

XXXXXXXXXXXX

XXXXXXXXXXXX
XXX XXX XXXXXXXX
XXXXXXX XXXX
XXXX2026 X X X

□□

“” Vue 3 + Vite + Element Plus Spring Boot
MyBatis-Plus /
Spring BootVue 3

Abstract

This thesis presents the design and implementation of a homestay recommendation system. The frontend is built with Vue 3, Vite, and Element Plus, while the backend leverages Spring Boot and MyBatis-Plus, combining collaborative filtering and content-based recommendation to provide personalized listings, online booking, and host property management. The work covers background, requirement analysis, system architecture, key technologies, database and process design, implementation, and testing.

Keywords: Homestay recommendation; Personalized recommendation; Spring Boot; Vue 3; Hybrid recommender

□□

1. □□
2. □□□□□□□□
3. □□□□
4. □□□□
5. □□□□
6. □□□□□ER □□
7. □□□□□□□□□□
8. □□□□□□□□
9. □□□□□□
10. □□□□□□
11. □□□□□□
12. □□

1. 项目背景

1.1 项目概述

本项目旨在开发一个基于微服务架构的在线旅游预订系统。系统核心功能包括用户注册登录、酒店/机票预订、订单管理、支付接口对接等。系统采用Spring Boot + MyBatis-Plus + JWT + Spring Validation + Lombok + MySQL + Redis技术栈。

系统采用微服务架构，使用Spring Cloud Alibaba进行服务治理。数据库采用MySQL，缓存采用Redis。系统支持高并发访问，具备良好的扩展性和稳定性。

系统采用前后端分离模式，前端使用Vue 3 + Vite + Element Plus + Pinia + Vue Router + Axios，后端使用Spring Boot + MyBatis-Plus + JWT + Spring Validation + Lombok + MySQL + Redis。

系统参考了Airbnb、携程、去哪儿网等知名旅游平台的架构设计。系统支持多端接入，包括PC端、移动端、小程序等。系统具备良好的用户体验，操作流程简洁明了。

系统采用分布式部署，支持水平扩展。系统具备良好的容错能力，能够在部分服务故障时保持整体系统的可用性。系统支持日志集中管理，便于故障排查。

系统采用QPS（每秒查询率）作为性能指标，系统支持高并发访问。系统具备良好的兼容性，能够支持不同版本的客户端接入。

1.2 技术选型

- 后端：Spring Boot + MyBatis-Plus + JWT + Spring Validation + Lombok
- 数据库：MySQL + Redis
- 前端：Vue 3 + Vite + Element Plus + Pinia + Vue Router + Axios

1.3 项目结构

项目采用模块化设计，主要分为前端、后端、数据库、缓存、中间件等模块。

2. 项目架构设计

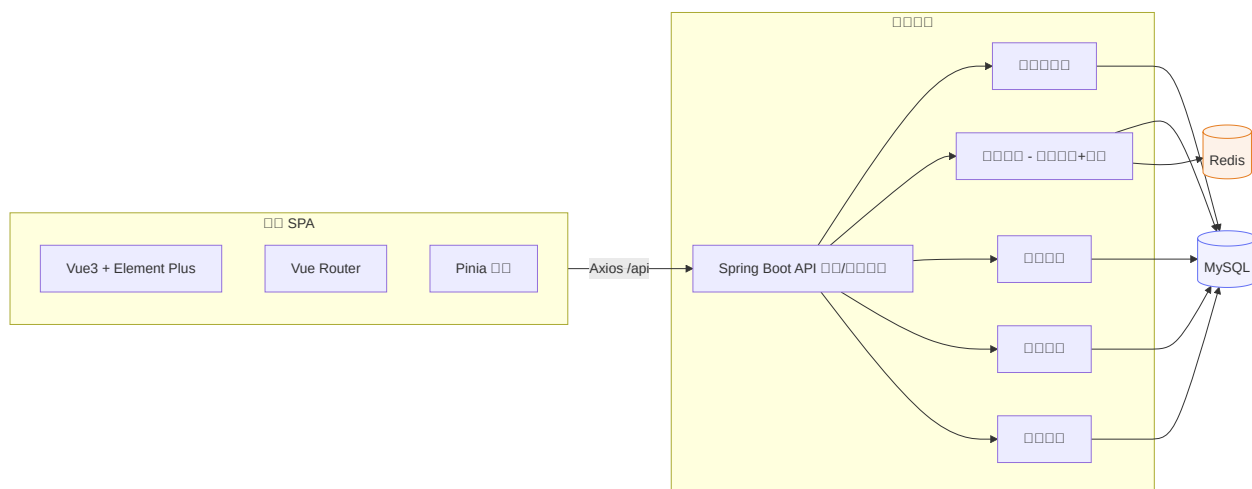
- 前端：Vue 3 + Vite + Element Plus + Pinia + Vue Router + Axios
- 后端：Spring Boot + MyBatis-Plus + JWT + Spring Validation + Lombok
- 数据库：MySQL + Redis
- 中间件：Spring Cloud Alibaba + Nginx + Docker + Kubernetes

- [illegible]

- [illegible]

- JWT 토큰은 Header, Payload, Signature로 구성된다
- Header는 토큰의 메타데이터를 담고 있다 (예: 토큰 타입, 버전)
- Payload는 클라이언트 정보를 담고 있다 (예: 사용자 ID, 권한)
- Signature는 토큰의 무결성을 검증하기 위해 생성된다

- 토큰화(Tokenization) : 텍스트를 단어나 구문으로 나눈다
- 임베딩(Embedding) : 토큰을 고차원 벡터로 변환한다
- 모델(Model) : 입력된 데이터를 처리하고 결과를 출력하는 알고리즘이다
- 평가(Evaluation) : 모델의 성능을 측정하는 방법이다



flowchart LR

```
subgraph Client[客户端 SPA]
    UI[Vue3 + Element Plus]
    Router[Vue Router]
    Store[Pinia 客户端]
end

subgraph Backend[服务端]
    APIGW[Spring Boot API 网关/服务端]
    Auth[认证]
    Rec[记录 - 记录+记录]
    Order[订单]
    Property[属性]
    UserSvc[用户服务]
end
```

```
DB[(MySQL)]:::db
Cache[(Redis)]:::cache
```

```
Client -->|Axios /api| APIGW
APIGW --> Auth
APIGW --> Rec
APIGW --> Order
APIGW --> Property
APIGW --> UserSvc
Auth --> DB
Rec --> DB
Order --> DB
Property --> DB
UserSvc --> DB
Rec --> Cache
classDef db fill:#f2f2ff,stroke:#6370f4;
classDef cache fill:#fdf2e9,stroke:#e67e22;
```

4.2 数据库

flowchart TB

```
View[前端<br/>Vue3 + Element Plus] --> BFF[后端<br/>Axios + 前端]
BFF --> Ctrl[控制层 Controller]
Ctrl --> Service[业务层 Service]
Service --> Mapper[数据层 MyBatis-Plus]
Mapper --> DB[(MySQL)]
Service --> RecCore[记录核心<br/>记录/记录]
RecCore --> Cache[(Redis/记录)]
```

```
subgraph Infra[ ]
  Security[JWT + Spring Security]
  Validation[ ]
  Logging[ ]
end
Ctrl --> Infra
```

5.

-
-
-
-
-
-

“/” frontend

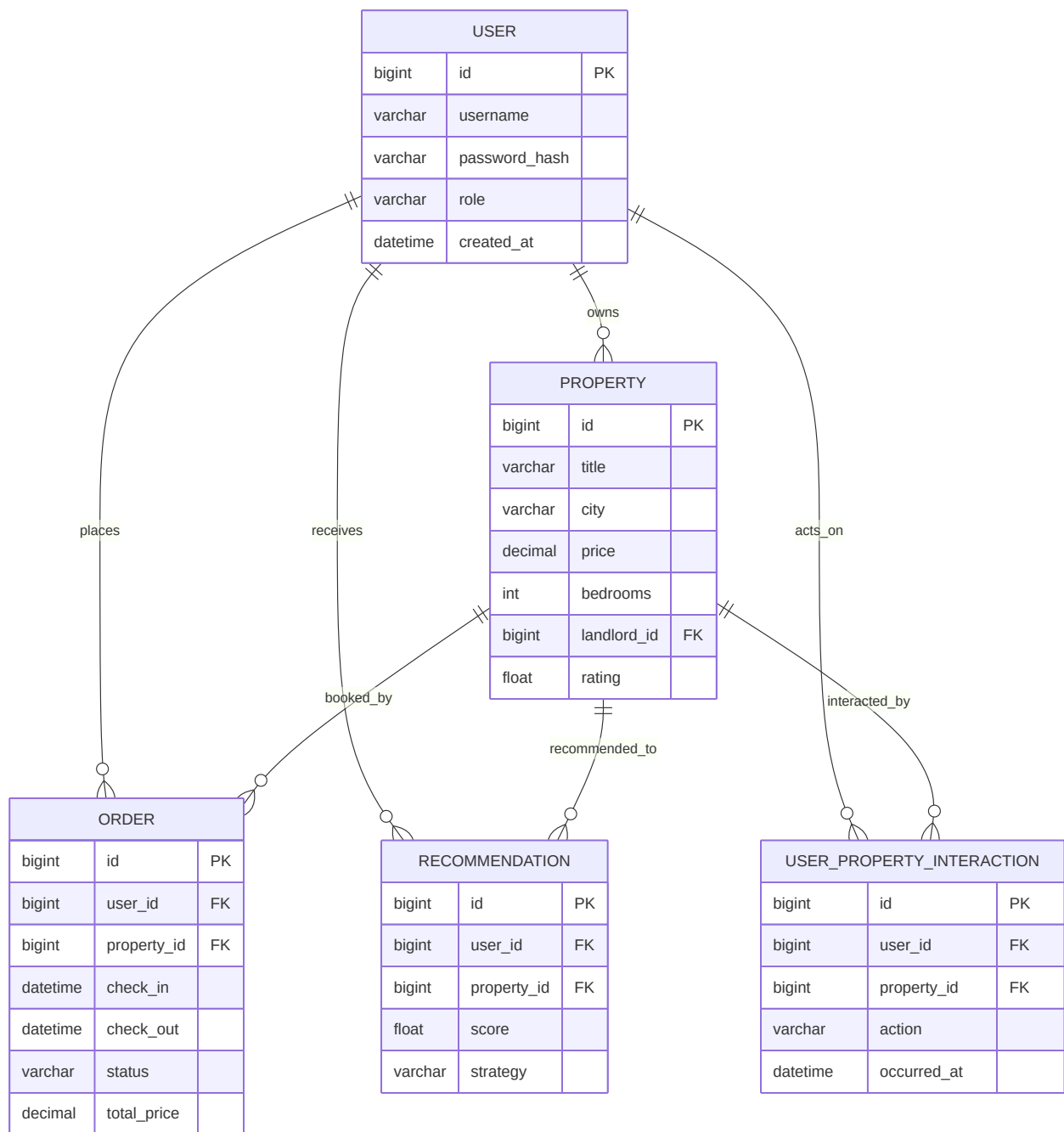
5.1

- **USER**
- **LANDLORD**
- **ADMIN**

5.2

- -**Top-N**
 -
 - **0.6** + **0.4** **A/B**
 - “**XX/YY**”
-

6. **ER**



```
erDiagram
```

```

USER {
    bigint id PK
    varchar username
    varchar password_hash
    varchar role
    datetime created_at
}

PROPERTY {
    bigint id PK
    varchar title
    varchar city
    decimal price

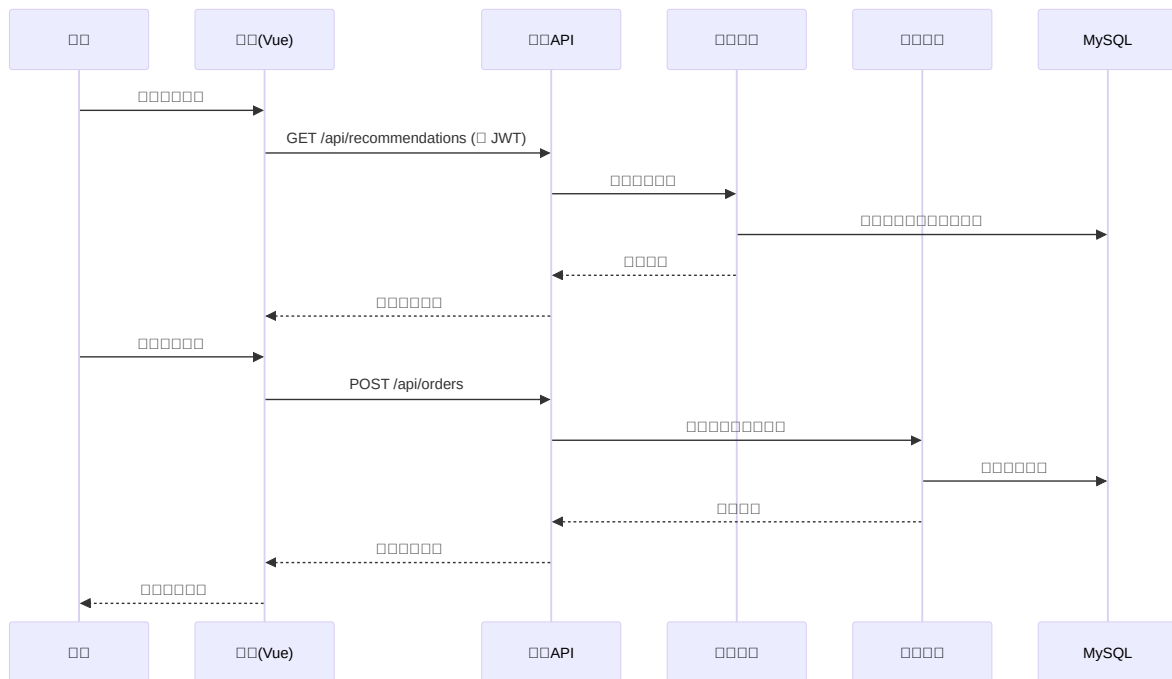
```

```

        int bedrooms
        bigint landlord_id FK
        float rating
    }
ORDER {
    bigint id PK
    bigint user_id FK
    bigint property_id FK
    datetime check_in
    datetime check_out
    varchar status
    decimal total_price
}
RECOMMENDATION {
    bigint id PK
    bigint user_id FK
    bigint property_id FK
    float score
    varchar strategy
}
USER_PROPERTY_INTERACTION {
    bigint id PK
    bigint user_id FK
    bigint property_id FK
    varchar action
    datetime occurred_at
}
USER ||--o{ ORDER : places
USER ||--o{ RECOMMENDATION : receives
USER ||--o{ USER_PROPERTY_INTERACTION : acts_on
PROPERTY ||--o{ ORDER : booked_by
PROPERTY ||--o{ RECOMMENDATION : recommended_to
PROPERTY ||--o{ USER_PROPERTY_INTERACTION : interacted_by
USER ||--o{ PROPERTY : owns

```

7. □□□□□□□□□□



□"□□□□□□□□□□"□□□□

sequenceDiagram

participant U as □□
 participant FE as □□(Vue)
 participant API as □□API
 participant REC as □□□□
 participant ORD as □□□□
 participant DB as MySQL

U->>FE: □□□□□□
 FE->>API: GET /api/recommendations (□ JWT)
 API->>REC: □□□□□□
 REC->>DB: □□□□□□□□□□
 REC-->>API: □□□□
 API-->>FE: □□□□□□
 U->>FE: □□□□□□
 FE->>API: POST /api/orders
 API->>ORD: □□□□□□□□
 ORD->>DB: □□□□□□
 ORD-->>API: □□□□
 API-->>FE: □□□□□□
 FE-->>U: □□□□□□

8. □□□□□□□□

- □□□cd frontend && npm install && npm run build□□□□□ frontend/dist□

- 使用 Postman/Newman 或 Rest Client 进行测试
- 使用 JMeter/Locust 进行测试 95/99 通过率
- 使用其他工具进行测试

10. 测试

测试用例设计

1. 测试用例设计
2. 测试用例设计
3. 测试 A/B 测试
4. 测试用例设计

11. 参考文献

- [1] Resnick P, Varian H R. Recommender systems. Communications of the ACM, 1997.
- [2] He X, et al. Neural Collaborative Filtering. WWW, 2017.
- [3] Sarwar B, et al. Item-based Collaborative Filtering Recommendation Algorithms. WWW, 2001.
- [4] 张三. 测试. 测试用例设计, 2016.
- [5] Kraska T. ML-based DBMS Design. SIGMOD, 2018.

12. 附录

附录 A: 测试用例设计

附录 B: 测试用例设计