目录

[引言 2](#_Toc192254693)

[1.1编写目的 2](#_Toc192254694)

[1.2项目背景 2](#_Toc192254695)

[1.3定义 2](#_Toc192254696)

[1.4参考资料 3](#_Toc192254697)

[2．可行性研究的前提 3](#_Toc192254698)

[2.1要求 3](#_Toc192254699)

[2.2目标 4](#_Toc192254700)

[2.3条件、假定和限制 4](#_Toc192254701)

[2.4可行性研究方法 4](#_Toc192254702)

[2.5决定可行性的因素 4](#_Toc192254703)

[3.技术可行性分析 4](#_Toc192254704)

[3.1系统简要描述 4](#_Toc192254705)

[3.2处理流程和数据流程 5](#_Toc192254706)

[4.经济可行性分析 5](#_Toc192254707)

[4.1支出 5](#_Toc192254708)

[4.2效益 5](#_Toc192254709)

[4.3投资回收周期 6](#_Toc192254710)

[4.4敏感性分析 6](#_Toc192254711)

[5.社会因素可行性分析 6](#_Toc192254712)

[5.1法律因素 6](#_Toc192254713)

[5.2用户使用可行性 6](#_Toc192254714)

[6.结论意见 7](#_Toc192254715)

7.CASE选取..........................................................................................................................................................7

8.软件过程模型分析与风险分析………………………………………………………………………………………………7

# 引言

## 1.1编写目的

随着互联网技术的快速发展和普及，博客作为一种重要的信息分享和交流平台，受到了越来越多用户的青睐。个人、企业和组织通过博客发布内容、分享见解、推广产品和服务，博客网站已成为互联网生态中不可或缺的一部分。然而，现有的博客平台在功能、用户体验和个性化需求方面仍存在诸多不足，无法完全满足用户多样化的需求。

本可行性研究报告旨在分析和评估开发一个新型博客网站系统的可行性。该系统将致力于解决现有平台的痛点，提供更加灵活、高效和个性化的博客服务。通过本报告的分析，我们希望为博客网站系统的开发提供清晰的指导和决策支持，确保项目能够顺利实施并取得成功。

## 1.2项目背景

1.开发软件名称：博客网站系统

2.项目任务提出者：山东大学计科软件工程开发小组

3.项目开发者：山东大学计科软件工程开发小组

4.用户：博客网站用户

5.实现软件单位：山东大学计算机科学与技术学院

6.项目与其他软件，系统的关系：

本项目采用浏览器和服务器架构模式，前端采用了HTML编程语言技术进行实现的，运行在用户的移动设备的浏览器上；后端我们采用Java开发语言和的技术框架进行开发的；数据库我们采用了MySQL数据库。

## 1.3定义

HTML: HTML的全称为超文本标记语言，是一种标记语言。它包括一系列标签．通过这些标签可以将网络上的文档格式统一，使分散的Internet资源连接为一个逻辑整体。HTML文本是由HTML命令组成的描述性文本，HTML命令可以说明文字，图形、动画、声音、表格、链接等。

CSS: CSS为HTML标记语言提供了一种样式描述，定义了其中元素的显示方式。CSS在Web设计领域是一个突破。利用它可以实现修改一个小的样式更新与之相关的所有页面元素。

JavaScript：JavaScript（简称“JS”） 是一种具有函数优先的轻量级，解释型或即时编译型的编程语言。虽然它是作为开发Web页面的脚本语言而出名，但是它也被用到了很多非浏览器环境中，JavaScript 基于原型编程、多范式的动态脚本语言，并且支持面向对象、命令式和声明式（如函数式编程）风格。

SpringBoot：Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。通过这种方式，Spring Boot致力于在蓬勃发展的快速应用开发领域成为领导者。

MySQL：MySQL是一个关系型数据库管理系统。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件之一。

Mybatis：目前为止最为简单的持久层框架之一，小巧并且简单易学。mybatis本身专注于SQL语句本身。它将SQL语句写在xml文件之中，几乎是彻底将程序代码与SQL语句隔离开，耦合度相当低。因此在SQL语句的编写上它相当的灵活。可以随时根据业务的要求变更SQL语 句而不需要动源程序。

## 1.4参考资料

《软件工程》，钱乐秋，清华大学出版社

《软件工程导论》，张海藩，清华大学出版社

软件工程可行性研究报告（ISO 标准）：https://wenku.baidu.com/view/56077f31f111f18583d05ac1.html

# 2．可行性研究的前提

## 2.1要求

1.主要功能：实现注册用户发表、管理文章，网站管理员进行相关的管理工作，不同用户之间的评论等。主要包括用户注册及登录模块、后台管理模块、用户发表、管理文章等模块。

2.性能要求：要求性能稳定，不能出现数据丢失等情况，显示数据要完整，对服务器上的数据必须进行及时正确的刷新。

3.输出要求：数据完整、详细，简便，快速，实时，显示出各个博客的内容更新、评论更新等

4.输入要求：用户进行规则内的博客发表、查询，评论的发表等。

## 2.2目标

实现注册用户发表、管理文章，网站管理员进行相关的管理工作，不同用户之间的评论等。主要包括用户注册及登录模块、后台管理模块、用户发表、管理文章等模块。

（1）用户注册及登录模块，实现用户的注册、登录等操作，登录成功时进入各自的个人博客页面。

（2）管理员的后台管理模块，主要实现管理员对相关信息的修改、查询、删除。

（3）用户管理模块，用户发表、修改和删除文章；浏览博客时可以对个人及他人博客的文章进行评论。

## 2.3条件、假定和限制

软件寿命：1年

经济来源：自筹

硬件环境：云端服务器，本地PC机

运行环境：Linux

数据库：MySQL

法律和政策方面的限制：严禁违反宪法确定的基本原则，严禁危害国家安全 ,泄露国家秘密 ,颠覆国家政权 ,破坏国家统一。严禁损害国家荣誉和利益。严禁传播发布会妨害第三方权益的文件或者信息包括但不限于病毒代码，黑客程序，软件破解注册信息。严禁抄袭剽窃他人作品。符合双方合同多签署的条款。

开发工具需求：

前端开发工具： WebStorm

后端开发工具：IDEA

数据库管理工具等

## 2.4可行性研究方法

复查项目目标和规模，研究目标正使用的系统等

## 2.5决定可行性的因素

成本和效益对比，学习收益大于小额成本

技术可行，现有技术可承担开发任务。

操作可行，软件能被开发人员和用户接受。

# 3.技术可行性分析

## 3.1系统简要描述

前端：使用HTML定义页面结构与内容，CSS控制样式布局，实现跨设备响应式设计。通过JavaScript开发动态交互功能（如表单验证、异步数据加载），并整合现代前端框架（如Vue.js/React）提升开发效率。

后端：基于Spring Boot快速搭建RESTful API服务，利用其自动化配置特性简化依赖管理与部署流程。通过MyBatis实现数据持久化，将SQL语句与Java代码解耦，支持灵活调整数据库操作逻辑。

数据库：采用MySQL作为核心关系型数据库，存储用户信息、业务交易记录等高一致性数据。结合Redis缓存热点数据（如用户会话、商品信息），降低数据库访问压力。

## 3.2处理流程和数据流程

注册账号结果

注册账号

用户

博客软件

用户

修改信息结果

修改信息

发表博客结果

发表博客

发表评论

发表评论结果

查询、管理结果

内容查询、管理

# 4.经济可行性分析

## 4.1支出

(1)开发者成本：PC机、网络设备、辅助配置、服务器

(2)管理员运维成本

(3)宣传费用

(4)数据库维护成本

(5)人工费用

(6)其他不可知额外支出

## 4.2效益

本软件开发的目的在于掌握相关知识，熟悉课程内容，因此并不产生额外的效益，只是作为训练的目的使用。在其中测试各种功能和实现各种软件设计。

## 4.3投资回收周期

1个学期

## 4.4敏感性分析

处理速度：

一般查询速度<4s

关键数据查询速度<4s

# 5.社会因素可行性分析

## 5.1法律因素

所有软件都选用正版。

所有技术资料都合法。

## 5.2用户使用可行性

在评估系统的用户使用可行性时，重点考察了目标用户的需求、接受度以及操作体验。通过与用户群体的沟通和调研，确认系统功能能够满足其主要需求，且界面设计简洁直观，符合用户的操作习惯。此外，系统支持团队将提供持续的技术支持和维护，确保用户在使用过程中遇到的问题能够及时解决。综合来看，用户对系统的接受度较高，且系统的引入不会对现有工作流程造成重大影响，因此用户使用可行性较高。

# 6.结论意见

由于学习效益高, 技术、经济、操作、指导都有可行性，可以进行开发。

**7.CASE选取**

基于项目需求与技术路线（假设为Java Web应用开发，结合团队规模（5人组）与成本考量，选取了以下CASE工具组合以协助开发：

.图稿绘制工具：Microsoft Visio

其深度集成Microsoft 365生态和行业标准符号库提升系统设计文档的专业性。项目需频繁迭代UML时序图与BPMN流程图以支持需求评审，Visio的VBA脚本自动化可批量生成类图，减少重复劳动。相较SmartDraw，其动态数据绑定功能更契合复杂工程场景。

.源码浏览工具：Understand

针对Java代码库的依赖分析与代码度量需求，Understand提供跨语言支持和调用关系可视化，帮助团队快速定位代码问题所在。相较Source Insight，其UML图生成与架构视图功能更利于技术评审中的逻辑展示。

.配置管理工具：Git

采用分布式版本控制与分支策略，支持并行开发与敏捷迭代。结合GitLab实现CI/CD流水线，规避SVN集中式架构的单点故障风险。Git的轻量化分支操作与冲突解决工具更适合高频提交的小型团队。

.数据库工具：MySQL Workbench

项目采用MySQL作为核心数据库，Workbench的ER模型逆向工程可直接从现有库生成ER图，其可视化查询构建器降低复杂SQL编写门槛。相比通用工具，其性能优化建议与迁移向导更贴合MySQL特性。

.UML建模工具：Visual Paradigm

在敏捷开发中需快速响应需求变更，Visual Paradigm的Scrum板集成与代码双向工程缩短设计-开发周期。PlantUML虽轻量但依赖文本描述，难以满足非技术成员的评审需求；而StarUML缺乏团队协作功能。Visual Paradigm的云端协作与版本历史追溯保障了模型一致性。

总结：

本方案平衡了功能性、协作性与生态适配，工具间通过自动化接口（Visio数据绑定→Git版本控制→Visual Paradigm代码生成）形成闭环，降低人工干预风险。初期学习成本可通过模板复用与培训文档缓解。

**8. 软件过程模型分析与风险分析**

传统开发模型：

瀑布模型：开发过程是通过设计一系列阶段顺序展开的，从系统需求分析开始直到产品发布和维护，每个阶段都会产生循环反馈，如果发现问题可以返回上一个阶段，从一个阶段“流动”到下一个阶段。结构清晰且风险可控。

但缺点在于交付周期长，需要等待完全开发，并且需要有明确的需求才能使用，并且到了后期修改成本高，需要进行回溯修改。

适合需求稳定且相对较小的项目

相比于其他两个传统开发模型更适合小型的项目

螺旋模型：兼顾了快速原型的迭代的特征以及瀑布模型的系统化与严格监控，最大的特点在于引入了其他模型不具备的风险分析，灵活适应需求并且有着渐进式交付。

缺点在于多轮原型开发和风险评估需要消耗的资源大，依赖团队的风险分析能力和管理复杂。

适合用作复杂的大项目，不太适合个人博客的简单开发

迭代增量模型：将系统拆解为多个功能模块（增量），分批次开发并集成。

有快速交付价值：用户可以快速优先地使用核心功能，然后补全剩余功能。

并且每次增量可独立测试，避免大爆炸式的集成问题，灵活性较高。

缺点在于需要预先规划好模块化接口，否则后期增量难以兼容，并且可能会带来冗余的重复性工作造成人力物力浪费。

适合需要做较长时间的大型项目，不适合本次个人博客软件

敏捷开发模型：轻文档, 客户参与, 拥抱变化,人与人的沟通

响应变化效率高：通过每日站会和迭代评审快速调整需求优先级

客户能够深度参与，便于确保产品价值导向，这是传统开发模型所不具备的

侧重可运行软件而非详尽文档，适合需求模糊或快速变化的领域

缺点在于需要高度自组织的团队，并且规模化和合规性具有一定挑战

适合需求多或容易变化的互联网产品，如Web应用等

因此此次个人博客软件开发倾向于采用敏捷开发模型

如何应用进个人博客软件开发项目中？

产品负责人和开发团队是5人小组，Scrum Master可省略，每个迭代周期（Sprint）设定明确目标，避免任务堆积。

从用户视角分析项目，比如：

如果是作为访客访问，可以做到浏览博客及其评论

如果是作为已经登录的作者，可以自由发表博客和评论

如果是作为管理员，可以删除博客或者评论等。

初步设立周期为1-2周（sprint），Sprint评审可以自我验收或邀请朋友。

**风险分级及应对预案**

**三级风险：发生概率低或者影响比较轻微的情况**

性能限制瓶颈：某个时间段访问次数较多，导致服务器性能无法匹配相应需求，出现的访问超时、无法加载的情况。

应对方案：采用云服务器的弹性扩容，或者将一些静态资源托管至CDN缓存

遭遇暴力攻击：注册登录接口等被暴力攻击、破解，密码较弱的账户被盗。

应对方案：采用流量速率限制，使用Nginx限制同一IP的访问，或采取其他人机验证方法

**二级风险：发生概率中等或者具有中等影响后果的情况**

时间估计偏差：错估前端交互的开发时间

应对方案：每个Sprint预留一定的剩余时间对应。

个人精力被分散导致开发受影响：

应对方案：每天固定一小时用于开发的时间

部署失效：本地开发正常但是在服务器上的部署因为版本等原因失效

应对方案：采用自动化部署脚本等

XSS攻击：用户评论未过滤HTML标签，导致恶意脚本注入

应对方案：采取输入过滤或使用CSP限制来源

第三方服务失效：依赖的系统因限流导致功能中断等。

应对方案：当第三方服务不可用时切换至静态页面显示“评论维护中”或实现故障隔离。

**一级风险：发生概率较高并且影响后果严重的情况**

数据丢失：数据库误删或未备份，导致所有文章及评论内容丢失。

应对方案：每日定时导出SQL数据到云存储

需求蔓延：开发中不断添加新功能，导致无法按期交付。

应对方案：采用需求冻结，迭代周期内禁止新增开发需求。