Software Design Document (SDD)

Project Name: PropMan

Prepared By: [Your Name]

Date: [Insert Date]

# 1. Introduction

## 1.1 Purpose

This Software Design Document (SDD) describes the architecture and design of PropMan, a SaaS-based platform for property management.

## 1.2 Scope

PropMan allows property managers to manage properties, tenants, units, staff, and maintenance complaints.

# 2. System Architecture

## 2.1 Overview

Frontend: React.js  
Backend: Python (FastAPI)  
Database: MySQL  
Authentication: JWT + OTP  
Deployment: Cloud-based

## 2.2 High-Level Diagram

[ React Frontend ]  
 |  
[ FastAPI Backend ]  
 |  
[ MySQL Database ]  
 |  
[ Email Service ]

# 3. Design Considerations

## 3.1 Security

- OTP authentication  
- JWT tokens  
- Role-based access  
- Password encryption

## 3.2 Performance

- Optimized APIs  
- Pagination  
- Caching

# 4. Module Design

## 4.1 Authentication & Registration

### 4.1.1 Property Manager Registration

- Fields: First Name, Last Name, Email, Phone, Password, Confirm Password, Property Address  
- Flow:  
 Submit registration form → Send OTP → Enter OTP → Verify → Show dashboard  
- API:  
 POST /register/manager  
 POST /verify-otp

### 4.1.2 Login

- Email & Password → OTP verification  
- API:  
 POST /login  
 POST /verify-login-otp

### 4.1.3 Forgot Password

- Flow: Enter email → Receive OTP → Verify OTP → Set new password  
- API:  
 POST /forgot-password  
 POST /verify-reset-otp  
 POST /reset-password

## 4.2 Property Manager Dashboard

### 4.2.1 Overview Cards

- Total Properties  
- Total Tenants  
- Units Availability  
- Recent Open Complaints

### 4.2.2 API Endpoints

- GET /dashboard/summary  
- GET /dashboard/recent-complaints

## 4.3 Property Management

### 4.3.1 Features

- Add/Edit/Delete properties  
- Fields: Title, Address, Unit Count, Available Units, Age, Distance to facilities

### 4.3.2 API Endpoints

- POST /properties  
- GET /properties  
- PUT /properties/{id}  
- DELETE /properties/{id}

## 4.4 Unit Management

### 4.4.1 Features

- Manage units per property  
- Fields: Unit Number, Block, Area, Maintenance Charges, Unit Type, Furnishing

### 4.4.2 API Endpoints

- POST /properties/{id}/units  
- GET /properties/{id}/units  
- PUT /units/{id}  
- DELETE /units/{id}

## 4.5 Tenant Management

### 4.5.1 Tenant Onboarding

- Manager adds tenant → Email sent → OTP verification → Dashboard access

### 4.5.2 Allocation & Details

- Allocate unit(s) to tenant  
- Add lease details (e.g., Rent, Agreement Dates, Deposit)

### 4.5.3 API Endpoints

- POST /tenants  
- GET /tenants  
- GET /tenants/{id}  
- PUT /tenants/{id}  
- DELETE /tenants/{id}  
- POST /tenants/{id}/allocate-unit

## 4.6 Tenant Dashboard

### 4.6.1 Features

- Show total complaints, total units  
- View personal info, recent property, emergency contacts

### 4.6.2 API Endpoints

- GET /tenant/dashboard  
- GET /tenant/profile

## 4.7 Complaints Management

### 4.7.1 Features

- Submit complaint (title, description, images, priority)  
- Manager assigns staff  
- Status updates & comments

### 4.7.2 API Endpoints

- POST /complaints  
- GET /complaints  
- GET /complaints/{id}  
- POST /complaints/{id}/assign  
- POST /complaints/{id}/status  
- POST /complaints/{id}/comments

## 4.8 Staff Management

### 4.8.1 Onboarding

- Add staff → Email + OTP → Access dashboard

### 4.8.2 Dashboard

- View assigned complaints, update status, add comments/photos

### 4.8.3 API Endpoints

- POST /staff  
- GET /staff  
- GET /staff/{id}/complaints  
- POST /staff/{id}/complete-complaint

## 4.9 Reports Module

### 4.9.1 Features

- HTML-based reports: tenants, units, complaints, staff

### 4.9.2 API Endpoints

- GET /reports/complaints  
- GET /reports/tenants  
- GET /reports/staff

# 5. Database Design

## 5.1 Tables Overview

|  |  |
| --- | --- |
| Table | Description |
| users | Stores all users (manager, tenant, staff) |
| properties | Property data |
| units | Units per property |
| tenants | Tenant personal info |
| staff | Staff details |
| complaints | Complaint records |
| complaint\_comments | Comments/photos for a complaint |
| otp\_verification | Temporary OTP data |

## 5.2 Sample Table Structures

users: id, first\_name, last\_name, email, phone, password\_hash, role

properties: id, title, address, unit\_count, available\_units, age\_years, distances...

complaints: id, tenant\_id, property\_id, unit\_id, title, description, priority, status

# 6. UI Design Guidelines

- React + Tailwind/Material UI  
- Responsive layout  
- Separate views for roles

# 7. External Integrations

- SMTP/SendGrid for email  
- S3/local for image upload  
- OTP via SendGrid/email service

# 8. Deployment Strategy

- Dockerized backend  
- React deployed via Vercel/Netlify  
- MySQL hosted  
- CI/CD via GitHub Actions

# 9. Future Enhancements

- Lease management  
- Chat interface  
- Push notifications  
- Mobile app