Community Compliance Control System (CCCS) Web Application Re-Design

By Alicia Gay
July 17, 2022

Database Technology Overview

Community Compliance Control System will use the relational database PostgreSQL. I originally planned on using the document data store mongoDB, however when I began building my structure, I found that I had more relationships and decided to switch directions. PostgreSQL's support of various data types, transactional workflows, multiple fail-safe and access control system makes it a better choice for my application.

Since I intend to use AWS hosting I will use Amazon RDS for PostgreSQL. The feature I most look forward to is the DB Event Notification which provides SNS notification for instance deployment, which I plan to tie into the verification required when someone submits a violation. This could also be useful in the pattern in which we send violation notifications to homeowners.

I plan to use AWS S3 as image storage to provide access to pictures from the violations report form.

Table stored in Database

Commented [AG1]: Removed all plural table names

1. Address

This table will have all addresses of the homes in the community. This table will be linked to the homeowners and report_viol tables, specifically the address.

```
CREATE TABLE address (
        addr_id         int4            PRIMARY KEY    DEFAULT,
        address         varchar (100)   NOT NULL,
        city            varchar (25)    NOT NULL,
        state           varchar (2)     NOT NULL,
        zip             varchar (5)     NOT NULL
);
```

2. Account

This table will hold the user accounts for use in the system. The email in this table will be linked to the homeowners table. The user_id will be linked to the roles table to identify how the user can use the app.

```
CREATE TABLE account (
        account_id              int4            PRIMARY KEY,
        username                varchar (50)    UNIQUE          NOT NULL,
        password                varchar (50)    NOT NULL,
        email                   varchar (255)   UNIQUE          NOT NULL,
        created_on              date            NOT NULL,
        last_logIn              date
);
```

Commented [AG2]: Changed from user_id to account_id

3. Role

This table will have the user account roles for use in the system. A role_id will hold an automatically generated id that will be linked to a role_name with unique values that establish level of use for a user.

```
CREATE TABLE role (
        role_id                 int4            PRIMARY KEY,
        role_name               varchar (250)   UNIQUE          NOT NULL,


);
```

4. Account Role

This table will establish user account with roles for use in the system and how that may change as a homeowner joins the board or leaves the board. This will be used to bring together the accounts table and roles table using the id information.

```
CREATE TABLE account_role (
        account_id              int4            NOT NULL,
        role_id                 int4            NOT NULL,
        grant_date              date,
        PRIMARY KEY (user_id, role_id),
        FOREIGN KEY (role_id)
                REFERENCES roles (role_id),
        FOREIGN KEY (user_id)
                REFERENCES accounts (user_id)
);
```

Commented [AG3]: Changed to account_id

5. Homeowner

This table will have the homeowner's contact information associated with the addresses of the homes in the community. The addr_id will link to the address table and the email will link to the account table.

```
CREATE TABLE homeowner (
        homeowner_id            int4            PRIMARY KEY,
        addr_id                 int4            FK,
        first_name              varchar (50)    NOT NULL,
        last_name               varchar (50)    NOT NULL,
        address                 varchar (100)   NOT NULL FK,
        phone                   varchar (12),
        email                   varchar (50)    UNIQUE NOT NULL  FK,
);
```

6. Violation

This table will have a list of violations using titles and descriptions to be used in other tables.

```
CREATE TABLE report_viol  (
        vi_id           integer         PRIMARY KEY,
        title           varchar (50)    NOT NULL,
        description     varchar (200)   NOT NULL,
);
```

7. Violation Report

This table will have a list of violation that are attached to each address along with warning letter dates, fines, etc. Images will be stored in AWS S3 and linked to the table. The address will

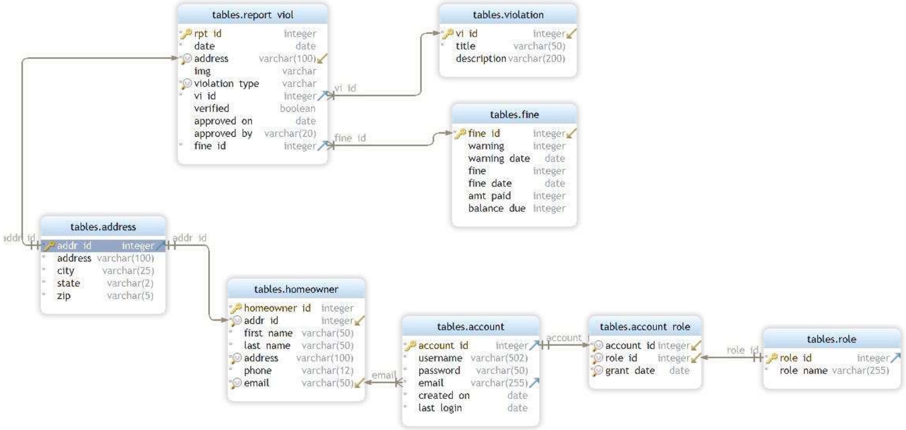
```
CREATE TABLE report_viol  (
          rpt_id            integer           PRIMARY KEY,
          date              date              NOT NULL,
          address           varchar (100)     NOT NULL        FK,
          img               varchar,
          violation         varchar (50)      NOT NULL        FK,
          verified          Boolean,
          approved_on       date,
          approved_by       varchar(20)
);
```

8. Fine

This table will have a list of fines that occur with each violation reported.

```
CREATE TABLE fine  (
          fine_id           integer           PRIMARY KEY,
          fine              integer,
          warning_date      date,
          fine_date         date,
          amt_paid          integer,
          balance_due       integer
);
```

Tables Diagram



Community Compliance Control System (CCCS) Web Application Re-Design

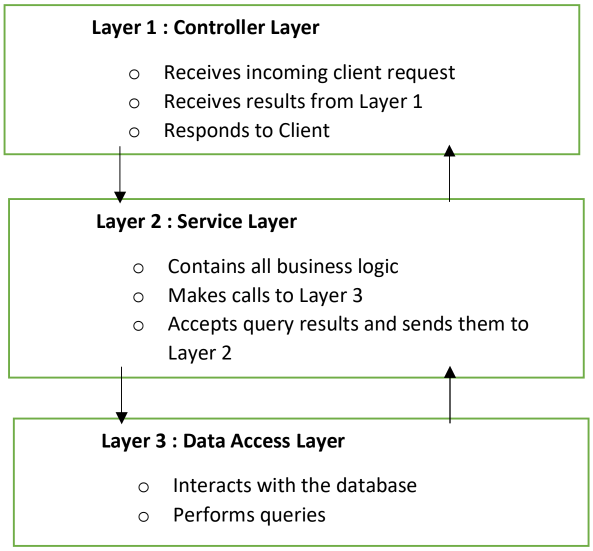
By Alicia Gay
July 17, 2022

2. Design Service Layers

Overview

I will build my web application using the backend service Express for simple routing. A REST API will be called by the front end for service.

Backend will have a 3-layer (Service-Oriented) Architecture consisting of the following:



Layer 1 : Controller Layer

- Receives incoming client request
- Receives results from Layer 1
- Responds to Client

Layer 2 : Service Layer

- Contains all business logic
- Makes calls to Layer 3
- Accepts query results and sends them to Layer 2

Layer 3 : Data Access Layer

- Interacts with the database
- Performs queries