# 线上商城项目计划文档

## 1. 简介

### 1.1 项目概述

我们设计的这个软件项目是一个线上商城软件，能够为顾客和商家提供一个交易的平台，并且需要客服从中进行管理以使软件功能正常运行。

#### 1.1.1 意图、范围和目标

本项目旨在设计并开发一个全新的线上商城系统，以满足现代消费者对于便捷、安全和多样化购物体验的需求。考虑到实现难度等问题，使用范围不宜过大，至少应该能够使学校的学生进行使用，我们的目标是提供一个直观、高效的平台，使用户能够轻松浏览和购买各种商品，并为商家提供一个稳定可靠的销售渠道。

#### 1.1.2 设想和限制

在设计这个系统时，我们将注重用户体验、安全性和性能。同时，我们也要考虑到技术可行性和预算限制、设计时间等因素。本项目将遵循现代软件开发实践，并尽可能采用开源技术和可扩展的架构。

#### 1.1.3 可交付项目

可交付项目将包括但不限于以下主要组成部分：

用户界面：提供直观友好的用户界面。

商品管理：包括商品搜索、详情展示等功能。

订单处理：实现订单下单、支付、物流跟踪等流程。

用户管理：支持用户注册、登录、个人信息管理等功能。

后台管理：提供管理界面，方便商家管理商品、订单等信息。

#### 1.1.4 时间表和预算概述

预计本项目将在六月初完成，预算约为200元。具体的时间表和预算分配将在后续详细规划阶段确定

### 1.2 项目管理计划的演化

我们将使用增量模型进行开发，并行开发提高效率，及时对项目进行分析、估算、计划并合理控制项目的进度、调度等等，并在后续开发中对项目管理计划不断优化改进。

## 2. 参考材料

我们将参考github上面有关线上商城的开源软件，如https://github.com/linlin java/litemall此类。

## 3. 定义和缩略语

在线商城系统：一个通过互联网提供商品或服务购买的平台，通常包括商品展示、购物车、结算等功能。

用户界面 (UI)：用户与系统进行交互的界面，包括网页、移动应用等。

后端：系统的服务器端部分，处理业务逻辑和数据存储。

前端：系统的客户端部分，负责与用户交互和展示内容。

数据库：用于存储和管理数据的系统。

API：应用程序接口，用于不同系统之间的通信和数据传输。

服务器：提供计算服务的硬件设备，通常运行后端程序。

客户端：使用系统的终端用户，可能是网页浏览器或移动设备上的应用程序。

## 4. 项目组织

### 4.1 外部接口

支付网关：

集成第三方支付服务，如PayPal、支付宝、微信支付等。

实现安全的支付流程，保护用户的支付信息。

供应商接口：

与供应商合作，确保商品库存充足和及时更新。

实现自动化的库存管理和订单处理，减少人工干预。

### 4.2 内部结构

前端开发团队：负责用户界面设计和开发，包括网站和移动应用。能够使用现代前端技术，如React、Vue.js等，确保良好的用户体验。

后端开发团队：负责后端系统设计和开发，包括业务逻辑和数据库管理。使用适当的后端技术栈，如Node.js、Django、Spring等。

测试团队：负责系统的功能测试、性能测试和安全测试。使用自动化测试工具，确保系统稳定性和安全性。

运维团队：负责系统的部署、监控和维护。

### 4.3 角色和责任

后端开发人员（2人）：负责后端系统设计和开发，包括业务逻辑和数据库管理。使用适当的后端技术栈，如Node.js、Django、Spring等。

前端开发人员（3人）：负责用户界面设计和开发，包括网站和移动应用。能够使用现代前端技术，如React、Vue.js等，确保良好的用户体验。

测试人员：根据测试计划，执行系统的功能测试和性能测试。报告和跟踪发现的问题，与开发团队合作解决缺陷。

**人员责任矩阵表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **活动\角色** | **顾田** | **孙龙灵** | **戴威扬** | **邢益鸣** | **赵晚甜** |
| **需求分析和规划** | R P | P | P | P | P |
| **设计和界面开发** | - | - | R P | P | P |
| **后端开发** | R P | P | - | - | - |
| **测试和调试** | P | R P | P | P | P |
| **部署和上线** | R P | P | P | P | P |

R-负责 P-执行

## 5. 管理过程计划

### 5.1 启动计划

#### 5.1.1 估算计划

整个项目开发时间周期为一个多月，成本估算在200块内，代码规模不大，实现难度适中。

#### 5.1.2 人员安排计划

项目负责人即项目经理为组长，五人共同组成开发小组和测试小组，一同完成项目的开发和测试，具体开发中两人负责后端三人负责前端。

#### 5.1.3 资源获取计划

在github等开源网站上寻找相关的开源项目框架等。

#### 5.1.4 项目人员培训计划

由开发人员根据具体开发需要学习的各项知识利用网络自行寻找相应学习内容，并互相分享帮助其他成员更快掌握相应知识基础。

### 5.2 工作计划

#### 5.2.1 工作活动

需求分析和规划：收集用户需求和功能要求；制定项目计划和开发路线图。

设计和界面开发：设计网站/应用的用户界面和交互流程。开发网站/应用的前端界面。

后端开发：设计数据库结构。开发后端业务逻辑和接口。

测试和调试：执行功能测试和用户验收测试。修复和调试系统中的问题。

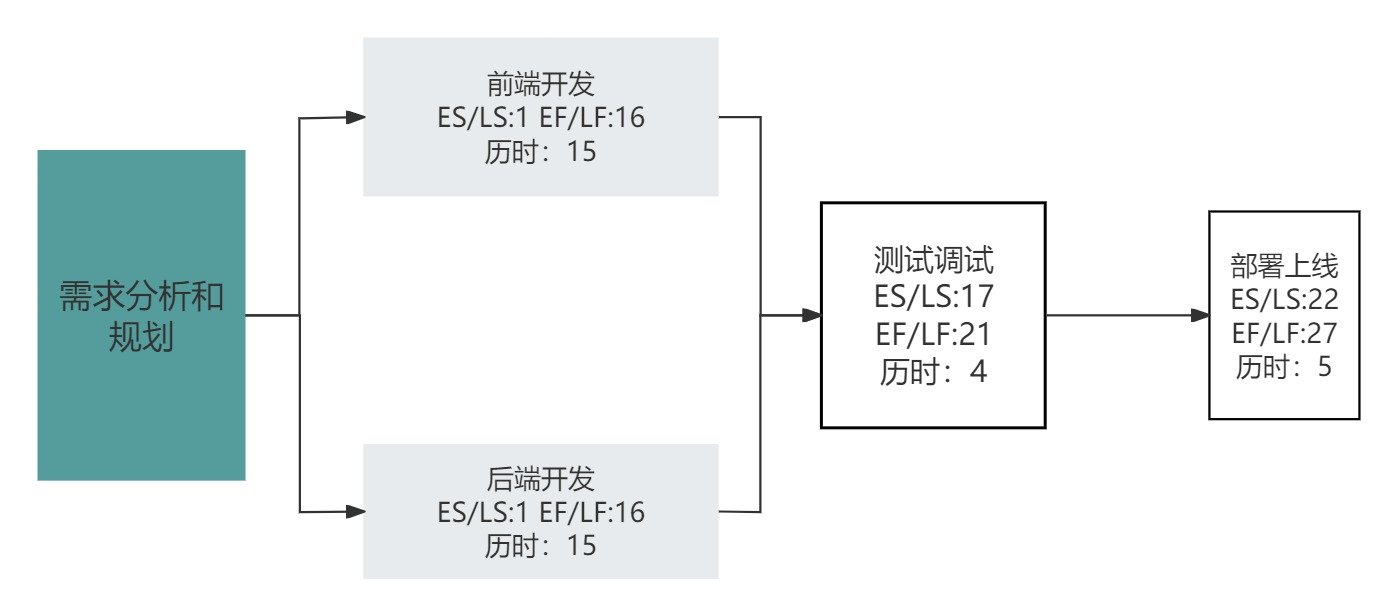
部署和上线：配置服务器环境。部署系统到线上环境。

#### 5.2.2 时间安排

甘特图

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 任务 | 开始时间 | 完成时间 | 持续时间 | 5月第二周 | 5月第三周 | 5月第四周 | 6月第一周 |
| 需求分析和规划 | 2024.4 | 2024.5.7 | 一个月 |  |  |  |  |
| 设计和界面开发 | 2024.5.7 | 2024.5.26 | 两周 | √ | √ |  |  |
| 后端开发 | 2024.5.7 | 2024.5.26 | 两周 | √ | √ |  |  |
| 测试和调试 | 2024.5.27 | 2024.5.31 | 一周 |  |  | √ |  |
| 部署和上线 | 2024.6.1 | 2024.6.6 | 一周 |  |  |  | √ |

任务网络图



历时 28

#### 5.2.3 资源分配

需求分析规划：五人共同讨论

设计和界面开发：三人

后端开发：两人

测试和调试：五人

部署和上线：五人

#### 5.2.4 预算分配

主要开支基本集中在服务器等等必须借助的外部服务收费。

### 5.3 控制计划

在软件开发过程中变更是不可避免的，变更可能来自于需求调整、技术限制、用户反馈等因素，因此需要考虑控制方法来确保变更的合理性和对软件的影响的可控性。

#### 5.3.1 需求控制计划

收到一个新的需求变更请求时，根据需求控制计划，会进行以下行动：

- 评估，首先会进入评估环节，充分考虑该需求是否与需求计划中的有所冲突或重复、该变更对现有项目的影响范围、对成本预算和质量进度的影响，并进行多方的沟通和协商。

- 更新，若需求通过评估，将会接受并记录到变更日志中，团队会相应地更新项目文档和计划。同时更新配置管理和版本控制等，以确保能够按照计划进行。必要时需要通知利益相关者，如用户。

#### 5.3.2 时间控制计划

项目进度若有了延误或其他异常，首先分析问题原因，重新分配资源和安排任务，以最大程度减少影响。制定明确的目标和优先级并按照计划执行任务。

#### 5.3.3 预算控制计划

团队发现某项任务的成本超出了预算，根据预算控制计划，首先分析成本偏差的原因，并确定是否需要调整预算或重新分配资源。然后与负责预算的利益相关者进行沟通，共同制定解决方案，并更新项目预算和成本管理计划。

#### 5.3.4 质量控制计划

若在产品开发阶段发现了功能缺陷，首先停止当前的开发工作，然后对缺陷进行详细的分析，并制定纠正措施。一旦纠正措施实施完毕，团队会进行再次测试，确保缺陷已被修复。最后更新测试报告和质量记录。

#### 5.3.5 报表计划

团队每周需向项目主管发送项目进展报告。报告包括项目进度、预算状况、风险和问题的摘要，以及下周的工作计划。报告采用简洁清晰的格式，以便干系人快速了解项目的最新状态。

#### 5.3.6 度量收集计划

定期收集项目数据，并使用成本绩效指数（CPI）和进度绩效指数（SPI）等指标进行分析。如果发现了任何异常或潜在的问题，团队将立即采取行动，并在下一次度量收集时进行跟踪和监控。

### 5.4 风险管理计划

- 风险识别：定期召开风险识别会议，邀请项目团队成员提供潜在的风险，并记录在风险登记册中。

- 风险分析：评估每个风险的可能性、影响程度和优先级，并使用概率-影响矩阵进行可视化分析。

- 风险响应规划：针对每个高优先级风险，制定相应的应对策略，包括避免、减轻、转移或接受风险。

- 风险监控与控制：定期审查风险登记册，更新风险状态和应对计划，并根据需要调整风险管理策略。

负责人领导风险管理活动，并确保项目团队按照风险管理计划的要求执行。开发组负责识别与软件开发相关的技术风险，并提供技术方面的意见和建议。测试组负责评估软件质量风险，并提供测试方面的意见和建议。所有成员都负有责任在日常工作中识别和报告潜在的风险。

风险信息表

|  |  |
| --- | --- |
| **编号** | **风险名称** |
| **1** | 软件项目规模的估算结果过于乐观 |
| **2** | 软件开发过程中技术困难或挑战超出预期 |
| **3** | 项目成员中途退出 |

|  |  |
| --- | --- |
| **软件风险管理计划** | |
| **风险编号** | 1 |
| **风险名称** | 软件项目规模的估算结果过于乐观 |
| **风险影响** | 可能导致项目进度延迟、项目质量下降 |
| **监控方法** | 对进度和质量进行实时跟踪，定期审查项目规模估算，并与实际进展进行比较 |
| **应急计划** | 提前规划额外的时间来应对可能的延迟。立即采取纠正措施，并重新评估项目规模和进度，以确保最终交付物的质量 |

|  |  |
| --- | --- |
| **软件风险管理计划** | |
| **风险编号** | 2 |
| **风险名称** | 软件开发过程中技术困难或挑战超出预期 |
| **风险影响** | 可能导致团队士气下降、项目进度延迟 |
| **监控方法** | 定期进行技术评估和代码审查，以识别可能的技术挑战或困难。团队内成员应及时反映技术问题 |
| **应急计划** | 立即召开紧急会议，团队共同讨论解决方案 |

|  |  |
| --- | --- |
| **软件风险管理计划** | |
| **风险编号** | 3 |
| **风险名称** | 项目成员中途退出 |
| **风险影响** | 项目进度延迟 |
| **监控方法** | 团队内定期沟通以了解成员情况 |
| **应急计划** | 立即重新分配任务，并调整项目计划 |

### 5.5项目收尾计划

- 项目交付：完成最终成果的交付，包括所有文档、软件产品和其他交付物。

- 汇报和总结：汇报总结项目成果、成本和时间情况。

- 文件整理和存档：对项目文档进行整理和归档，确保归档的文件易于查阅和访问。创建项目存档，包括项目计划、会议记录、报告和其他相关文档。

- 风险管理和问题解决：对项目期间未解决的风险和问题进行跟踪和解决，确保不留后患。归档风险管理文档和问题解决记录，以备将来参考。

## 6. 技术过程计划

### 6.1 过程模型

极限编程模型（Extreme Programming），也称XP，是一个轻量级的、灵巧的软件开发方法，同时它也是一个非常严谨和周密的方法。它的基础和价值观是交流、朴素、反馈和勇气（任何一个软件项目都可以从四个方面入手进行改善：加强交流、从简单做起、寻求反馈、勇于实事求是。）

XP是一种近螺旋式的开发方法，它将复杂的开发过程分解为一个个相对比较简单的小周期，通过积极的交流、反馈以及其它一系列的方法，开发人员和客户可以非常清楚开发进度、变化、待解决的问题和潜在的困难等，并根据实际情况及时地调整开发过程。

### 6.2 方法，工具和技术

- 方法：结对编程（Pair Programming），是极端编程的组成部分。两个程序员在一个计算机上共同工作。一个人输入代码，而另一个人审查他输入的每一行代码，两人会经常互换角色。两名开发人员共同编写代码，提高代码质量和产品稳定性。

- 工具：使用Git进行代码版本控制，GitHub作为代码仓库。

- 技术

前端：HTML5、CSS3、JavaScript、Vue框架

后端：SpringBoot

### 6.3 基础结构计划

操作系统：开发团队使用各自的操作系统，包括Windows、macOS和Linux。

开发工具：使用IntelliJ IDEA作为主要的集成开发环境（IDE）。

### 6.4 产品验收计划

功能验收：根据需求文档中的功能清单，逐项验证系统功能是否符合预期。

性能验收：使用压力测试工具对系统进行性能测试，确保系统在高负载情况下能够稳定运行。

安全验收：进行安全审计和漏洞扫描，确保系统的安全性和稳定性。

内部验收：项目团队内部进行初步验收，确保系统符合设计和功能要求。

客户验收：将系统交付给客户进行最终验收，根据客户的反馈进行调整和修复。

## 7. 支撑过程计划

### 7.1 配置管理计划

- 配置项标识：使用唯一标识符对项目中的所有配置项进行标识，包括代码、文档、库存和硬件设备等。

- 配置控制：确立配置变更控制流程，包括变更请求的提交、评审、批准和实施，并记录变更的历史记录和影响。例如，当有新的功能需求提出时，需要提交一个变更请求，经过评审、批准后才能实施变更，并记录变更历史和影响。

### 7.2 测试计划

确定测试的范围，包括功能测试、性能测试、安全测试等，以及测试的对象和版本。采用自动化测试和手动测试相结合的策略，优先执行单元测试、集成测试，然后进行系统测试和验收测试。

准备测试用例，例如需要搜索的产品列表，需要加入购物车的列表，以涵盖项目中的各种情况。准备异常情况测试数据包含压力测试等，以确保系统在各种情况下都能正常运行。

先执行单元测试，确保各个模块的功能正确性。在本项目中即为测试各页面显示是否正常，操作逻辑是否正常，可能包括用户注册、产品搜索、购物车功能等的测试。在每个单元开发完成后进行单元测试。

然后进行集成测试，验证模块之间的交互是否正常，检查页面跳转是否正常。最后进行系统测试，确保整体功能符合预期。在每个迭代周期之后安排一次集成测试，并在项目接近发布时进行一次全面的系统测试。

### 7.3 归档计划

归档项目的所有关键文档和记录，包括需求文档、设计文档、测试报告、项目计划和会议记录等。将归档文档存储在安全可靠的位置，确保文档的完整性、可用性和机密性。

### 7.4 质量保证计划

制定质量标准，包括功能性、性能、可靠性、安全性等方面的标准，以确保产品符合客户需求和期望。实施质量控制活动，包括质量检查、测试和审查，及时发现和纠正质量问题。

### 7.5 评审和审计计划

定期进行代码评审，发现潜在的代码缺陷和改进机会，并提供反馈意见。定期进行功能评审，避免出现功能缺陷。进行系统审计，评估项目的安全性、可靠性和合规性，确保项目达到预期的标准和要求。

### 7.6 问题解决计划

建立问题跟踪系统，追踪和记录项目中出现的问题和缺陷，并由相关责任人进行解决。制定问题解决流程，包括问题的报告、分析、优先级确定和解决方案的实施，以确保问题及时得到解决。

### 7.7 过程提升计划

收集项目过程的度量数据，包括进度、成本、质量和风险等，用于评估项目的绩效和改进机会。基于过程度量数据，识别项目中存在的问题和瓶颈，并制定改进措施和计划，持续改进项目的执行效率和质量。

## 8. 附加计划