

A horizontal row of twelve empty square boxes, intended for children to draw or color in.

□□□XXX□□□□□□□□

□□□XXX□□ □□□XXX □□□XXXXXXXX

□□□□□XXX □□□XXX

□□□□□2026 □ X □ X □

2

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

Spring Boot + Vue 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

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

1. 前言

1.1 项目简介

本项目是一个基于Spring Boot + Vue 3的后台管理系统，主要功能包括用户管理、权限管理、菜单管理等。项目采用前后端分离架构，前后端通过API进行交互。

本项目使用了流行的框架和技术栈，如Spring Boot、MySQL、Redis、JWT、Axios等，确保系统的稳定性和易用性。

项目“前后端分离”设计使得开发和维护更加方便，同时通过前后端的分离，提高了系统的可扩展性和可维护性。

1.2 项目结构

- 前端目录（Vue 3 + Element Plus）
- 后端目录（Spring Boot）
- 配置文件目录

1.3 项目特点

本项目具有以下特点：

2. 技术栈

- 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

3. 安全性

- 使用JWT进行身份验证，保证数据传输的安全性
- 实现了完善的权限控制，确保只有授权用户才能访问相应的资源
- 使用HTTPS协议，保证数据在传输过程中的安全性
- 实现了输入校验，防止SQL注入等安全问题

3.1 登录模块

- 登录/注册功能
- 密码加密存储
- 两步验证功能

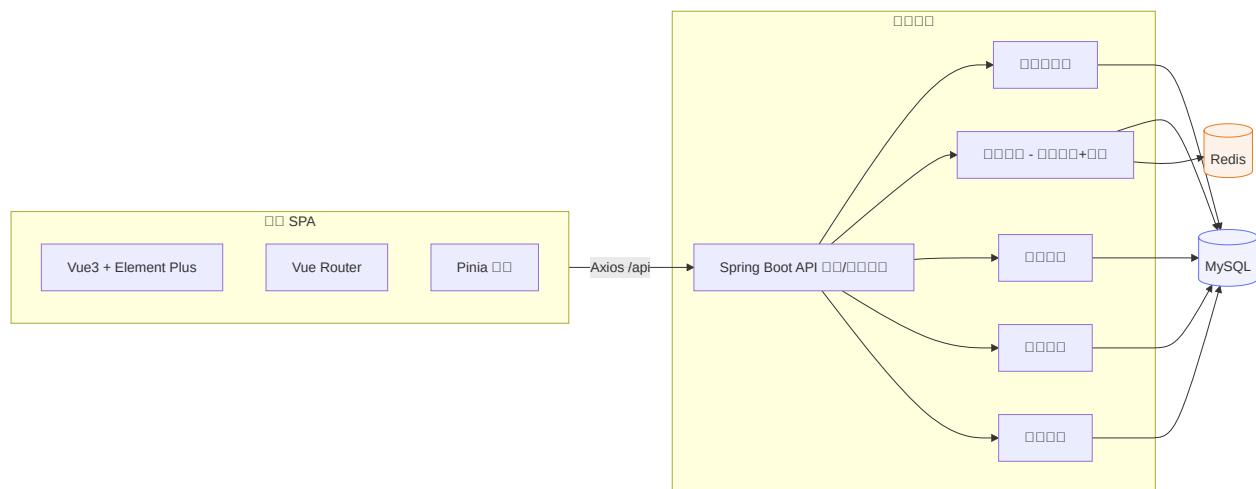
3.2 □□□□□

- JWT 令牌
 - 通过密钥解密
 - 通过密钥解密
 - 通过密钥解密

3.3 □□□□□

- `http://www.123.com/api/v1/auth/login`
 - `http://www.123.com/api/v1/auth/logout`
 - `http://www.123.com/api/v1/auth/renewToken` token
 - `http://www.123.com/api/v1/auth/refreshToken` HTTPS Token

4.



4.1 □□□□□

```
graph TD; subgraph Client [Client]; UI[UI]; Router[Router]; Store[Store]; end; subgraph Backend [Backend]; APIGW[APIGW]; Auth[Auth]; Rec[Rec]; Order[Order]; Property[Property]; UserSvc[UserSvc]; end;
```

```

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 网络架构

```

flowchart TB
    View[  
Vue3 + Element Plus] --> BFF[Axios + BFF]
    BFF --> Ctrl[Controller]
    Ctrl --> Service[Service]
    Service --> Mapper[MyBatis-Plus]
    Mapper --> DB[(MySQL)]
    Service --> RecCore[RecCore]
    RecCore --> Cache[(Redis)]
    subgraph Infra[Infra]
        Security[JWT + Spring Security]
        Validation[Validation]
        Logging[Logging]
    end
    Ctrl --> Infra

```

5. 安全性

- 使用JWT进行身份验证，支持USER/LANDLORD/ADMIN角色
- 实现了对敏感数据的加密存储和传输
- 支持多因素认证（MFA）
- 遵循PCI DSS标准处理支付信息

- 60% **LANDLORD** + 40% **USER**
 - 60% **LANDLORD** + 40% **USER**
- “**LANDLORD**/USER” **frontend** **LANDLORD** **USER**

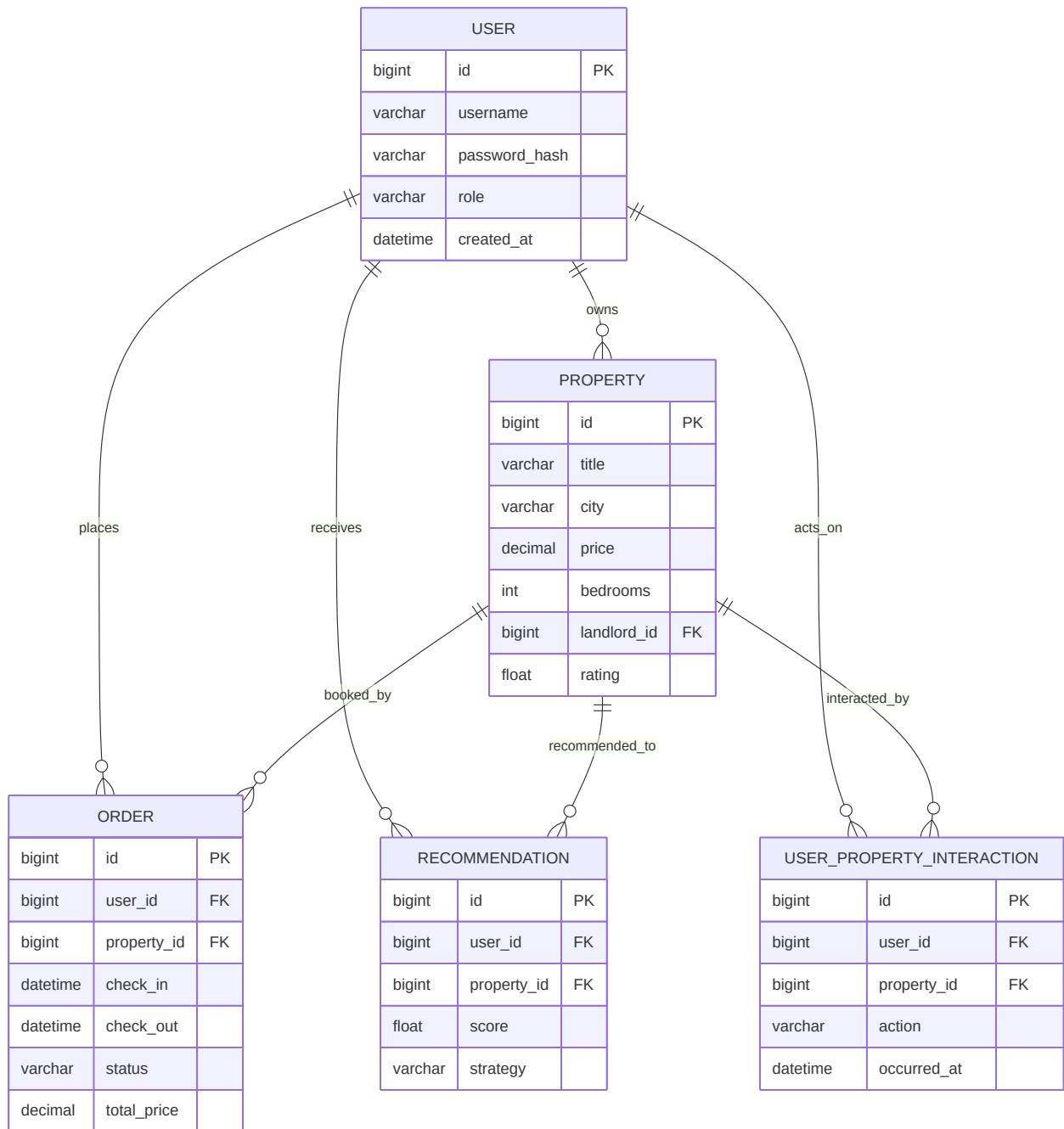
5.1 **LANDLORD**

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

5.2 **USER**

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

6. **ER**



erDiagram

```

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

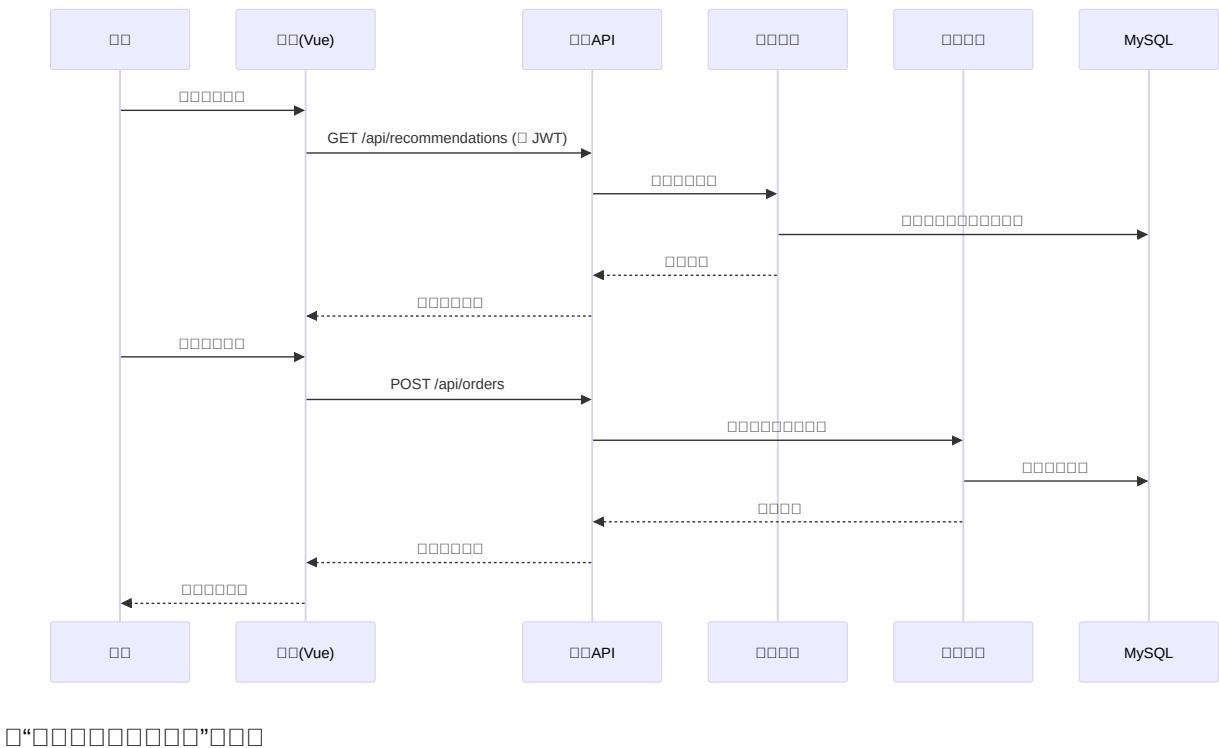
PROPERTY {
    bigint id PK
    varchar title
    varchar city
    decimal price
}
  
```

```

        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. □□□□□□□□□□



“MySQLへデータを送る”

```

sequenceDiagram
    participant U as ユーザー
    participant FE as フロントエンド(Vue)
    participant API as リンクAPI
    participant REC as レコメンド
    participant ORD as オーダー
    participant DB as MySQL

    U->>FE: ログイン
    FE->>API: GET /api/recommendations (JWT)
    API->>REC: レコメンドデータ
    REC->>DB: データベースへ
    REC-->>API: データベースから
    API-->>FE: 新規オーダー
    FE->>API: POST /api/orders
    API->>ORD: データベースへ
    ORD->>DB: データベースへ
    ORD-->>API: データベースから
    API-->>FE: オーダー確認
    FE-->>U: ログイン成功
  
```

8. フロントエンド

- cd frontend && npm install && npm run build

- mvn spring-boot:run
 - MySQL 8.x
 - application.yml

8.1

- **JWT + Axios Token 401**
 - **JWT + Axios Token 401**
 - **JWT + Axios Token 401**
 - **JWT + Axios Token 401**

8.2 □□□□□□

- `user_id` `property_id` `city` `created_at`
 - `ID` `UUID`
 - `/` `—`

8.3 □□□□□□

- **ELK**/MongoDB/Prometheus+Grafana/QPS/DB 监控
 - MySQL + Redis/CDN/CDN
 - HTTPS+JWT 安全性

9.

-  / /
 -  Postman/Rest Client API
 -  JMeter

9.1

- `http://www.example.com/api/v1/tokens` Token `XXXXXXXXXX`
 - `http://www.example.com/api/v1/tokens` / `http://www.example.com/api/v1/tokens`
 - `http://www.example.com/api/v1/tokens` / `http://www.example.com/api/v1/tokens` / `http://www.example.com/api/v1/tokens`
 - `http://www.example.com/api/v1/tokens` / `http://www.example.com/api/v1/tokens` / `http://www.example.com/api/v1/tokens`
 - `http://www.example.com/api/v1/tokens` Token `XXXXXXXXXX` 401 `XXXXXXXXXX` 403 `XXXXXXXXXX`

9.2

- JUnit/MockMvc Vitest

- Postman/Newman □ Rest Client
 - JMeter/Locust □ 95/99
 -

10.

1. □□□□□□□□□□□□□□□□□□
 2. □□□□□□□□□□□□□□□□□□□□□□
 3. □□ A/B □□□□□□□□□□□□
 4. □□□□□□□□□□□□□□□□□□□□

11.

- [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] ○○○. ○○○○. ○○○○○○○, 2016.
 - [5] Kraska T. ML-based DBMS Design. SIGMOD, 2018.

12. □□