软件开发综合实训

项目计划书

|  |  |
| --- | --- |
| **学院** | **软件学院** |
| **专业** | **软件工程** |
| **小组成员** | **欧若琪 宁炤宇 卓深伟朱怡超** |
| **提交日期** | **2025年 3月 30日** |

**变更记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **变更说明** | **作者** |
| 2025/3/28 | 1.0 | 创建 |  |
| 2025/3/30 | 1.1 | 修改了部分内容和分工 |  |
|  |  |  |  |

目录

[1.基本信息 5](#_Toc165643242)

[1.1 需求分析 5](#_Toc165643243)

[**1.2 项目基本信息** 5](#_Toc165643244)

[**1.3 项目人员分工** 6](#_Toc165643245)

[**1.4 里程碑和项目阶段性成果提交** 6](#_Toc165643246)

[2.基本任务 7](#_Toc165643247)

[**2.1 项目内容** 7](#_Toc165643248)

[**2.2 项目计划** 8](#_Toc165643249)

[**2.2.1 项目生命周期** 8](#_Toc165643250)

[**2.2.2 项目框架的搭建** 8](#_Toc165643251)

[**2.2.3 项目有关代码的实现** 8](#_Toc165643252)

[**2.2.4 项目的测试与维护** 9](#_Toc165643253)

[**2.2.5 项目管理** 10](#_Toc165643254)

[**2.2.6 部署上线** 11](#_Toc165643255)

[**2.2.7 项目文档的攥写** 11](#_Toc165643256)

[**2.3 预期成果** 12](#_Toc165643257)

[**2.4 性能要求** 13](#_Toc165643258)

[3.项目访问权限及其存档目录 14](#_Toc165643259)

[**3.1 开发库结构** 14](#_Toc165643260)

[**3.1.1 springboot结构** 14](#_Toc165643261)

[**3.1.2 Vue结构** 15](#_Toc165643262)

[**3.2 开发库访问权限** 16](#_Toc165643263)

[**3.2.1 角色和权限** 16](#_Toc165643264)

[**3.2.2 访问控制策略** 16](#_Toc165643265)

[**3.2.3 权限控制实现** 17](#_Toc165643266)

[**3.3 开发项目阶段及其对应产品的存放地点** 17](#_Toc165643267)

[**3.3.1计划阶段** 17](#_Toc165643268)

[**3.3.2设计阶段** 17](#_Toc165643269)

[**3.3.3开发阶段** 17](#_Toc165643270)

[**3.3.4测试阶段** 17](#_Toc165643271)

[**3.3.5部署阶段** 18](#_Toc165643272)

[**3.3.6维护阶段** 18](#_Toc165643273)

[4.项目协调与跟踪 18](#_Toc165643274)

[**4.1 协调方式与频率** 18](#_Toc165643275)

[**4.1.1团队会议** 18](#_Toc165643276)

[**4.1.2沟通工具** 18](#_Toc165643277)

[**4.1.3定期报告** 18](#_Toc165643278)

[**4.2 跟踪方法与频率** 19](#_Toc165643279)

[**4.2.1项目进度跟踪** 19](#_Toc165643280)

[**4.2.2变更管理** 19](#_Toc165643281)

1.基本信息

# 1.1 需求分析

打卡功能在多种应用场景中具有重要价值，如企业考勤、校园签到、活动签到等。本实训项目旨在让学生结合业务背景需求，在以产品级应用为目标的实践开发中，掌握软件工程的完整流程，同时锻炼解决实际问题的能力。

## **1.2 项目基本信息**

|  |  |
| --- | --- |
| **项目名称** | Sisyphus Tempus打卡系统 |
| **项目组长** | 欧若琪 |
| **项目组员** | 宁炤宇 卓深伟 朱怡超 |
| **项目开始时间** | 2025/3/30 |
| **项目结束时间** | 2025/6/1 |
| **项目目标** | 本项目的目标是开发一款兼容多平台（Windows、macOS、Android等系统）的打卡软件，满足用户在不同设备上便捷使用的需求。软件将具备跨平台同步功能，确保用户的打卡记录能够无缝衔接，同时提供友好的用户界面和高效的操作体验，以适应不同操作系统的特点。此外，项目将注重数据安全性和隐私保护，确保用户的个人信息得到妥善管理。 |
| **成功标准** | 1.制作使用体验良好、操作流畅简洁、安全性高的打卡应用  2.按照项目管理流程按时完成项目管理文档。 |
| **方式** | 通过瀑布式开发的方式制作Web应用 |

## **1.3 项目人员分工**

|  |  |
| --- | --- |
| 姓名 | 职责 |
| 欧若琪 | 组织项目分工、设计前端UI并编写相关代码、撰写开发文档 |
| 宁炤宇 | 项目的前端开发、原型设计和系统设计 |
| 卓深伟 | 项目的后端开发与维护 |
| 朱怡超 | 项目的后端开发与维护 |

## **1.4 里程碑和项目阶段性成果提交**

| **阶段** | **里程碑** | **提交产品** | **时间** |
| --- | --- | --- | --- |
| 第一阶段：实训概述，计划 | 计划 | 1. 项目开发计划书 | 2025/3/30 |
| 第二阶段：需求分析，原型设计 | 需求 | 1. 需求分析报告   ②原型系统 | 2025/4/13 |
| 第三阶段：技术方案 | 设计 | 1. 技术方案文档 | 2025/4/20 |
| 第四阶段：软件编程，集成和测试 | 实现 | ①demo功能演示视频   1. 个人阶段性工作汇报和周实训报 | 2025/5/11 |
| 第五阶段：系统迭代与功能完善，缺陷修复，答辩准备 | 测试 | ①系统功能演示视频   1. 个人阶段性工作汇报和周实训报 | 2025/5/25 |
| 第六阶段：项目组汇报总结 | 项目交付 | ①系统源代码  ②最终产品演示视频  ③实训综合报告 | 2025/6/1 |

2.基本任务

## **2.1 项目内容**

本项目的目标是开发一款兼容多平台（如 Windows、macOS、Android 等系统）的打卡软件，满足用户在不同设备上便捷使用的需求。软件将具备跨平台同步功能，确保用户的打卡记录能够无缝衔接。同时，应用将提供友好的用户界面和高效的操作体验，以适应不同操作系统的特点。此外，项目将重点关注数据安全性和隐私保护，确保用户的个人信息得到妥善管理。

本应用的前端采用 Vue，后端采用 Java 和 Python，数据存储和管理使用 MySQL。

## **2.2 项目计划**

### **2.2.1 项目生命周期**

本项目的开发由六个阶段构成：项目计划，需求分析与系统设计，前后端技术方案设计，前后端编程、集成与测试，系统迭代与功能完善，项目部署与总结，每个阶段一周到两周时长，共六周完成。

### **2.2.2 项目框架的搭建**

（1）前端框架搭建：使用Vue完成前端的搭建。

实现图形化用户交互界面：

普通用户：按照管理员用户发布的打卡链接完成打卡

管理员用户：创建、设置和发布打卡

遵循UI/UX设计原则，确保界面用户友好、功能实用、分区明确、操作简便、反馈积极。

与后端对接：用规范的前后端接口，确保前后端数据交互的顺畅。

（2）后端框架搭建：使用Java、Spring Boot、Python搭建后端，连接MySQL数据库。

实现对打卡数据进行管理，人脸识别打卡等功能

设计高内聚、低耦合的系统，做到功能模块化、架构合理、可读性高。

实现与数据库的交互：让用户的注册和权限信息以及打卡的创建、编辑、发布、修改、删除等信息能够实时存储，并且能够通过Excel表格的方式导出，便于管理和统计

与前端对接：设计接口，与前端进行数据交互；将数据格式化进行传输，确保数据的完整性和可读性。

### **2.2.3 项目有关代码的实现**

（1）前端代码实现：

设计和开发普通用户：用户登录与注册、打卡记录查询、打卡提醒设置等页面的 UI；管理员用户：用户管理、打卡记录分析、数据统计与导出等页面的 UI。

利用 Vue 组件化开发，将界面拆分为可复用的组件，提高开发效率。在 Vue 项目中，components 文件夹里存放可复用的组件。

配置路由：在 Vue 项目中，router 文件夹里创建 index.js 文件，用于路由管理，实现页面跳转与权限控制。在 main.js 中引入并使用路由。

实现与后端的数据交互：将用户信息的认证与修改、打卡记录的获取与提交等操作与后端进行通信，使用工具发送 HTTP 请求，实现数据交互与后端的实时同步。

（2）后端代码实现：

使用稳定性高、应用广泛的Java作为后端开发的主要语言；使用Spring Boot作为后端框架，简化项目搭建和开发；连接MySQL数据库，实现用户信息、打卡信息的存储与修改。使用Python中的OpenCV库进行人脸识别，并通过 REST API 实现 Python 和 Java 交互

实现与前端的数据交互：在后端定义规范的相应接口和类，接收和处理前端的HTTP请求。使用JSON等格式进行数据传输，确保数据的完整性和可读性。

### **2.2.4 项目的测试与维护**

（1）测试

单元测试：

对后端Java代码进行单元测试，确保各个模块的功能正常。

集成测试：

将系统的不同模块集成后进行测试，确保模块之间的能够正常协同工作。

端到端测试：

对整个系统进行端到端测试，模拟用户行为使用系统，确保系统各功能流程正常。

（2）维护

Bug修复：

在开发过程中及时修复发现的bug，在单元测试、集成测试、端对端测试后及时修复发现的bug。部署发布后，及时收集用户反馈，修复用户反馈的Bug。

功能优化：

根据每周的讨论，及时根据需求分析的变化，更新功能设计，优化系统设计的不足和缺陷。每轮测试后根据测试人员反馈优化系统功能。部署发布后，根据用户反馈和需求变化，持续对系统功能进行优化。

性能优化：

定期进行性能优化，提高系统的响应速度和稳定性。

用户支持：

提供用户支持，解决用户使用过程中遇到的问题，并发布和定期更新系统使用说明，帮助用户更高效地使用系统。

### **2.2.5 项目管理**

（1）项目范围：

明确项目的范围和目标，包括功能需求、技术需求、非功能需求等，确保团队对项目目标有清晰的认识。

（2）项目计划：

制定清晰的项目计划，包括项目进度计划、任务分配等，确保项目按时交付。

（3）项目团队：

明确团队分工，明确每个团队成员任务的分配，明确每个任务的交付时间。自组织团队，不以职位分工，而以能力分工，实现敏捷开发。

（4）项目沟通：

制定良好的项目沟通机制，团队成员通过每周的线下会议和每天的线上交流进行沟通。

（5）项目质量：

设立质量标准，进行质量控制，确保项目交付的产品符合质量要求。

（6）项目变更：

管理项目变更，包括需求变更、技术变更等，以便控制变更、及时应对变更。

（7）项目评估：

每一到两周进行项目评估，分析项目进展和问题，并及时采取改进措施。

（8）项目总结

发现和记录系统开发过程中遇到的问题，项目完成后总结，为下一次项目实施提供经验和教训。

### **2.2.6 部署上线**

（1）环境安装：

确保部署环境的安装已经完成，包括前后端软件配置、系统环境变量配置、数据库配置、网络设置等。

（2）部署流程：

制定部署计划，包括代码部署、数据库迁移、配置文件更新等，确保部署流程清晰明确。

（3）测试环境部署验证：

在测试环境中进行部署验证，确保部署的系统稳定可靠。

（4）监控与报警：

设计系统监控和报警机制，确保在部署过程中能够及时发现和处理问题。

（5）上线验证：

在部署完成后，进行上线验证，包括功能验证、性能验证、安全性验证等，确保部署的系统符合预期。

（6）用户支持：

在系统部署上线后，提供用户支持，确保用户能够顺利使用新系统。

（7）持续监控与优化：

在系统上线后，持续进行系统监控和优化，及时发现和解决问题，提高系统的稳定性和性能。

### **2.2.7 项目文档的攥写**

（1）项目计划文档：

明确项目计划文档的内容和格式，包括项目基本信息、基本任务项目访问权限、项目协调与跟踪等。制定设计文档的编写计划，明确文档编写的时间和责任人。

（2）需求分析文档：

明确需求文档的内容和格式，包括功能需求、非功能需求、用例分析等。制定需求文档的编写计划，明确文档编写的时间和责任人。

（3）系统设计文档：

明确系统设计文档的内容和格式，包括系统架构、数据库设计、接口设计等。制定设计文档的编写计划，明确文档编写的时间和责任人。

（4）技术方案文档

明确技术方案文档的内容和格式，包括系统适用对象、系统整体架构、功能模块设计、用例时序图、解决方案设计等。制定设计文档的编写计划，明确文档编写的时间和责任人。

（5）开发编程文档（可选）：

明确系统开发、编程实现文档的内容和格式，包括代码注释、接口文档、模块说明等。制定开发文档的编写计划，明确文档编写的时间和责任人。

（6）系统测试文档（可选）：

明确测试文档的内容和格式，包括测试计划、测试用例、测试报告等。制定测试文档的编写计划，明确文档编写的时间和责任人。

（7）系统迭代文档（可选）：

明确系统迭代文档的内容和格式，包括系统更新日志、问题解决记录、版本变更说明等。

（8）项目总结文档：

明确项目总结文档的内容和格式，包括项目信息、项目说明、项目周期、开发工具和环境、开发技术、任务分配、项目问题及解决、组间协调跟踪、提交产品清单、项目经验总结等。制定系统使用说明的编写计划，明确文档编写的时间和责任人。

## **2.3 预期成果**

打卡系统实现的主要功能：

**普通用户：**

1. **用户注册与登录**  
   用户可以注册、登录账号，进行身份验证。
2. **打卡任务选择与提交**  
   用户可以选择打卡任务，进行打卡并提交记录。
3. **查看历史记录与统计**  
   用户可以查看自己的打卡历史和统计数据。
4. **个人信息管理**  
   用户可以修改个人资料和设置。

**管理员用户：**

1. **用户管理**  
   管理员可以审核用户注册信息、管理用户账户。
2. **打卡任务管理**  
   管理员可以创建、编辑、发布、删除打卡任务。
3. **数据统计与分析**  
   管理员可以查看打卡任务的统计数据和用户活跃度。
4. **通知与公告发布**  
   管理员可以发布通知和活动信息。

## **2.4 性能要求**

|  |  |
| --- | --- |
| **性能要求** | |
| **精度** | 打卡系统应对用户的打卡操作进行精准响应，特别是人脸识别部分，系统返回的识别结果错误率应低于10%。 |
| **稳定性和可靠性** | 本系统应稳定运行，能应对处理大量用户同时访问的情况，降低网页崩溃或数据丢失等问题发生的可能。 |
| **响应时间** | 1. **查询服务部分**：用户提交打卡请求到返回结果之间的响应时间不超过3秒钟。 2. **数据管理部分**：用户提交打卡记录到结果返回之间的响应时间不超过3秒钟。 |
| **用户友好性** | 本系统的网页界面设计应简洁清晰，操作方便，使用户能够轻松进行打卡操作、查看打卡记录、修改个人信息等。特别是在进行人脸识别时，界面应提供清晰的操作指导，确保用户顺利完成打卡。 |
| **兼容性** | 支持在多种操作系统环境下使用（如Windows系统，macOS系统、Android系统、鸿蒙系统等）。 |

3.项目访问权限及其存档目录

## **3.1 开发库结构**

### **3.1.1 springboot结构**

|  |
| --- |
| survey-system/  ├── src/  │ ├── main/  │ │ ├── java/  │ │ │ ├── com/  │ │ │ │ ├── example/  │ │ │ │ │ ├── SurveySystem/  │ │ │ │ │ │ ├── controller/  │ │ │ │ │ │ ├── service/  │ │ │ │ │ │ ├── repository/  │ │ │ │ │ │ └── SurveyApplication.java  │ │ │ │ │ └── ...  │ │ ├── resources/  │ │ │ ├── mapper  │ │ │ ├── Survey.xml  │ │ │ └── static/  │ │ │ ├── static/  │ │ │ ├── css/  │ │ │ ├── fonts/  │ │ │ ├── img/  │ │ │ ├── js/  │ │ │ ├── favicon.ico  │ │ │ ├── index.html  │ │ │ ├── application.yml  │ │ │ ├── application.properties  │ ├── test/  │ │ ├── java/  │ │ │ ├── com/  │ │ │ ├── …//与src部分类似  └── pom.xml |

### **3.1.2 Vue结构**

|  |
| --- |
| Survey-system-frontend/  ├── build/  ├── config/  ├── node\_modules/  ├── static/  │ ├── favicon.ico  ├── src/  │ ├── assets/  │ │ ├── css/  │ │ │ ├── app.css  │ │ │ └── ...  │ │ ├── images/  │ │ └── js/  │ │ ├── app.js  │ ├── components/  │ │ └── HelloWorld.vue  │ ├── pages/  │ │ └── page1.vue  │ │ └── page2.vue  │ │ └── ….vue  │ │ └── page\_n.vue  │ ├── views/  │ │ ├── Home.vue  │ │ └── About.vue  │ ├── router/  │ │ └── index.js  │ ├── store/  │ │ └── index.js  │ ├── App.vue  │ └── main.js  ├── package.json  ├── index.html |

## **3.2 开发库访问权限**

### **3.2.1 角色和权限**

|  |  |
| --- | --- |
| 角色 | 权限 |
| 项目管理员 | 完全访问所有代码库、数据库和文档 创建、删除、编辑用户角色和权限 管理项目设置和配置 |
| 开发人员 | 访问代码库以编写、测试和部署代码 查看数据库和文档 不能编辑用户角色和权限 |
| 测试人员 | 访问代码库以进行测试 查看数据库和文档 不能编辑用户角色和权限 |

### **3.2.2 访问控制策略**

项目管理员：拥有最高权限，可以管理项目的各个方面，包括团队成员的权限。

开发人员：可以访问和修改代码库，参与项目的开发工作。

测试人员：可以访问代码库以执行测试，确保软件质量。

### **3.2.3 权限控制实现**

1.通过用户名和密码或更安全的认证方式（如OAuth、双因素认证等）登录系统。

2.角色基础权限控制，系统根据用户的角色分配不同的权限。

3.对特定资源（如文件、目录、数据库记录等）进行权限设置，以限制对敏感信息的访问。

4.记录用户的操作历史，以便于审计和追踪。

## **3.3 开发项目阶段及其对应产品的存放地点**

### **3.3.1计划阶段**

* 产品需求文档
* 项目计划和时间表
* 项目预算和资源分配

存放地点：企业的文档管理系统或共享文件夹

### **3.3.2设计阶段**

* 系统架构设计文档
* UI/UX设计原型和图表
* 数据库设计文档

存放地点：设计团队的如Git仓库

### **3.3.3开发阶段**

* 代码文件
* 开发过程中生成的测试用例和报告
* 开发日志和任务跟踪

存放地点：Git仓库、源代码管理平台

### **3.3.4测试阶段**

* 测试计划和测试用例
* 缺陷报告和修正记录
* 测试结果和性能分析报告

存放地点：测试团队的共享工作区、缺陷跟踪系统

### **3.3.5部署阶段**

* 部署计划和步骤文档
* 生产环境配置和设置记录
* 用户培训材料和操作手册

存放地点：运维团队的知识库或文档管理系统

### **3.3.6维护阶段**

* 用户反馈和投诉记录
* 维护和更新日志
* 安全补丁和升级记录

存放地点：客户关系管理系统、IT服务管理系统

4.项目协调与跟踪

在整个项目周期中，项目协调与跟踪是确保项目按计划进行并达到预期目标的关键活动。

## **4.1 协调方式与频率**

在项目进行过程中，协调团队成员之间的合作沟通至关重要。以下是我们将采用的协调方式和频率：

### **4.1.1团队会议**

按照项目计划书推进，每周举行一次团队会议，用于讨论项目进展、问题和解决方案。在软件编程阶段可能需要成员根据分工分别讨论，然后再进行集体会议

### **4.1.2沟通工具**

使用微信，QQ等即时通讯工具进行日常沟通。

使用Git进行代码管理

使用腾讯会议进行线上会议。

### **4.1.3定期报告**

每个成员每阶段提交周报总结自己的工作和下一阶段的计划，整个团队提交小组报告。

## **4.2 跟踪方法与频率**

项目跟踪是确保项目按照计划进行并及时采取必要的措施来解决问题的关键方案。

### **4.2.1项目进度跟踪**

每周更新项目进度，跟踪任务的完成情况。

及时发现问题并采取措施解决问题，记录解决方案和经验。

### **4.2.2变更管理**

定期审查和更新项目计划，确保变更得到适当的管理和控制