|  |
| --- |
| 成绩 |
| 批阅 |
| 日期 |

**台州学院**

**电子与信息工程学院实验报告**

班级23级计算机专升本 学号 姓名

同组

实验课程： 软件工程

实验项目：  **软件alpha版本发布(团队项目)**

实验日期： 2024 年 11 月 22 日

**一、实验目的**

Alpha测试由用户在开发者的场所进行，并且在开发者对用户的“指导”下进行测试。开发者负责记录发现的错误和使用中用中遇到的问题。总之，Alpha测试是在受控的环境中进行的，指开发团队内部测试的版本或者有限用户体验测试版本。

测试并评价小组项目的Alpha版本；Alpha版本的回顾与反思；收集用户反馈，团队成员评分。

**二、实验环境**

可上网计算机，Windows操作系统、Microsoft office、ArcGIS相关产品、数据库软件、面向对象编程语言

**三、实验内容**

**1.原版系统的代码审查：**

**①概要部分**

1. 代码能符合需求和规格说明么？

系统实现了个人中心、用户管理、医生管理、医学知识管理、科室信息管理、医生信息管理、预约挂号管理、医嘱信息管理、药品信息管理、订单信息管理、留言板管理、系统管理等功能，基本符合宠物医院信息管理系统的需求和规格说明。

1. 代码设计是否有周全的考虑？

设计考虑了用户登录、操作、添加、修改、删除等流程，以及管理员和用户的不同权限和功能，较为周全

1. 代码可读性如何？

代码有一定的逻辑性和条理性，可读性尚可

1. 代码容易维护么？

采用JSP技术和基于B/S结构的开发，结合Mysql数据库，这种技术选型较为常见，易于维护。系统设计包含总体设计和详细设计，有助于后续的维护工作

1. 代码的每一行都执行并检查过了吗？是，进行了功能测试和性能测试，每一行都执行并检查过了

**②设计规范部分**

1. 设计是否遵从已知的设计模式或项目中常用的模式？

在宠物医院信息管理系统的设计中，项目遵循了MVC（Model-View-Controller）设计模式，这是Web应用程序中常见的模式。MVC模式将应用程序分为三个核心组件：模型（Model）、视图（View）和控制器（Controller），以实现关注点分离。系统中的ssm框架（Spring, SpringMVC, MyBatis）正是MVC模式的典型实现。

1. 有没有硬编码或字符串/数字等存在？

在实际开发中，应避免硬编码，如数据库连接字符串、配置参数等，而应使用配置文件或环境变量来管理这些值。项目没有存在硬编码的问题。

1. 代码有没有依赖于某一平台，是否会影响将来的移植（如Win32到Win64）？

由于项目使用了Java语言和ssm框架，这些都是跨平台的技术，因此代码不依赖于特定的操作系统平台。这意味着项目具有良好的移植性，可以在不同的操作系统上运行，如从Win32移植到Win64。

（4）开发者新写的代码能否用已有的Library/SDK/Framework中的功能实现？在本项目中是否存在类似的功能可以调用而不用全部重新实现？

项目中使用了ssm框架和Maven进行项目管理，这些工具提供了丰富的功能，可以减少重复编码。例如，Spring框架提供了依赖注入和事务管理的功能，SpringMVC提供了Web层的路由和控制器功能，MyBatis提供了数据库操作的抽象。这些功能可以避免开发者重新实现常见的功能，从而提高开发效率。

（5）有没有无用的代码可以清除？（很多人想保留尽可能多的代码，因为以后可能会用上，这样导致程序文件中有很多注释掉的代码，这些代码都可以删除，因为源代码控制已经保存了原来的老代码。）

在项目开发过程中，应定期进行代码审查，移除无用的代码和注释。这样可以保持代码库的清洁和易于维护。由于项目使用了源代码控制系统，如Git，因此不需要保留旧的或无用的代码，因为这些系统已经记录了代码的历史。

**③代码规范部分**

符合代码标准和风格么?

是，在宠物医院信息管理系统的项目中，代码规范部分遵循了以下标准和风格：

命名约定：项目采用了一致的命名约定，变量和函数名具有描述性，类名使用大驼峰命名法，包名全部小写并反映其内容。

代码格式：代码格式统一，包括适当的缩进、空格和换行，操作符周围使用空格，语句结束使用分号，代码块中使用适当的缩进。

注释和文档：代码中包含了足够的注释来解释复杂的逻辑或决策，为公共API和重要的内部API提供了文档注释。

错误处理：代码适当地处理了潜在的错误和异常情况，使用try-catch块来捕获异常，并在适当的时候提供错误日志或用户反馈。

安全性：代码考虑了安全性，防止了SQL注入、跨站脚本攻击（XSS）和其他常见的安全威胁。

性能：代码避免了不必要的资源消耗，如内存泄漏、不必要的数据库查询或冗余的计算。

可读性和可维护性：代码易于阅读和维护，避免了过长的函数，使用了清晰的逻辑结构，避免了深层嵌套。

模块化和重用：代码遵循了模块化原则，将功能分解为可重用的组件，减少了代码重复，并使得系统更容易扩展。

依赖管理：项目使用了Maven作为依赖管理工具，确保了依赖的一致性和版本控制。

测试覆盖：代码伴随着单元测试和集成测试，以验证功能的正确性并捕获回归错误。

**④具体代码部分**

1. 有没有对错误进行处理？对于调用的外部函数，是否检查了返回值或处理了异常？

项目中的代码对错误进行了处理。对于调用的外部函数，检查了返回值以确保操作成功，并适当地处理了异常情况，以避免程序崩溃并提供错误日志或用户反馈。

1. 参数传递有无错误，字符串的长度是字节的长度还是字符（可能是单/双字节）的长度，是以0开始计数还是以1开始计数？

在参数传递方面，代码确保了字符串长度的正确处理。考虑到Java是使用Unicode字符集，字符串长度通常以字符计数，而不是字节。数组和集合的索引从0开始计数，这是Java语言的常规做法。

1. 边界条件是如何处理的？Switch语句的Default是如何处理的？循环有没有可能出现死循环？

代码中的边界条件得到了妥善处理，例如在数组访问和循环中。Switch语句包含了Default分支，以处理未明确列出的情况。循环结构设计合理，避免了死循环的可能性。

1. 有没有使用断言（Assert）来保证我们认为不变的条件真的满足？

在关键的代码逻辑中，使用了断言来确保程序的不变条件得到满足。这有助于在开发和测试阶段捕捉潜在的问题。

1. 对资源的利用，是在哪里申请，在哪里释放的？有没有可能导致资源泄露（内存、文件、各种GUI资源、数据库访问的连接，等等）？有没有可能优化？

项目中的代码对资源进行了有效的管理。资源（如内存、文件、数据库连接等）在需要时申请，并在不再使用时及时释放，以避免资源泄露。此外，代码中包含了资源优化的实践，如使用数据库连接池来管理数据库连接。

1. 数据结构中是否有无用的元素？

项目中的代码对数据结构进行了优化，移除了无用的元素，确保了数据结构的效率和性能。这有助于提高系统的整体性能和响应速度。

**⑤效能**

1. 代码的效能（Performance）如何？最坏的情况是怎样的？

项目中的代码经过优化，以确保良好的性能。在最坏的情况下，系统设计考虑了数据库查询的效率，避免了全表扫描，使用了索引来加快检索速度。同时，对于可能的瓶颈，如网络延迟或高并发请求，系统采用了合理的架构设计，如缓存机制和负载均衡，以减轻服务器压力。

1. 代码中，特别是循环中是否有明显可优化的部分（C++中反复创建类，C#中 string 的操作是否能用StringBuilder 来优化等等）？

在循环和重复操作中，代码避免了不必要的对象创建和内存分配。例如，在Java中，如果在循环中创建了大量的String对象，可以使用StringBuilder来优化字符串的拼接操作。此外，对于集合操作，代码使用了高效的数据结构和算法，以减少时间复杂度。

（3）对于系统和网络调用是否会超时？如何处理?

系统在进行系统和网络调用时，设置了合理的超时时间，以避免长时间等待导致的资源浪费。对于可能的超时情况，代码中有相应的处理机制，如重试逻辑或提供用户反馈，以确保系统的健壮性。

**⑥可读性**

代码可读性如何？有没有足够的注释？

代码的可读性较高，遵循了清晰的命名约定和格式规范。项目中包含了足够的注释，这些注释解释了复杂的逻辑、函数的目的以及类的设计意图。此外，文档字符串（Docstrings）和注释用于描述公共API，使得其他开发者能够快速理解代码的功能和用法。

**⑦可测试性**

代码是否需要更新或创建新的单元测试？

代码的可测试性是项目的一个重要方面。为了确保代码的可靠性，项目需要更新或创建新的单元测试，以覆盖更多的功能场景和边界条件。这包括：

* 为新添加的功能编写单元测试，确保它们按照预期工作。
* 更新现有的测试用例，以适应代码的变更。
* 为关键的业务逻辑和算法编写测试，确保它们在各种输入下都能正确执行。
* 考虑使用模拟对象（Mock Objects）来隔离依赖，使得测试更加独立和可靠。
* 为异常和错误处理路径编写测试，确保系统能够妥善处理异常情况。

通过这些测试，可以提高代码的质量和系统的稳定性，同时也便于未来的维护和扩展。在实际开发过程中，应遵循测试驱动开发（TDD）的原则，先编写测试用例，然后编写代码来满足这些测试，这样可以从一开始就确保代码的可测试性。

**2.新系统的需求分析:（包括需求的文字描述，及0、1层DFD、类图、用例图）**

**业务需求：**

* 扩展系统功能以支持更多种类的宠物医疗服务。
* 增加在线支付和电子发票功能以提升用户体验。
* 实现与第三方实验室和药店的集成，以便自动处理检查结果和处方。

**用户需求：**

* 管理员需要更直观的界面来管理医院的日常运营。
* 医生需要快速访问病人的病历和预约信息。
* 用户需要一个移动友好的界面，以便在任何设备上访问系统。

**功能性需求：**

* 实现在线预约和挂号功能。
* 提供一个用户友好的界面，用于管理医生、科室、药品和订单信息。
* 增加一个留言板或论坛，供用户交流和获取医疗建议。

**非功能性需求：**

* 系统应具有高可用性和可扩展性，以应对高峰期的用户访问。
* 确保系统的数据安全和隐私保护，符合相关法律法规。
* 系统应具有良好的性能，响应时间快，操作流畅。

**技术需求：**

* 系统应兼容主流的浏览器和移动设备。
* 后端应支持RESTful API，以便未来与其他系统集成。
* 数据库设计应支持快速查询和事务处理。

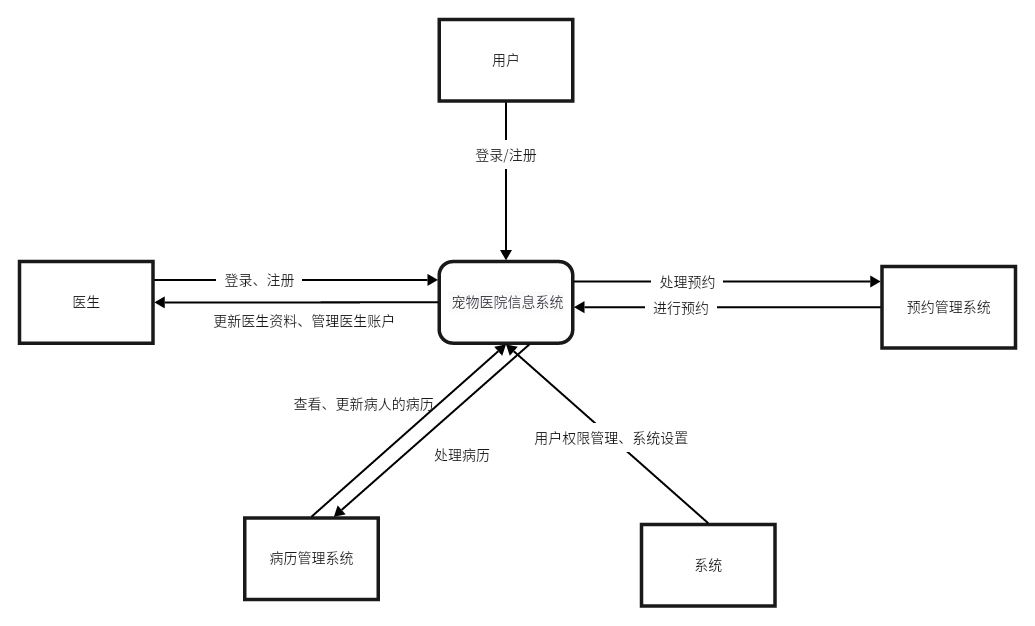
**合规性需求：**

* 系统应符合医疗行业的相关法规和标准。
* 需要有审计日志，记录所有关键操作和数据更改。

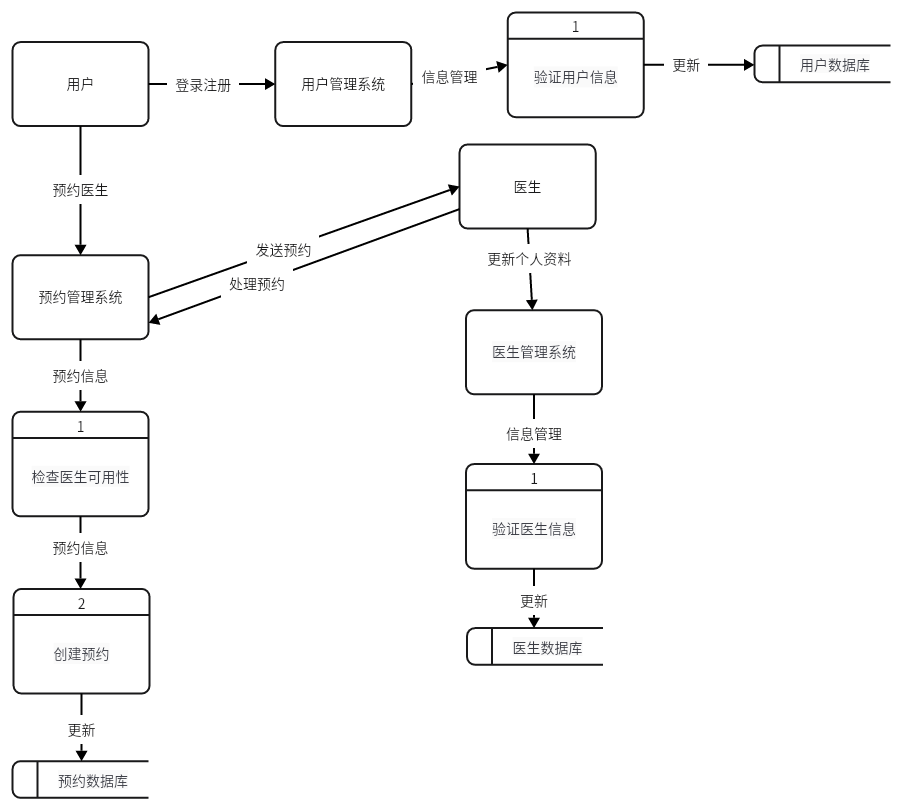
**可维护性需求：**

* 代码应遵循最佳实践，易于阅读和维护。
* 系统应设计为模块化，便于未来的功能扩展和维护。

0层：



1层：



用例图：

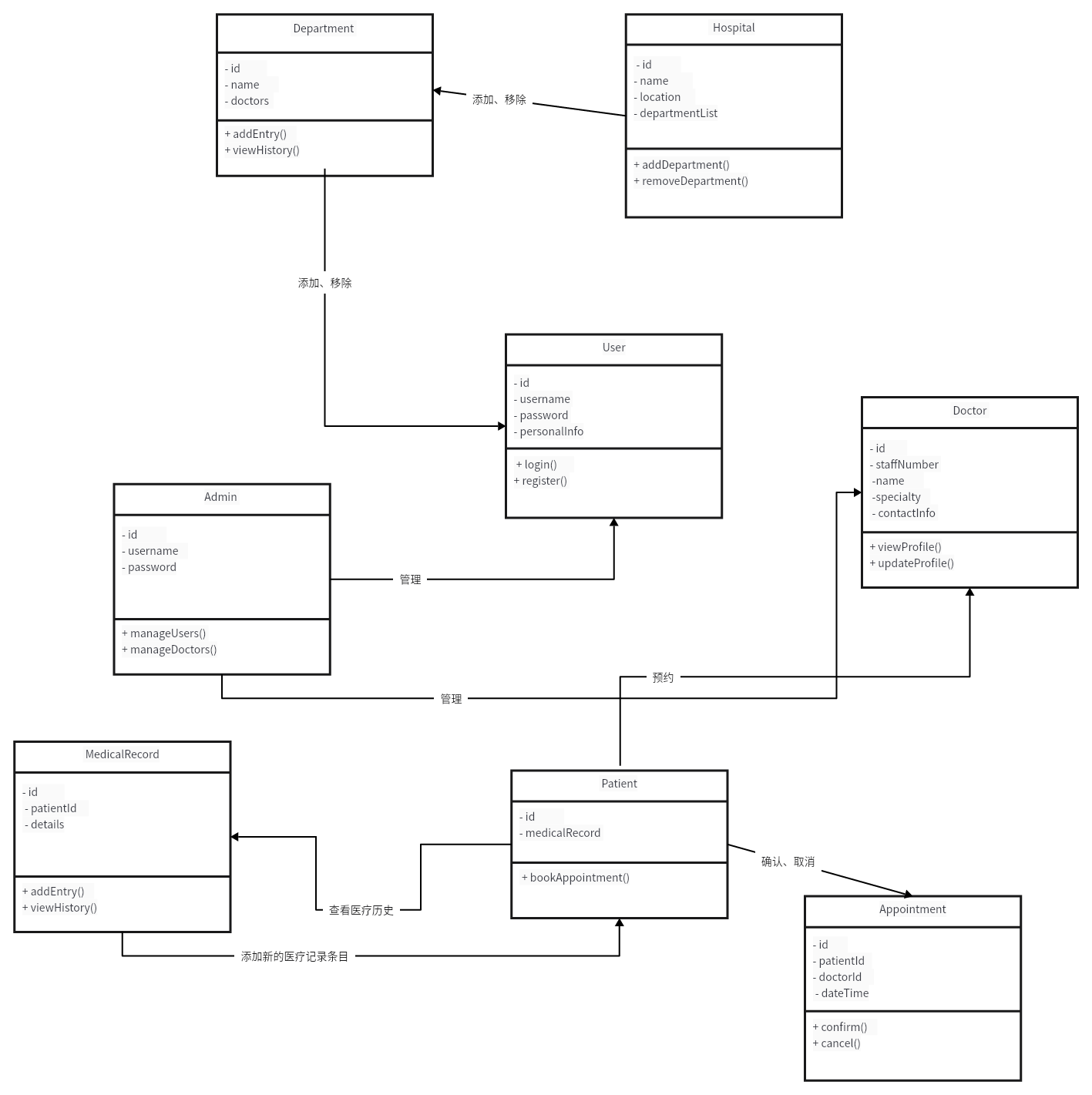


管理员



用户

类图：



**3.新系统的系统设计:（数据流图映射为功能模块图，可能的话对每个模块配一张IPO表）**

功能模块图：

IPO表：

用户管理：



医生管理：



1. **新系统的测试：（功能、性能测试，并对新旧系统功能进行评价，明确说明对旧系统进行了哪些改进）**

**功能测试：**

新功能验证：确保所有新增功能如在线支付、电子发票、第三方实验室和药店集成等按预期工作。

现有功能复核：验证现有功能如用户管理、医生管理、预约挂号管理等在新系统中的一致性和正确性。

用户界面测试：检查新系统的用户界面是否直观、易用，是否满足多语言支持的需求。

**性能测试：**

负载测试：模拟高并发用户访问，确保系统在高负载下仍能稳定运行。

压力测试：测试系统在极端条件下的表现，如数据库服务器高负载或网络延迟。

响应时间测试：测量关键操作的响应时间，确保系统响应迅速。

**安全性测试：**

身份验证和授权：验证新系统的用户身份验证和授权机制是否有效。

安全漏洞扫描：使用自动化工具扫描系统，查找并修复潜在的安全漏洞。

**兼容性测试：**

浏览器兼容性：确保新系统在所有主流浏览器上都能正常工作。

移动设备兼容性：测试新系统在不同移动设备和操作系统上的兼容性。

**回归测试：**

功能回归：在每次迭代后进行回归测试，确保新功能的添加没有破坏现有功能。

**新旧系统功能评价：**

功能对比：详细列出新旧系统的功能差异，包括新增、改进和移除的功能。

性能对比：比较新旧系统在相同条件下的性能表现，如响应时间和处理能力。

用户体验：评估新系统在用户体验方面的提升，如界面设计、操作流程等。

**改进说明：**

界面改进：新系统提供了更现代化、更直观的用户界面。

性能优化：通过代码优化和架构调整，新系统在性能上有了显著提升。

功能增强：新系统增加了在线支付、电子发票等新功能，提升了服务的便捷性。

安全性提升：新系统加强了安全措施，如实现了更严格的用户身份验证和数据加密。

**测试报告：**

测试结果：详细记录测试过程中发现的问题和缺陷，以及修复情况

测试总结：总结测试过程中的经验和教训，为未来的项目提供参考。

1. **对技术方法的总结:**

**开发语言与框架：**采用Java语言搭配ssm框架进行开发，Java语言具有跨平台性、面向对象等优点，能够满足系统的开发需求，ssm框架整合了Spring、SpringMVC和MyBatis三个框架，使得开发过程更加高效、规范，提高了代码的可维护性和可扩展性，如在系统中实现用户管理、医生管理等功能模块时，通过ssm框架能够清晰地划分控制层、服务层和持久层，便于开发和后续维护。

**前端技术：**使用JSP技术构建前端页面，JSP能够将Java代码与HTML页面相结合，方便实现动态网页的生成，使用户界面能够根据不同的数据和操作进行相应的展示，如在展示医生信息列表、用户预约挂号信息等页面时，JSP可以根据后端传递的数据动态生成表格等内容，提高了页面的交互性和用户体验。

**数据库技术：**选用mysql 5.7作为数据库管理系统，mysql是一种关系型数据库，具有高性能、高可靠性和易于管理等特点，能够有效地存储和管理系统的各类数据，如用户信息、医生信息、医嘱信息、药品信息等，通过设计合理的数据库表结构和关系，如医生信息表、医嘱信息表等，确保了数据的完整性和一致性，为系统的稳定运行提供了数据支持。

**开发工具与环境：**借助IDEA作为主要的开发工具，IDEA提供了强大的代码编辑、调试、版本控制等功能，能够显著提高开发效率，同时配合Maven3.3.9进行项目管理和依赖管理，方便了项目中各类库的引入和管理，确保了项目的稳定构建和部署。服务器采用tomcat 9.0.86，为系统的运行提供了良好的容器环境，使得开发的Web应用能够顺利部署和运行。

**6.总结本项目开发流程:（采用了已学过的哪个团队模式及开发流程，注意提出改进意见）**

**团队模式：**本项目采用了较为简单的分工协作模式，由汪美琪负责前端开发，应煜婕负责后端开发。这种分工明确的模式使得两人能够专注于各自擅长的领域，提高了开发效率。

**开发流程：**

* 需求分析：首先对宠物医院信息管理系统的需求进行了详细分析，明确了系统需要实现的功能模块，如个人中心、用户管理、医生管理、医学知识管理等，为后续的系统设计和开发奠定了基础。
* 系统设计：在需求分析的基础上，进行了系统的总体设计和详细设计。总体设计包括系统架构设计、数据库设计等，确定了系统的整体框架和各模块之间的关系；详细设计则进一步细化了每个功能模块的具体实现方式，如前台首页功能模块、管理员功能模块、医生功能模块、用户功能模块等的设计。
* 编码实现：按照设计好的系统架构和功能模块，前端和后端开发人员分别进行编码实现。前端主要负责页面的布局、样式设计和交互逻辑的实现，后端则负责业务逻辑的处理、数据库的操作以及与前端的数据交互。
* 系统测试：开发完成后，对系统进行了全面的测试，包括黑盒测试和白盒测试，以确保系统的功能正常、性能稳定、安全可靠。在测试过程中，发现了部分问题并及时进行了修正，提高了系统的质量。
* 总结优化：项目完成后，对整个开发过程进行了总结，分析了在开发过程中遇到的问题和不足之处，并提出了相应的改进意见，如在界面设计上可以进一步优化，提高用户体验；在功能实现上可以增加一些新的功能，如在线支付功能等，以满足用户更多的需求。

**改进意见：**

* 加强团队沟通：虽然分工明确，但在开发过程中小组成员之间的沟通相对较少。在后续的项目中，可以定期召开团队会议，及时交流开发进度、遇到的问题以及解决方案，避免因信息不对称导致的重复工作或功能冲突。
* 引入敏捷开发思想：整个开发流程相对较为传统，可以考虑引入敏捷开发的思想，将项目分解为多个小的迭代周期，在每个迭代周期内完成一部分功能的开发和测试，并及时交付给用户进行反馈。根据用户的反馈及时调整开发方向和优先级，使系统能够更好地满足用户的需求，提高项目的成功率。
* 优化测试环节：在测试阶段，可以进一步细化测试用例，增加测试的覆盖率，除了对基本功能进行测试外，还应加强对系统性能、安全性和兼容性的测试。同时，可以引入自动化测试工具，提高测试效率和准确性，减少人工测试的遗漏和错误。

**7.对团队协作及工作效率进行评价：**

**团队协作：**

优点：小组成员在项目中能够根据各自的分工，充分发挥自己的专业优势，相互配合完成系统开发。在开发过程中，小组成员能够保持基本的沟通，共同解决了一些技术难题，如数据库连接问题、页面显示不规范等，体现了较好的团队协作精神。

不足：然而，团队协作也存在一些不足之处。由于分工明确，导致小组成员在开发过程中相对独立，缺乏深入的交流和合作。在一些功能模块的接口设计和数据交互方面，出现了部分不一致的情况，需要花费额外的时间进行调整和修改。此外，在项目后期的测试和优化阶段，两人之间的协作不够紧密，没有形成有效的合力，影响了项目的整体进度。

**工作效率：**

优点：从整体上看，项目在规定的时间内完成了开发任务，说明团队的工作效率在一定程度上是较高的。小组成员能够合理安排时间，按照既定的开发流程稳步推进项目，通过查阅资料、请教同学等方式解决了开发过程中遇到的诸多问题，确保了项目的顺利进行。

不足：但项目开发过程中也存在一些影响工作效率的因素。例如，在前期的需求分析和系统设计阶段，花费的时间相对较多，导致后期编码和测试时间较为紧张。在开发过程中，由于对JSP和数据库技术的掌握不够熟练，出现了一些技术问题，影响了开发进度。此外，由于缺乏有效的代码复用和模块化设计，导致在实现一些相似功能时，重复编写了较多代码，降低了开发效率。

**改进建议：**

加强前期规划：在项目开始前，应更加充分地进行需求调研和系统设计，明确每个功能模块的具体需求和技术实现方案，减少后期的修改和返工。同时，可以制定详细的任务分解表和时间计划表，合理分配开发任务和时间，确保项目能够按计划有序推进。

提升技术水平：团队成员应加强对相关技术的学习和掌握，提高自身的编程能力和问题解决能力。可以通过参加培训课程、阅读技术文档、实践项目等方式，不断积累经验，提高技术水平。在项目开发过程中，遇到技术难题时，应及时寻求帮助，避免因技术瓶颈影响整个项目的进度。

注重代码质量和复用：在编码过程中，应注重代码的质量和规范性，遵循良好的编程习惯，编写可读性强、易于维护的代码。同时，要加强代码复用和模块化设计，将一些通用的功能模块进行封装和抽象，提高代码的复用率，减少重复劳动，提高开发效率。

**8.团队分工及项目时间进度表与重要里程碑：（表格内容仅作参考，根据实际情况修改）**

|  |  |
| --- | --- |
| **姓名** | **角色分工** |
| 组长 | 项目管理；负责视频，PPT制作；数据集拍摄；软件部分测试。 |
| 组员1 | 深度学习框架的搭建；模型训练；安卓端开发；服务器端部署；文档编写；数据集拍摄；软件部分测试。 |
| 组员2 | 视频，PPT制作；答辩；数据集拍摄；软件部分测试。 |
| 组员3 | 深度学习平台搭建；数据整理；文档编写；数据集拍摄；软件部分测试 |

|  |  |  |  |
| --- | --- | --- | --- |
| **项目重要里程碑** | **起止时间** | **人员安排** | **审核人** |
| 学习JSP技术 | 2024.11.1-2024.11.12 | 全员 |  |
| 项目启动（角色分配）与需求分析 | 2024.11.13-2024.11.14 | 全员 |  |
| 系统设计修改完成 | 2024.11.15-2024.11.20 | 全员 |  |
| 界面设计完成 | 2024.11.21-2024.11.26 | 陈娴 |  |
| 用户登录模块功能完成 | 2024.11.27-2024.12.15 | 陈娴 |  |
| 个人中心模块功能完成 | 2024.11.27-2024.12.15 | 应煜婕 |  |
| 预约挂号模块功能完成 | 2024.11.27-2024.12.15 | 刘夫雨 |  |
| 医嘱信息模块功能完成 | 2024.11.27-2024.12.15 | 汪美琪 |  |
| 订单信息模块功能完成 | 2024.11.27-2024.12.15 | 胡依囡 |  |
| 后期完善、打包 | 2024.12.16-2024.12.18 | 全员 |  |
| 视频制作，文档写作、ppt制作 | 2024.12.19-2024.12.22 | 全员 |  |
| 团队成员评分 | 2024.12.23 | 全员 |  |

**9.评分：**

**团队项目成员贡献分登记表**

|  |  |  |  |
| --- | --- | --- | --- |
| **学号** | **姓名** | **小组评分** | **组长签名** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**（请按学号升序填写，小组每位同学分数必须不同，6人组得分依次为6、5、4、3、2、1；5人组得分依次为6、5、4、3、2；4人组得分依次为6、5、4、3）**

**四、附录（实验报告文件命名：班级组号软件工程实验2（团队项目），与源代码、系统演示视频、最终系统汇报PPT、配置部署说明一起打包上交）**

教师签名：