问题清单及解决方案选择：

一. 需求分析不明确：

1. 澄清并明确需求

集合所有利益相关者：确保所有项目利益相关者参与到需求收集过程中来，包括管理层、最终用户、技术团队等。

组织访谈和会议：通过个体访谈或团体研讨会的形式深入讨论需求。

使用调查问卷：对大群体进行调查以获取更多用户需求信息。

2. 初步设计方案

创建原型：初步设计界面原型，可以通过软件工具如Balsamiq、Sketch等快速完成。

对原型进行评审：让利益相关者对原型进行审查，收集反馈。

3. 试做与测试

开发MVP（最小可行产品）：选择一些核心功能进行初步实现。

开展用户测试：让一小部分目标用户测试MVP，收集使用体验反馈。

4. 修改与改进

分析反馈：全面分析用户的反馈和测试的结果。

确定改进建议：将反馈分析成具体的可执行项。

迭代开发：进行产品的功能增强和界面优化。

5. 沟通表达

持续沟通：与所有利益相关者保持定期沟通，更新项目进展。

文档记录：详细记录所有讨论、决定和更改，确保信息透明。

变更管理：对需求的任何更改都要有严格的变更管理流程。

6. 最终实施

培训用户：对用户进行系统使用培训。

部署上线：将系统投入生产环境。

收集反馈：上线后继续收集用户的反馈，以供未来更新和迭代。

二. 技术选型：

通过技术调研和原型验证对比各种技术栈，后端选择使用Java作为主要的开发语言，使用了SpringBoot3+SpringSecurity6+mybatis-plus框架。使用MVC的设计模式，为产品的拓展性提供了保障。

Java结合SpringBoot 3框架可以实现快速开发和自动化配置，大幅提高开发效率。SpringSecurity 6为应用提供了稳固的安全防护，支持灵活且强大的认证和授权机制。同时，mybatis-plus进一步简化了数据库的交互操作，通过提供通用的CRUD接口减少了重复代码的编写。采用MVC设计模式则清晰地区分了应用程序的不同功能层次，有助于代码组织和后期维护。这样的技术组合不仅继承了Java语言本身的成熟和稳定性，还通过各个框架强大的生态支持和社区资源来加速开发过程，保证了项目的可扩展性和可靠性。同时SpringBoot框架也有利于解耦，通过控制反转和依赖注入的基本思想，可以更容易的对程序进行修改和拓展。

前端采用vue+Element,Vue.js 结合 Element UI 库是构建现代网页应用程序的流行选择，两者的组合提供了如下优点：数据双向绑定、组件化、易上手、高效的UI开发、自适应和响应式、良好的社区支持和文档、轻量级和快速、可定制性等。

三. 安全性问题：

方案：在线观影系统采用SHA-256加密算法为用户数据提供强有力的安全保障，以确保敏感信息如密码不会以明文形式存储或传输，降低数据泄露的风险。同时，整合Spring Security进行权限控制，允许系统根据不同用户角色区分接口调用权限，这样能有效地管理用户权限，确保只有授权用户才能访问特定资源或执行特定操作，既维护了系统的安全性，又提高了资源的管理效率和访问控制的精细度。

四. 依赖管理：

方案：使用依赖管理工具，如npm，带来了丰富的优点：它能自动化处理项目的依赖项，保证了版本控制的一致性，简化了库的安装、更新和配置过程。此外，依赖管理工具提高了开发效率，使得开发者能够专注于编码而不是手动管理各个包，同时还通过对依赖树的优化，来减少项目的体积和提升加载速度。最终，这些工具支持了可重复的构建过程，从而在多个环境中保证了应用程序的一致性和稳定性。。

五. 代码版本控制：

方案：使用Github进行版本控制，遵守好的提交习惯，采取合适的分支策略。使得团队成员可以轻松地共享代码、追踪和合并更改，以及维护项目的历史记录。它的Pull Request机制有助于进行代码复审，保证代码质量。此外，GitHub的强大的社区支持和丰富的集成服务（如CI/CD工具）增强了开发流程的自动化和效率。GitHub还具有良好的可视化和项目管理工具，使得非技术用户也能容易地了解项目进展。最后，GitHub作为开源项目的主要托管平台，也便于代码的公开共享和贡献，促进了开放源代码文化的扩展。