**PROG2002 A3 项目完成步骤指南**

根据评估要求，我将完成项目分为**8个主要步骤**，每个步骤都包含具体的任务和检查点。

**🗓️ 项目完成时间表**

| 阶段 | 时间 | 主要任务 | 交付物 |
| --- | --- | --- | --- |
| 第1周 | Week 5 | 项目规划、数据库设计、环境搭建 | 项目计划、数据库schema |
| 第2周 | Week 6 | API开发、基础功能实现 | 核心API、基础页面 |
| 第3周 | Week 7 | 功能完善、测试、部署准备 | 完整功能、测试报告 |
| 提交周 | Week 7 Monday | 最终测试、提交 | 所有提交材料 |

**📋 详细完成步骤**

**步骤 1: 项目规划与环境搭建 (3-4天)**

**1.1 团队组建与分工**

* 确认团队成员（2人）
* 分配具体任务角色
* 制定沟通计划（会议频率、工具）

**1.2 项目初始化**

* 创建GitHub仓库
* 设置项目目录结构
* 初始化package.json文件
* 安装必要依赖

**1.3 开发环境配置**

* 安装Node.js和npm
* 安装MySQL并创建数据库
* 配置开发服务器
* 设置代码编辑器配置

**GitHub提交**: 初始项目结构提交

**步骤 2: 数据库设计与实现 (2-3天)**

**2.1 数据库设计**

* 分析现有数据库结构（基于A2）
* 设计新的registrations表
* 确定表关系和约束
* 设计索引以提高查询性能

**2.2 数据库实现**

* 创建数据库schema
* 建立表结构和关系
* 插入初始测试数据（至少10个注册记录）
* 创建数据库备份脚本

**2.3 数据库连接配置**

* 配置Node.js与MySQL连接
* 测试数据库连接
* 创建数据库操作工具函数

**交付物**: database/schema.sql, database/initial-data.sql

**步骤 3: RESTful API 开发 (3-4天)**

**3.1 基础API设置**

* 设置Express服务器
* 配置中间件（CORS、body-parser等）
* 创建基础路由结构

**3.2 事件相关API**

* GET /api/events - 获取所有事件
* GET /api/events/:id - 获取单个事件详情（包含注册记录）
* POST /api/events - 创建新事件
* PUT /api/events/:id - 更新事件
* DELETE /api/events/:id - 删除事件（有注册时阻止）

**3.3 注册相关API**

* POST /api/registrations - 创建注册记录
* GET /api/events/:id/registrations - 获取事件注册列表

**3.4 天气API集成**

* GET /api/weather - 获取天气信息
* 实现错误处理和响应格式化

**3.5 API测试**

* 使用Postman测试所有API端点
* 验证错误处理
* 测试数据验证和安全性

**交付物**: 完整的API后端，通过Postman测试

**步骤 4: 客户端网站开发 (4-5天)**

**4.1 AngularJS应用设置**

* 设置AngularJS主应用模块
* 配置路由和导航
* 创建基础控制器和服务

**4.2 页面开发**

* **首页** (/): 展示活动列表和搜索功能
* **活动列表页** (/events): 显示所有活动
* **活动详情页** (/events/:id):
  + 显示活动详细信息
  + 显示注册列表（按时间排序）
  + 集成天气信息显示
* **注册页面** (/register/:eventId):
  + 表单验证
  + 提交注册信息
  + 显示确认信息

**4.3 功能实现**

* 实现事件搜索和筛选
* 实现注册表单验证
* 集成天气API显示
* 实现响应式设计

**4.4 用户体验优化**

* 添加加载状态指示
* 错误处理和用户反馈
* 导航和面包屑

**交付物**: 完整的客户端AngularJS应用

**步骤 5: 管理端网站开发 (3-4天)**

**5.1 管理端AngularJS应用**

* 设置独立的管理端应用
* 配置管理端路由
* 创建管理界面布局

**5.2 CRUD功能实现**

* **事件列表**: 显示所有事件，支持删除操作
* **添加事件**: 表单创建新事件
* **编辑事件**: 表单更新现有事件
* **事件详情**: 显示事件详细信息和注册统计

**5.3 管理功能**

* 实现数据表格和分页
* 添加确认对话框（特别是删除操作）
* 实现表单验证
* 添加管理仪表板

**5.4 管理界面优化**

* 设计专业的管理界面
* 添加数据统计和可视化
* 实现批量操作功能

**交付物**: 完整的管理端AngularJS应用

**步骤 6: 集成与测试 (2-3天)**

**6.1 前后端集成**

* 确保所有API调用正常工作
* 测试数据流和状态管理
* 验证错误处理机制

**6.2 功能测试**

* 测试所有用户流程
* 验证表单提交和数据持久化
* 测试边界情况和错误场景

**6.3 兼容性测试**

* 测试不同浏览器兼容性
* 测试移动设备响应式设计
* 验证API安全性

**6.4 性能测试**

* 测试页面加载速度
* 优化数据库查询
* 压缩和优化静态资源

**交付物**: 测试报告，修复所有关键bug

**步骤 7: 部署到SCU cPanel (1-2天)**

**7.1 部署准备**

* 获取SCU cPanel账户信息
* 准备部署包
* 配置生产环境设置

**7.2 数据库部署**

* 在cPanel中创建MySQL数据库
* 导入数据库schema和数据
* 验证数据库连接

**7.3 文件部署**

* 上传客户端文件到public\_html/client
* 上传管理端文件到public\_html/admin
* 上传API文件到public\_html/api
* 配置.htaccess文件

**7.4 部署验证**

* 测试所有功能在生产环境
* 验证外部API调用
* 检查所有链接和资源

**交付物**: 在线可访问的完整网站

**步骤 8: 文档与提交准备 (2-3天)**

**8.1 项目文档**

* 更新[README.md](https://readme.md/)文件
* 编写API文档
* 创建部署指南
* 编写用户手册

**8.2 演示视频制作**

* 规划视频内容（15分钟以内）
* 录制代码讲解和功能演示
* 编辑和优化视频
* 上传到SCU OneDrive并生成分享链接

**8.3 最终提交准备**

* 打包源代码文件
  + usernameA3-clientside.zip
  + usernameA3-adminside.zip
  + usernameA3-api.zip
* 准备GitHub仓库链接
* 填写同伴评估表
* 编写GenAI使用声明

**8.4 最终检查**

* 验证所有要求是否满足
* 检查代码质量和注释
* 确认GitHub提交记录展示工作进度
* 测试最终部署版本

**交付物**: 所有要求的提交材料

**🔍 关键检查点**

**技术检查点**

* 数据库设计符合要求，包含注册表
* 所有API端点正常工作
* 客户端网站包含注册功能和改进的事件页面
* 管理端网站实现完整的CRUD功能
* 网站成功部署到SCU cPanel
* AngularJS正确集成和使用

**评估要求检查点**

* 团队协作通过GitHub提交记录展示
* 演示视频覆盖所有要求的讲解点
* 同伴评估表按时提交
* GenAI使用声明包含在提交中
* 所有源代码按要求打包

**🛠️ 推荐的工作流程**

**每日工作流程**

1. **早上同步** (15分钟): 讨论前一天进展和当天计划
2. **专注开发** (4-6小时): 各自完成分配的任务
3. **代码审查** (30分钟): 互相审查代码更改
4. **提交和同步**:
   * 频繁提交到GitHub（至少每天1-2次）
   * 使用有意义的提交信息
   * 及时解决合并冲突

**每周检查点**

* **周一**: 制定本周目标，分配任务
* **周三**: 中期检查，调整计划
* **周五**: 总结本周进展，规划下周工作

**⚠️ 常见问题与解决方案**

**技术问题**

1. **数据库连接失败**: 检查cPanel数据库配置
2. **API跨域问题**: 确保CORS正确配置
3. **AngularJS路由问题**: 检查base href和服务器配置

**团队协作问题**

1. **代码冲突**: 建立明确的分工，频繁同步
2. **进度不一致**: 定期沟通，及时调整任务分配
3. **质量不一致**: 建立代码标准和审查流程

通过按照这8个步骤系统地进行，您的团队可以高效地完成PROG2002 A3评估，确保满足所有要求并获得最佳成绩。记得在整个过程中保持定期的GitHub提交，这不仅是评估要求，也是展示团队协作的重要证据。