Web-Based Information System Design

Gram Panchayat Management System

Team

```
postgres=# SELECT * FROM PRQL ORDER BY rollnum;
  rollnum   |         name
------------+----------------------
 22CS10030  | Harshit Jain
 22CS10049  | Nived Roshan Shah
 22CS30041  | Parth Shashin Patil
 22CS30045  | Rishit Garg
 22CS30046  | Sachish Singla
(5 rows)
```

ER Diagram

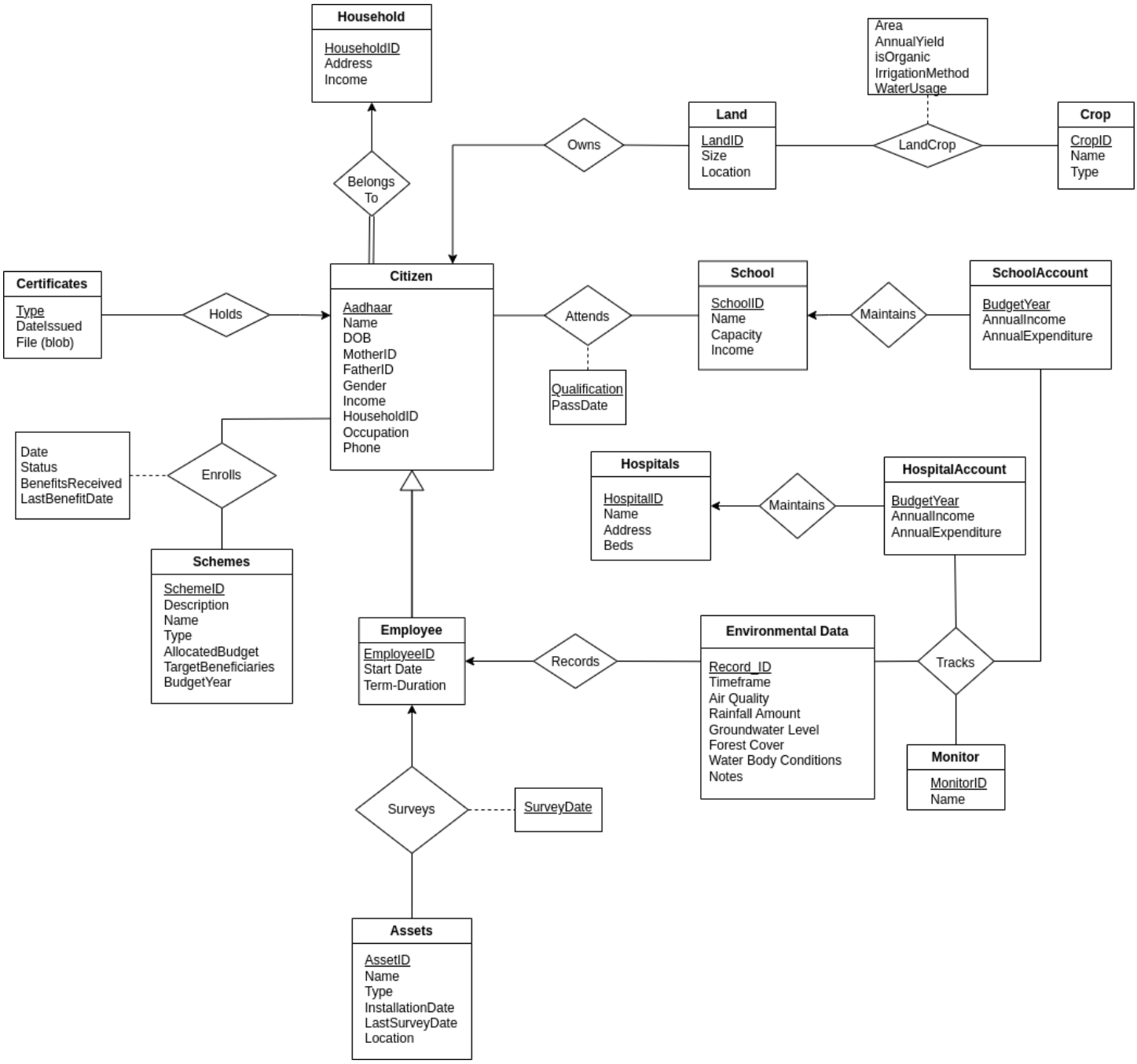
Household
HouseholdID
Address
Income
Area
AnnualYield
isOrganic
IrrigationMethod
WaterUsage
Land
LandID
Size
Location
Crop
CropID
Name
Type
Owns
LandCrop
Belongs To
Certificates
Type
DateIssued
File (blob)
Holds
Citizen
Aadhaar
Name
DOB
MotherID
FatherID
Gender
Income
HouseholdID
Occupation
Phone
Attends
School
SchoolID
Name
Capacity
Income
Maintains
SchoolAccount
BudgetYear
AnnualIncome
AnnualExpenditure
Qualification
PassDate
Date
Status
BenefitsReceived
LastBenefitDate
Enrolls
Hospitals
HospitalID
Name
Address
Beds
Maintains
HospitalAccount
BudgetYear
AnnualIncome
AnnualExpenditure
Schemes
SchemeID
Description
Name
Type
AllocatedBudget
TargetBeneficiaries
BudgetYear
Employee
EmployeeID
Start Date
Term-Duration
Records
Environmental Data
Record_ID
Timeframe
Air Quality
Rainfall Amount
Groundwater Level
Forest Cover
Water Body Conditions
Notes
Tracks
Monitor
MonitorID
Name
Surveys
SurveyDate
Assets
AssetID
Name
Type
InstallationDate
LastSurveyDate
Location

Table Schema

Citizen and Household Management

```
CREATE TABLE Citizen (
    Aadhaar VARCHAR(16) PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    DOB DATE,
    MotherID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    FatherID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    Gender VARCHAR(10),
    Income DECIMAL(12, 2),
    HouseholdID INTEGER REFERENCES Households(HouseholdID),
    Occupation VARCHAR(100),
    MigrationStatus VARCHAR(50) CHECK (MigrationStatus IN ('Native',
'Immigrant')),
    ResidenceSince DATE,
    Phone VARCHAR(20)
);
```

```
CREATE TABLE Households (
    HouseholdID SERIAL PRIMARY KEY,
    Address TEXT NOT NULL,
    Income DECIMAL(12, 2)
);
```

Employee and Monitor Management

```
CREATE TABLE EmployeeCitizens (
    EmployeeID SERIAL PRIMARY KEY,
    CitizenID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    StartDate DATE NOT NULL,
    TermDuration INTEGER, -- in months
    Role VARCHAR(50)
);
```

```
CREATE TABLE Monitors (
    MonitorID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL
);
```

Authentication System

```
CREATE TABLE users (
    UserID SERIAL PRIMARY KEY,
    CitizenID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    MonitorID INTEGER REFERENCES Monitors(MonitorID),
    username VARCHAR(64) UNIQUE NOT NULL,
    password VARCHAR(256) NOT NULL,
    auth VARCHAR(20) CHECK (auth IN ('citizen', 'employee', 'monitor')),
    salt VARCHAR(256) NOT NULL,
    CONSTRAINT one_id_not_null CHECK (
        (CitizenID IS NULL AND MonitorID IS NOT NULL) OR
        (CitizenID IS NOT NULL AND MonitorID IS NULL)
    )
);
```

Education Management

```sql
CREATE TABLE Schools (
    SchoolID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Capacity INTEGER,
    Income INTEGER
);
```

```sql
CREATE TABLE AttendsSchool (
    CitizenID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    SchoolID INTEGER REFERENCES Schools(SchoolID),
    Qualification VARCHAR(100),
    PassDate DATE,
    PRIMARY KEY (CitizenID, SchoolID, Qualification)
);
```

```sql
CREATE TABLE SchoolAccount (
    SchoolID INTEGER REFERENCES Schools(SchoolID),
    AnnualIncome DECIMAL(15, 2),
    AnnualExpenditure DECIMAL(15, 2),
    BudgetYear INTEGER
);
```

Healthcare Management

```
CREATE TABLE Hospitals (
    HospitalID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Address TEXT,
    Beds INTEGER
);
```

```
CREATE TABLE HospitalAccount (
    HospitalID INTEGER REFERENCES Hospitals(HospitalID),
    AnnualIncome DECIMAL(15, 2),
    AnnualExpenditure DECIMAL(15, 2),
    BudgetYear INTEGER
);
```

Environmental Data

```
CREATE TABLE EnvironmentalData (
    RecordID SERIAL PRIMARY KEY,
    TimeFrame DATE NOT NULL,
    AirQuality DECIMAL(5, 2),
    RainfallAmount DECIMAL(8, 2),
    GroundwaterLevel DECIMAL(8, 2),
    ForestCover DECIMAL(8, 2),
    WaterBodyConditions TEXT,
    RecordedBy INTEGER REFERENCES EmployeeCitizens(EmployeeID),
    Notes TEXT
);
```

Agricultural Management

```sql
CREATE TABLE Land (
    LandID SERIAL PRIMARY KEY,
    OwnerID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    Size DECIMAL(10, 2),
    Location TEXT
);
```

```sql
CREATE TABLE Crop (
    CropID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Type VARCHAR(100)
);
```

```sql
CREATE TABLE LandCrop (
    LandID INTEGER REFERENCES Land(LandID),
    CropID INTEGER REFERENCES Crop(CropID),
    Area DECIMAL(10, 2), -- in acres
    AnnualYield DECIMAL(10, 2), -- in kg
    isOrganic BOOLEAN,
    IrrigationMethod VARCHAR(100),
    WaterUsage DECIMAL(10, 2), -- in liters per acre
    PRIMARY KEY (LandID, CropID)
);
```

Welfare Schemes & Certificates

```
CREATE TABLE Schemes (
    SchemeID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Description TEXT NOT NULL,
    Type VARCHAR(100),
    AllocatedBudget DECIMAL(15, 2),
    TargetBeneficiaries INTEGER,
    BudgetYear INTEGER
);
```

```
CREATE TABLE SchemeEnrollment (
    SchemeID INTEGER,
    CitizenID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    Date DATE NOT NULL,
    PRIMARY KEY (SchemeID, CitizenID),
    EnrollmentStatus VARCHAR(50) DEFAULT 'Active' CHECK (EnrollmentStatus IN
('Active', 'Inactive', 'Pending')),
    BenefitsReceived DECIMAL(15, 2) DEFAULT 0,
    LastBenefitDate DATE,
    FOREIGN KEY (SchemeID) REFERENCES Schemes(SchemeID) ON DELETE CASCADE ON
UPDATE CASCADE
);
```

```
CREATE TABLE Certificates (
    -- CertificateID SERIAL PRIMARY KEY,
    Category VARCHAR(100) NOT NULL,
    Name VARCHAR(100) NOT NULL,
    CitizenID VARCHAR(16) REFERENCES Citizen(Aadhaar),
    DateIssued DATE NOT NULL,
    File BYTEA, -- Binary data for the file
    PRIMARY KEY (Category, Name, CitizenID)
);
```

Asset Management

```
CREATE TABLE assets (
    asset_id SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Type VARCHAR(100) NOT NULL,
    InstallationDate DATE,
    LastSurveyedDate DATE,
    Location TEXT
);
```

```
CREATE TABLE AssetSurveys (
    asset_id INTEGER REFERENCES assets(asset_id),
    SurveyDate DATE,
    SurveyorID INTEGER REFERENCES EmployeeCitizens(EmployeeID),
    SurveyData TEXT,
    PRIMARY KEY (asset_id, SurveyDate)
);
```

Triggers

```
CREATE OR REPLACE FUNCTION update_household_income()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE Households
    SET Income = (SELECT COALESCE(SUM(Income), 0) FROM Citizen WHERE HouseholdID
= NEW.HouseholdID)
    WHERE HouseholdID = NEW.HouseholdID;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_update_household_income
AFTER INSERT OR UPDATE OR DELETE ON Citizen
FOR EACH ROW
EXECUTE FUNCTION update_household_income();
```

```
CREATE OR REPLACE FUNCTION update_last_surveyed()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE assets
    SET LastSurveyedDate = (
        SELECT COALESCE(MAX(SurveyDate), InstallationDate)
        FROM AssetSurveys
        WHERE AssetSurveys.asset_id = assets.asset_id
    );

    RETURN NULL;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_update_last_surveyed
AFTER INSERT OR UPDATE OR DELETE
ON AssetSurveys
FOR EACH STATEMENT
```

```
EXECUTE FUNCTION update_last_surveyed();
```

Functionalities Implemented

The project requires the development of an information system for a Gram Panchayat, designed with the perspective of four different kinds of users kept in mind. This approach strongly emphasizes applicability, relevance, usability, and accessibility to create a system that ensures effectiveness and scalability.

The system implements role-based access control with distinct functionalities for citizens, employees, and monitors:

Citizen Portal

Profile Management: Allows a citizen to view and edit their personal details

Certificate Access: Users can view their education and vaccination certificates in a PDF viewer interface. The system shows when a certain certificate for an entry is unavailable

Statistics Dashboard: Statistics are displayed under four categories, which can easily be extended to incorporate more data. It also uses powerful yet simple visualizations to describe trends and relevant numbers for higher understandability. Here is a high-level description of the information shown:

Education: Gender-wise literacy rates, schools in the panchayat, and number of enrolled students

Health: Vaccination rates, total count of beds in hospitals, and list of nearby hospitals with relevant details

Agriculture: Private land area, per-crop land farming land utilization, share of organic crops, annual yield

Demographic: Total population, Sex Ratio, number of households, average family size, population pyramid, income distribution, and occupation information

Scheme Enrollment: Citizens can view government welfare schemes and their application status and also filter by types of available schemes

Employee Directory: Contact information for panchayat officials holding different roles

Employee Portal

Citizen Management: Employees can view and update citizen information. Additionally, they can view the family information and household income for each citizen.

Certificate Management: The system allows Panchayat employees to upload certificates (education and vaccination)

Scheme Management: Add, edit, or remove government schemes. Employees can also track information about the enrollments of each scheme.

Asset Management: Employees can track community assets and conduct surveys. For each asset, information such as current conditions, complaints, and comments can be stored in the database.

Monitor Portal

Government Monitors can view everything that a citizen can, along with advanced analytics, as described below.

Education: Analysis of financials such as annual income and expenditure can be viewed for each school. Overall aggregate figures regarding this information are visualized using graphs and charts.

Health: An overview is presented, with key figures such as Doctor to Citizen and Nurse to Citizen ratios to provide an understanding of the health facilities available. Financials for hospitals are visible as well.

Agriculture: Detailed information on topics such as irrigation methods, per-crop water usage, land area utilization can be seen. Additional data can be added as applicable.

Environmental Data: Gives the information of average AQI, rainfall, groundwater level and forest cover for the current year. Also shows the yearly variation of AQI and rainfall as a line chart.

Scheme Analysis: Budget allocation tracking, enrollment statistics, and impact assessment are important information that can be viewed by government monitors.

Admin Portal

Database Management: The admin acts as a superuser and can perform direct SQL queries. The system provides a dynamic interface by fetching all table names and columns present in the database. The results of SELECT queries are also shown within the browser window.

User Management: The admin can view all users present in the system. They can add users to the system corresponding to a certain citizen or employee.

Shared Features

Authentication System: Ensures secure login by implementing hashed passwords with salts. Every session is validated before executing SQL queries, preventing unauthorized access.

Role-Based Access: Access control mechanisms restrict functionalities based on user roles. Citizens, employees, monitors, and admins each have separate permissions, ensuring data confidentiality and controlled modifications.

Parameterized Queries: The system strictly enforces the use of parameterized queries to prevent SQL injection attacks. All database interactions are handled securely by ensuring user input is never directly concatenated into SQL statements. This approach eliminates the risk of malicious query manipulation.

Frontend Tools

The application employs a modern web-based interface built with the following technologies:

Core Technologies

- **Flask**: Python web framework serving as the application backbone
- **Jinja2**: Templating engine for dynamic HTML generation
- **HTML5/CSS3**: Structure and styling of web pages
- **JavaScript**: Client-side interactivity and dynamic content
- **psycopg2**: PostgreSQL adapter for Python, providing database connectivity

UI Components and Libraries

- **Font Awesome**: Icon library for intuitive interface elements
- **Chart.js**: Data visualization for statistics and analytics dashboards
- **Bootstrap Components**: Form elements, tables, and responsive grid layouts
- **Custom CSS Framework**: Tailored styling system for consistent design

The frontend employs a consistent color scheme with green accents for the citizen portal, red for the admin interface, and a professional layout that prioritizes usability across different user roles.

Future Scope

Scheme Eligibility: While the database tracks enrollments for each scheme, it will be useful functionality to incorporate conditional eligibility of schemes based on citizen attributes.

Integration with Government Services: Currently, certificates and welfare schemes are uploaded and added by Panchayat Employees. Ideally, this data must be fetched and automatically synced with the Panchayat database.