| API: 微信统一下单 | 调用微信支付 v3，写入 wechat\_payment | 后端 | 2 天 | 微信商户配置 |
| API: 微信支付回调 | 验签、更新订单状态、生成 payment\_receipt | 后端 | 2 天 | 微信支付 |
| API: 查询订单状态 | 返回支付状态与回执信息 | 后端 | 1 天 | 数据库 |

**3. 数据库（MySQL）**

| **任务** | **说明** | **负责人** | **工期** | **依赖** |
| --- | --- | --- | --- | --- |
| 初始化数据库 | 创建数据库实例、用户、权限 | DBA | 0.5 天 | 无 |
| 建表脚本执行 | 按设计说明书创建所有表 | DBA | 1 天 | 设计文档 |
| 索引与外键配置 | 添加唯一约束、外键、索引 | DBA | 0.5 天 | 建表完成 |
| 基础数据导入 | 学校、年级、班级、行政区划 | DBA | 1 天 | 数据源 |

**4. 支付集成**

| **任务** | **说明** | **负责人** | **工期** | **依赖** |
| --- | --- | --- | --- | --- |
| 微信商户配置 | 申请/配置 AppID、商户号、APIv3 密钥、证书 | 运维 | 1 天 | 无 |
| 支付沙箱环境 | 配置沙箱商户，测试支付流程 | 后端 | 1 天 | 商户配置 |
| JSAPI 支付调试 | 小程序端支付调试 | 前端/后端 | 1 天 | API: 微信统一下单 |
| H5 支付调试 | 公众号网页支付调试 | 前端/后端 | 1 天 | API: 微信统一下单 |
| 回调验签与幂等 | 确保回调安全与重复调用处理 | 后端 | 1 天 | 微信回调 |

**5. 测试与上线**

| **任务** | **说明** | **负责人** | **工期** | **依赖** |
| --- | --- | --- | --- | --- |
| 单元测试 | 前后端核心逻辑单测 | 前端/后端 | 2 天 | 开发完成 |
| 联调测试 | 前端调用后端 API，支付全链路测试 | 全栈 | 2 天 | 单元测试 |
| UAT 用户验收 | 学校方试用，收集反馈 | 产品/测试 | 2 天 | 联调完成 |
| 正式部署 | 部署到生产环境（HTTPS 域名、小程序发布） | 运维 | 1 天 | UAT 通过 |
| 监控与告警 | 接口监控、支付成功率监控 | 运维 | 1 天 | 上线完成 |

**三、时间规划（建议）**

假设团队 3~4 人并行开发，整体工期可控在 **3 周左右**：

* **第 1 周**：环境搭建 + 数据库初始化 + 前后端项目初始化 + UI 设计
* **第 2 周**：核心功能开发（表单提交、订单生成、支付调用、回调处理）
* **第 3 周**：联调测试 + UAT + 上线部署

**四、关键风险与应对**

* **支付回调延迟/丢失** → 增加定时任务主动查询订单状态
* **多端兼容问题** → 提前在微信开发者工具和多浏览器测试
* **数据一致性** → 关键表操作加事务，支付回调加幂等锁
* **性能瓶颈** → 缴费高峰期提前压测，优化索引