# 客户移动端 - 移动端设计文档

**文档版本**: V1.0  
**日期**: 2025年2月21日  
**作者**: Grok 3 (xAI)  
**目标读者**: 移动端开发团队、技术负责人

## 1. 概述

### 1.1 目标

实现“EduInsight AI”系统中客户移动端的功能，提供学生、教师和家长三个角色入口，支持 iOS 和 Android 平台。功能包括：

* **学生端**: 学习进度、作业提交、资源中心、智能助手与激励机制。
* **教师端**: 仪表盘、教学优化、资源贡献。
* **家长端**: 反馈提交、学习报告、指导工具。

### 1.2 技术选型

* **框架**: Flutter（跨平台开发，支持 iOS/Android）。
* **状态管理**: Provider（轻量、适用于Flutter）。
* **网络请求**: Dio（支持拦截器和缓存）。
* **本地存储**: SharedPreferences（轻量数据）、Hive（离线缓存）。
* **插件**:
  + **flutter\_ocr\_sdk**: 作业拍照OCR识别。
  + **video\_player**: 播放共享资源视频。
  + **push\_notification**: 推送支持。
* **构建工具**: Flutter CLI + Fastlane（自动化打包）。

### 1.3 设计原则

* **跨平台**: 单一代码适配 iOS 和 Android。
* **直观**: 简洁界面，角色动态切换。
* **离线支持**: 关键数据缓存，离线可用。
* **性能**: 流畅动画，快速响应。

## 2. 功能设计

### 2.1 总体页面结构

text

WrapCopy

[启动页] -> [登录页] -> [角色选择页] |   
+----------------+----------------+   
| 学生端 | 教师端 / 家长端 |   
| - 首页 | - 仪表盘 |   
| - 作业提交 | - 教学优化 |   
| - 资源中心 | - 资源贡献 |   
| - 助手与激励 | - 反馈/报告 |   
+----------------+----------------+

* **启动页**: Logo动画，初始化配置。
* **登录页**: 账号密码登录，支持角色标识。
* **角色选择页**: 动态切换学生/教师/家长视图（首次登录后缓存角色）。

### 2.2 功能模块设计

#### 2.2.1 学生端

* **首页**
  + **功能**: 显示学习进度（趋势图）、任务清单。
  + **组件**:
    - **LineChart**: 使用fl\_chart绘制进度趋势。
    - **ListView**: 任务卡片（任务名、截止时间）。
  + **接口**: GET /student/progress 返回 { trend, tasks }。
* **作业提交**
  + **功能**: 拍照上传作业，支持OCR识别。
  + **组件**:
    - **CameraButton**: 调用摄像头拍照。
    - **OCRPreview**: 显示识别结果，允许编辑。
    - **SubmitButton**: 提交作业。
  + **流程**:
    1. 点击拍照 -> 调用OCR -> 显示结果。
    2. 编辑确认 -> 调用API提交。
  + **接口**: POST /homework 请求 { image, text }。
* **资源中心**
  + **功能**: 浏览共享资源，接收推荐。
  + **组件**:
    - **GridView**: 资源卡片（封面、标题、评分）。
    - **VideoPlayer**: 点击播放视频。
  + **接口**: GET /resources/recommended 返回 { id, url, tags }。
* **助手与激励**
  + **功能**: 实时问答、积分兑换、排行榜。
  + **组件**:
    - **ChatBox**: 输入问题，显示助手回复（WebSocket）。
    - **PointsCard**: 显示积分和兑换按钮。
    - **Leaderboard**: 匿名排名列表。
  + **接口**:
    - GET /student/points 返回 { points, rank }。
    - POST /assistant/ask 请求 { question }。

#### 2.2.2 教师端

* **仪表盘**
  + **功能**: 学生表现、建议、反馈概览。
  + **组件**:
    - **Card**: 关键指标（平均成绩、分层人数）。
    - **ListView**: 建议和反馈摘要。
  + **接口**: GET /dashboard 返回 { students, suggestions, feedback }。
* **教学优化**
  + **功能**: 内容分析、资源推荐。
  + **组件**:
    - **PieChart**: 内容覆盖率分布。
    - **ResourceList**: 推荐资源卡片。
  + **接口**: GET /analysis/teacher 返回 { coverage, recommendations }。
* **资源贡献**
  + **功能**: 上传视频/文档。
  + **组件**:
    - **FilePicker**: 选择本地文件。
    - **TagInput**: 添加资源标签。
  + **流程**: 选择文件 -> 输入标签 -> 提交。
  + **接口**: POST /upload/resource 请求 { file, tags }。

#### 2.2.3 家长端

* **反馈提交**
  + **功能**: 输入辅导情况和期望。
  + **组件**:
    - **TextField**: 多行输入框。
    - **RatingBar**: 评分教师期望。
  + **接口**: POST /feedback 请求 { content, rating }。
* **学习报告**
  + **功能**: 查看趋势图和班级对比。
  + **组件**:
    - **LineChart**: 成绩趋势。
    - **BarChart**: 班级对比。
  + **接口**: GET /student/report 返回 { trend, comparison }。
* **指导工具**
  + **功能**: 薄弱点建议+资源。
  + **组件**:
    - **ListView**: 建议卡片（薄弱点+资源链接）。
    - **VideoPlayer**: 播放推荐视频。
  + **接口**: GET /parent/guidance 返回 { weaknesses, resources }。

## 3. 页面设计

### 3.1 学生端页面

**首页**:

[趋势图] [任务清单: 任务1 | 截止日期]

**作业提交**:

[拍照按钮] -> [预览+编辑框] -> [提交]

**资源中心**:

[网格: 资源卡1 | 资源卡2]

**助手与激励**:

[聊天框] | [积分: 100] | [排行榜]

### 3.2 教师端页面

**仪表盘**:

[卡片: 平均分 | 分层] | [建议列表]

**教学优化**:

[饼图: 内容分布] | [推荐资源]

**资源贡献：**

[文件选择] | [标签输入] | [上传]

### 3.3 家长端页面

**反馈提交**:

[文本框] | [评分: ★★★★☆] | [提交]

**学习报告**:

[趋势图] | [对比柱图]

**指导工具**:

[建议: 薄弱点 | 视频链接]

## 4. 技术实现细节

### 4.1 项目结构

text

WrapCopy

lib/ ├── models/ # 数据模型   
│ ├── student.dart   
│ └── resource.dart   
├── pages/ # 页面模块   
│ ├── student/   
│ │ ├── home.dart   
│ │ └── resources.dart   
│ ├── teacher/   
│ └── parent/   
├── widgets/ # 公共组件   
│ ├── chart.dart   
│ └── video\_player.dart   
├── services/ # 服务层   
│ ├── api.dart   
│ └── ocr.dart   
├── main.dart # 主入口

### 4.2 示例代码（学生端 - 作业提交）

dart

WrapCopy

import 'package:flutter/material.dart'; import 'package:flutter\_ocr\_sdk/ocr.dart'; import 'package:dio/dio.dart'; class HomeworkPage extends StatefulWidget { @override \_HomeworkPageState createState() => \_HomeworkPageState();   
}   
class \_HomeworkPageState extends State<HomeworkPage> { String \_ocrResult = ''; File? \_image;   
Future<void> \_pickImage() async { final picked = await ImagePicker().pickImage(source: ImageSource.camera); if (picked != null) { setState(() => \_image = File(picked.path)); \_ocrResult = await OCRService.recognize(picked.path); }   
}   
Future<void> \_submitHomework() async { final formData = FormData.fromMap({ 'image': await MultipartFile.fromFile(\_image!.path), 'text': \_ocrResult, }); await ApiService.post('/homework', data: formData); ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text('提交成功'))); }   
@override Widget build(BuildContext context) { return Scaffold( appBar: AppBar(title: Text('作业提交')), body: Column(   
children: [ ElevatedButton(onPressed: \_pickImage, child: Text('拍照')), if (\_image != null) ...[ Image.file(\_image!),   
TextField(   
controller: TextEditingController(text: \_ocrResult),   
onChanged: (val) => \_ocrResult = val,   
), ElevatedButton(onPressed: \_submitHomework, child: Text('提交')), ],   
],   
),   
);   
}   
}

### 4.3 API封装

dart

WrapCopy

*// lib/services/api.dart* import 'package:dio/dio.dart'; class ApiService { static final Dio \_dio = Dio(BaseOptions(baseUrl: 'https://api.eduinsight.ai')); static Future<Response> get(String path) => \_dio.get(path); static Future<Response> post(String path, {dynamic data}) => \_dio.post(path, data: data); }

## 5. 跨平台适配

### 5.1 iOS

* **样式**: 遵循iOS设计规范（如圆角按钮、下拉刷新）。
* **权限**: Info.plist配置相机、存储权限。
* **推送**: 集成APNs支持。

### 5.2 Android

* **样式**: Material Design（浮动按钮、卡片阴影）。
* **权限**: AndroidManifest.xml配置相机、存储。
* **推送**: 集成FCM（Firebase Cloud Messaging）。

### 5.3 适配策略

* **屏幕适配**: 使用Flutter的MediaQuery动态调整布局。
* **平台检测**: Platform.isIOS切换样式和逻辑。

## 6. 非功能性需求

### 6.1 性能

* **加载优化**: 懒加载资源列表，分页获取数据（每页10条）。
* **缓存**: Hive存储离线任务和资源数据。

### 6.2 稳定性

* **错误处理**: Dio拦截器捕获网络错误，显示友好提示。
* **热更新**: 集成CodePush，支持快速修复。

### 6.3 可维护性

* **组件化**: 抽取Chart、VideoPlayer等复用组件。
* **文档**: 注释关键代码，维护API映射表。

## 7. 风险与应对

* **OCR精度**: 提供手动编辑选项，确保提交准确。
* **推送延迟**: 本地缓存推送内容，网络恢复时同步。
* **兼容性**: 测试多版本iOS/Android（如iOS 14+、Android 10+）。

### 设计亮点

1. **跨平台**: Flutter确保iOS/Android一致体验。
2. **功能整合**: OCR、视频播放、推送无缝集成。
3. **用户友好**: 简洁流程，动态角色切换。