Community Property Management System Team

Design Review Doc

Team members:
Yuan Xue, Zhiqi Bei, Liang Liu, Zihan He, Daiming Yang, Ziyue Wang, Yuhan Zhang, Jiawen Wang, Boyan Tian, Chen Tang, Haolin Li, Lisha Xie

Table of Content

Goal

This document aims at addressing the challenge of difficult community property management. It provides the solution to this challenge at an abstract level, including the background, user research, as well as structure of a technical solution we see as optimal for facing the challenge and the thought process we went through to reach the solution.

Background

The property office of a community face management issues and is receiving complaints from residents. First, the third-party management company is slow and inefficient in response to issues raised by residents, such as repair requests and common room reservations. Second, for the office, there are issues of management fee payments overdue, as well as a lack of communication over community issues. They request developing a trilateral management platform to boost management efficiency.

Requirements

1. For management service providers, enable service request ticket review to standardize and speed-up management service process

2. For residents, automate the common room booking process to reduce booking clashes and ill tracking
3. For property service vendors, provide a platform for payment to reduce overdue service fees
4. For property service vendors, enable sending notifications and community happenings to better communicate with residents

Technical Problem Challenges

Challenge #1: Unsettled credential system design
Challenge #2: Function-based team assignment makes each function separate and different

Technical Proposal

Entity–relationship diagram

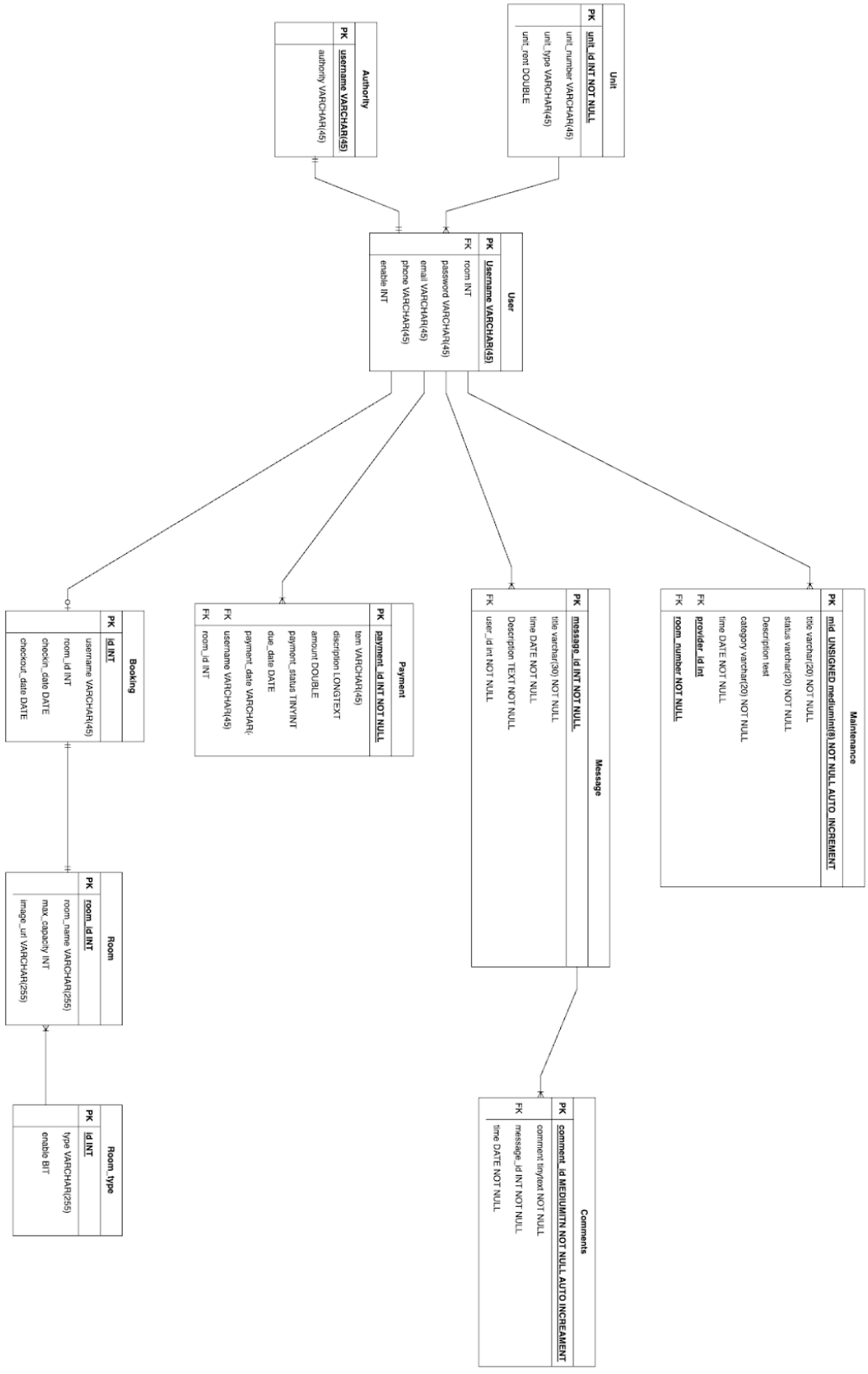
Click to view or see Appendix

Restful Endpoints

Click to view or see Appendix

Appendix

Appendix 1. Entity-Relationship Diagram



Appendix 2. Restful Endpoints

Login & Register

POST /login

INPUT:

```
{
        "username": "user1",
        "password": "****"
}
```

OUTPUT:

```
{
        "status": true
}
```

PUT /register

INPUT:

OUTPUT:

Service Request

POST / request

- Purpose : for tennent Create a new request

INPUT:
{
Header: Bearer Token

Body: {

“Title” : “...”,
“Category” : “...”,
“Description” : “...”,
}
}

OUTPUT:
{
Status : “200”
}

POST /request

- Purpose : for manager change request status

INPUT:
{
"Request ID":
"New Status":
}

OUTPUT:
{
"Status": 200
}

GET/request

- Purpose : get request list

INPUT:
{
Header: Bearer Token
}

OUTPUT:

ListOF
{
“Title”: “...”,

“Status” : “...” ,
“Category” : “...”,
“UploadTime” : “...”,
“roomNumber” : “...”,
“Description” : “...”
}

DELETE/ request

- Purpose: Delete Request

INPUT:
{
"Request ID":
}

OUTPUT:
{
"Status": 200
}

Message & announcement

Features priority:
P0, manager post announcement
P0, manager deleted announcement
P0, user post messages
P0, user filter/find his/her own message
P0, user delete his/her own message
P1, post comments, delete comments

POST /announcement

- Purpose : for managers to pose announcement to show on tenant’s homepage

INPUT:
{
“title”: ”xxx”,
“time”: Date,
“content”: “...”,
“manager_id”:”xxx”
}

OUTPUT:
{
"status": true
}
1. store to DB
2. notify the manager

GET /announcement:

- Purpose : to fetch announcement to show on manager and tenant's homepages

OUTPUT:
{
"title": "xxx",
"time": Date,
"content": "...",
"manager_id":"xxx"
}

DELETE /announcement:

- Purpose : for manager to delete the announcement

INPUT:
{
"announce_id:""
}

OUTPUT:
{
"status": true
}

POST /message

- Purpose : for tenant and manager to post a new thread

INPUT:
{
"title": "xxx",
"time": Date,

“content”: “...”,
“user”:”xxx”
}

OUTPUT:
{
"status": true
}
1. store to DB
2. notify this user

POST /comment

- Purpose : for tenant and manager to reply a thread

INPUT:
{
“title”: ”xxx”,
“time”: Date,
“content”: “...”,
“user”:”xxx”,
“message_id”: “xxx”
}

OUTPUT:
{
"status": true
}
1. store to DB
2. notify this user

GET /message:

- Purpose : to fetch all message and comments to show on manager and tenant’s homepages

OUTPUT:
{
“message_id”: ”...”
“title”: ”xxx”,
“time”: Date,
“content”: “...”
“comments”:
{
“comment_id” : “....”

“time”: Date,
“content”: “...”
}
}

DELETE /message:

- Purpose : for users to delete his/her own posted message and comments under this message

INPUT:
{
“message_id:””
}

OUTPUT:
{
"status": true
}

DELETE /comment:

- Purpose : for users to delete his/her own posted comment

INPUT:
{
“comment_id:””
}

OUTPUT:
{
"status": true
}

Booking

POST /reservation

- Purpose : creat a new reservation

INPUT:
{
*id (key): int
user_id:varchar(255)
room_id:varchar(255),

checkin_date: date,
checkout_date: date
enabled:bit
}

OUTPUT:
{
"status": true
}
1. store to DB
2. notify the manager and user

DELETE /reservation

- Purpose : delete reservation

INPUT:
{
*id (key): int
}

OUTPUT:
{
"status": true
}
1. store to DB
2. notify the manager and user

GET/ application

- Purpose : to fetch all applictions

INPUT:
{
user_id:varchar(255)

}

OUTPUT:
{
*id (key): int
user_id:varchar(255)
room_id:varchar(255),
checkin_date: date,
checkout_date: date
enabled:bit
}
1. store to DB
2. notify the manager and user

UPDATE /reservation
INPUT:
{
*id (key): int
enabled:bit
}

OUTPUT:
{
"status": true
}
1. store to DB
2. notify the manager and user

Payment

POST/pament:

- Purpose : make a new payment order

INPUT:
{
“manager_id”:

Body: {

term : “...”,
description : “...”,
roomNumber : “...”,
amount : “...”,
balance : “...”,
paymentStatus : “...”,
payDay: “...”,
dueDay: “...”
}
}

OUTPUT:
{
Status : “200”
}

GET/ payment

- Purpose : get list of payments

INPUT:

```
{
        UserID VARCHAR(255);
}
```

OUTPUT:
ListOF

```
{
        term VARCHAR(255);
        description VARCHAR(255);
        roomNumber INT;
        amount INT;
        balance INT;
        paymentStatus BIT(1);
        payDay Date;
        dueDay Date;


}
```

1. store to DB
2. notify the manager and user

POST/payment

- Purpose : for manager to Change Payment Status

INPUT:

```
{
        "UserID":
        "manager_id":
}
```

OUTPUT:

```
{
        "Status": 200
}
```

DELETE Delete Payment

- Purpose : for manager to delete Payment

INPUT:

```
{
```

```
        "Payment ID":
        “manager_id”:
}

OUTPUT:
{
        "Status": 200
}
```