**Greenviro SPAJ ISWMC Operation Data Programs**

This project is a web-based data collection application for the Greenviro SPAJ ISWMC (Integrated Solid Waste Management Center). It allows users to input operational data for various facilities within the center.

**FEATURES**

* **Data Entry Forms:** Separate data entry forms for different facilities: ◦ Security Guard Post (SGP) ◦ Material Recovery Facility (MRF) ◦ Leachate Treatment Plant (LTP)
* **Database Integration:** Data submitted through the forms is stored in a PostgreSQL database.
* **Responsive Design:** The application is designed to be user-friendly on both desktop and mobile devices, using the Bootstrap framework.
* **Placeholder Pages:** Includes placeholder pages for future development of the following modules:

◦ Weighbridge (WBS)

◦ Anaerobic Digester (ADS)

◦ Depot Repair Area (DRA)

* **Stored Data View:** A dedicated page to view the last seven days of operational data for SGP, MRF, and LTP in a tabular format.

**System and Program Architecture**

The application follows a client-server architecture:

• **Client-Side (Frontend):**

◦ Users interact with the application through web browsers.

◦ The user interface is built using **EJS (Embedded JavaScript)** templates for dynamic content rendering.

**Bootstrap** is used for responsive design and styling, ensuring a consistent look and feel across various devices.

Client-side JavaScript handles basic form interactions and date calculations (e.g., determining the day of the week from a selected date).

**Server-Side (Backend):**

◦ Powered by **Node.js** and the **Express.js** framework.

◦ Handles HTTP requests from the client (e.g., displaying pages, submitting form data).

◦ Processes incoming data, validates it, and interacts with the database.

◦ Renders EJS templates and sends them back to the client.

**Database:**

* **PostgreSQL** is used as the primary data store.
* The application connects to the PostgreSQL database to store and retrieve operational data for SGP, MRF, and LTP facilities.
* Database schema creation (tables for sgp, mrf, ltp) is handled programmatically on application startup if tables do not exist.

**Deployment:**

* The application is designed for deployment on cloud platforms like **Render**, which provides an environment for both the web service (Node.js/Express) and the PostgreSQL database.
* Environment variables are used to securely manage database connection strings and other configuration settings in production environments.

**TECHNOLOGY USED**

**Backend:** Node.js, Express.js, Moment.js

* **Database:** PostgreSQL
* **Frontend:** EJS (Embedded JavaScript), Bootstrap • **Version Control:** Git

**Deployment on Render**

This application is configured for easy deployment on [Render.](https://render.com/)

**Step-by-Step Instructions:**

1. **Push to GitHub:** Make sure your project code is pushed to a GitHub repository.

2. **Create a Render Account:** Sign up for a free account on Render using your GitHub profile.

3. **Create a New PostgreSQL Database:**

◦ In the Render Dashboard, click **"New +"** and select **"PostgreSQL"**.

◦ Give your database a unique name (e.g., iswmc-database).

◦ Select a region close to you.

* Click **"Create Database"**.
* Wait for the database to become available. Once it's ready, find the **"Connections"** section and copy the **"Internal Connection String"**. You will need this for the web service.

4. **Create a New Web Service:**

◦ In the Render Dashboard, click **"New +"** and select **"Web Service"**.

◦ Connect your GitHub account and select the repository for this project.

◦ Give your web service a unique name (e.g., iswmc-app).

◦ Under **"Environment"**, ensure the **"Runtime"** is set to **"Node"**.

◦ The **"Build Command"** should be npm install.

◦ The **"Start Command"** should be npm start.

5. **Add Environment Variables:**

◦ Under the **"Environment"** section for your new web service, click **"Add Environment Variable"**.

◦ Add the first variable:

**Key:** DATABASE\_URL

**Value:** Paste the **"Internal Connection String"** you copied from your PostgreSQL database.

◦ Add the second variable:

**Key:** NODE\_ENV

**Value:** production

6. **Deploy:**

* Click **"Create Web Service"** at the bottom of the page.
* Render will automatically build and deploy your application. You can monitor the progress in the deploy logs.
* Once deployed, you will be given a public URL (e.g., https-iswmc-app.onrender.com) where you can access your live application.

**Local Development Setup**

To run this application on your local machine, you will need to have Node.js and PostgreSQL installed.

|  |  |
| --- | --- |
| 1. | **Clone the repository:** |
|  | git clone <repository-url> cd iswmc\_monitor |

1. **Install dependencies:** npm install
2. **Set up Local PostgreSQL Database:**

* Create a new PostgreSQL database on your local machine (e.g., using psql or a GUI tool like pgAdmin).
* Obtain your database connection string. This string contains the necessary credentials (username, password, host, port, and database name) to connect to your PostgreSQL instance. It typically looks like this: postgresql://YOUR\_USERNAME:YOUR\_PASSWORD@YOUR\_HOST:YOUR\_PORT/YOUR\_DATABASE\_NAME **Important:** Replace YOUR\_USERNAME, YOUR\_PASSWORD, YOUR\_HOST, YOUR\_PORT, and YOUR\_DATABASE\_NAME with your actual database credentials.

4. **Create Environment File (.env):**

* Create a new file named .env /Users/rogerwoolie/Documents/Pers\_CScience\_2025/iswmc\_monitor/).
* Add your local database connection string to this file. For example:

DATABASE\_URL=postgresql://myuser:mypassword@localhost:5432/mydatabase

* **Security Note:** Never commit your .env file to version control (like Git) as it contains sensitive information. It's already included in .gitignore for this project.

5. **Install dotenv (if not already installed):**

* This package helps your Node.js application load environment variables from the .env file during local development. Run: npm install dotenv

6. **Configure index.js to load .env:**

* Ensure the following line is at the very top of your index.js file. This line tells your application to load the variables from .env: require('dotenv').config();

7. **Run the application:** npm start.

* The application will be available at http://localhost:3000.

**Database**

The application uses a PostgreSQL database. The schema is defined in index.js and the tables (sgp, mrf, ltp) are created automatically when the application starts if they do not already exist.

This program and application were developed by Azlan. Email: [coderazlan@gmail.com](mailto:coderazlan@gmail.com)