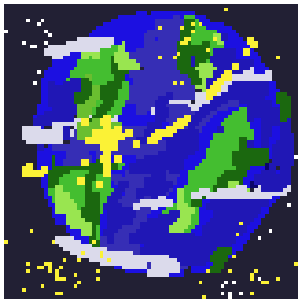
HelloWorldByMe

User Manual



April 30th, 2025

Project Sponsor:

Kevin Daily

Faculty Mentor(s):

Brian Donnelly, Savannah Chappus

Team Members:

Elizabeth Knight

Joey Banaszak

Jessica Maldonado Olivas

Samantha Madderom

Table of Contents

Introduction

We are pleased that you chose the WorldByMe Role-Based Access System for your business needs. The WorldByMe Role-Based Access System is a reliable platform meant to streamline communication between the Tucson government and local service providers, client navigation, and manage intraorganizational role-based access. Some of the key highlights include:

- Role-based access control driven by AWS RDS PostgreSQL
- Real-time communication via WebSockets
- Componential microservices architecture using Django and Node.js
- Intuitive client mapping and service referral interface

The purpose of this manual is to help you successfully install, configure, maintain, and troubleshoot issues with the WorldByMe Role-Based Access System in your context. We aim to ensure that you can rely on and benefit from this system for years to come.

Installation

For the final delivery, the WorldByMe Role-Based Access System has been deployed onto the server. However, as years go by, you might want to upgrade your infrastructure, conduct a server migration, or reinstall the product. This section will go over the steps needed to help you install the system on an Ubuntu-based server environment using Amazon EC2 and a PostgreSQL database hosted on AWS RDS.

Prerequisites

Before you begin with the system installation, make sure you have the following completed:

- ***An AWS EC2 instance running Ubuntu 24.04 LTS***

Background: Amazon EC2 (Amazon Elastic Compute Cloud) is an on-demand, scalable computing platform provided by AWS that allows customization when it comes to the scalability, security and networking, and storage. It reduces hardware costs so application development and deployment can happen rapidly.

More information: [What is Amazon EC2? - Amazon Elastic Compute Cloud]

Installation Process:

1. Create an AWS account
2. Open the AWS EC2 console at https://console.aws.amazon.com/ec2/
3. In the navigation bar at the top of the console, choose a region:
    a. **Ohio, Oregon, N. Virginia, N. California, etc.**
4. From the console dashboard, click 'Launch Instance'
5. Once redirected to the configuration page, provide the following information:
    a. A name for your instance,
    b. The preferred OS,
    c. Instance type,
    d. Key pair (click 'Create new key pair' and download it securely)
    e. Networking (use default VPC/subnet or configure as needed)
        i. The default VPC setting can implement the default subnet and security group that allows connections to your instance from anywhere. **If there needs to be specific security protocols in place, the settings can be easily edited when you select your instance, click on 'Security Groups', and click on the 'Edit' button.**
    f. Storage and advanced alert options
6. Once the instance summary is reviewed **carefully**, choose 'Launch Instance'
7. After launching the instance, verify that it is running and passes all status checks

More information: Get started with Amazon EC2 - Amazon Elastic Compute Cloud

- ***Access to an AWS RDS Instance with PostgreSQL***
    1. Log in to your AWS console
    2. Search for 'Aurora and RDS' in the search engine on the navigation bar
    3. Once the console has changed to the RDS dashboard, click **DB Instances > Create Database**
    4. Once redirected to the database configuration page, choose:
        a. engine type,
        b. template options (**production database or test/development database**),
        c. Instance specs and allow **EC2 connectivity**
    5. Configure networking/security group settings to allow EC2 access
    6. Review the database setup, **including the price range**, and click 'Create Database'
    7. After creating the database, ensure that the database instance is in the 'Available' state and take note of the endpoint

More information: Creating an Amazon RDS DB instance - Amazon Relational Database Service

- ***Tools Installed***
    - Node.js (v22+)
    - Python (v3.13+)
    - Git (most recent version)
    - Nginx (v1.24.0)
    - Docker and Docker Compose (most recent version)

    - This can be used for a **local** testing environment, which is **extremely** important when creating new features. It is not necessary to install this on the EC2 instance.

- ***Environment Variables***

  Prepare set values for:

  1. JWT secret (JSON web token)
     a. This might need to be generated separately
  2. Database endpoint (host)
  3. Database master username
  4. Database password
  5. Database name
  6. Database port

Next Steps

Below is a table summarizing all the commands you'll need to run:

Command	Directory	Purpose
pm2 list	backend	Check if the Node.js server is running
pm2 start	backend	Start Node.js backend logic
systemctl start nginx	frontend	Starts the NGINX server
npm run build	frontend	Compiles the application before use

1. Edit the .env file in the backend directory
    a. Using your prepared environment variables, edit the values like so:

2. Clone the repository in the home directory
    a. **Clone directly into the instance**
        i. ssh -i /path/to/key.pem *username*@[ec2-public-ip]
        ii. git clone ***https://github.com/HelloWorldByMe/HelloWorldByMe.git***
        iii. cd HelloWorldByMe-main
3. Connect the database instance to the EC2 instance
    a. Adjust AWS RDS settings to connect to the EC2 instance
        i. Skip this step if you did this already in the database setup section
        ii. If a connection is still not happening, the security settings is most likely the problem and adjustment of security groups is needed (**see troubleshooting section**)
    b. Make sure that it is fully connected by connecting to the database through the EC2 instance
        i. psql -h <rds-endpoint> -U <db-username>
4. SSH into the EC2 instance:
    a. Navigate to the HelloWorldByMe-main directory that was created earlier when you cloned the repository

b. To be able to start the system, you must have root privileges. Type in the following command to which the user to root: **sudo su root**

   i. Enter the password if prompted to switch to the root user

      1. Note: to confirm you are operating as a root user now, the header of the command line should start with **root@[ec2-public-ip]**

   ii. Navigate to backend directory: **cd backend**

   iii. Run **pm2 list** to see a list of processes currently running

   If you see something like this:

id	name	mode	□	status	cpu	memory
0	backend	fork	16	stopped	0%	0b

That means the backend process is not currently running. To start it, run: **pm2 start all**

Then run **pm2 list** again. You should now see something like this:

id	name	mode	□	status	cpu	memory
0	backend	fork	16	online	0%	67.5mb

If you don't see the **Nginx & PM2** part of our troubleshooting section!

   iv. Next, navigate to the frontend (run this command from the backend directory): **cd ../frontend**

   Run the following commands:

   systemctl start nginx

   npm run build

After running the last command, you should get an output telling you it was compiled successfully.

c. Then, there should be a message displayed saying that the application is successfully compiled

    i. If you see any errors, **navigate to the troubleshooting section**

d. If compiled successfully, type in the browser of your choice:

    i. **http://[ec2-public-ip]/login**

e. The login page should load and be displayed.

5. **If new features are to be added or any bugs are resolved, use Docker as a testing environment to make sure that any bad builds or mistakes are made are separate from the actual application**

    a. On your local machine, after cloning the repository:

        i. Navigate to the *frontend* folder: **cd frontend**

        ii. In terminal: **docker compose up --build**

Configuration and Daily Operation

Signup/Login

Our site automatically defaults to the login page. If you do not have an account, you can click “Signup” and you will be directed to a page where you can create an account. You’ll need to provide your full name, username, email address and password. Once that is done you’ll be automatically logged in. From the login page you will need to provide your username and password. Once logged in, you are able to log out from the sidebar using the very bottom button that says “Log Out”.

Profile

Once logged in, the user is greeted with their personal profile page. The user has the ability to add a blog status to have their own personal updates. There are also file and folder updates which are a part of future work for now but make their appearance on the profile website. The user can then see their organizations that they are active on as well as the date they joined and their role status.

Messaging

Our messaging system has three main message modes currently. They are Individual, Role, and Custom Group. For individual messages and group messages, you have the ability to search for users to add to the message by either username or role. When searching by username, suggested users are displayed as you type. Selecting the option to search by role will give you a drop down list from which you can select the role you wish to search by. Once you have done that, it will list everyone in the system with that role that you can select from. When creating a Custom Group message you can also provide a name for the group. If you neglect to do so, it will default to Unnamed Group. Finally, when sending a role-based message, it will present you will a drop down list from which you can select the role you wish to contact. Once you do, it will display everyone who has that role. This is for clarity so you know exactly who you are contacting.

You are able to create a new message at the top of the page, and your message threads will be displayed below. They all start as closed and you are able to open as many as you like to view the messages in the message thread. If there is a new message you haven't seen yet, it will display a red number showing how many new messages you have for that particular thread. Similarly, an unread message counter will be displayed nexts to the messages tab on the sidebar, showing the total number of unread messages you have.

People and Organizations

From this tab you are able to create a new organization by providing it with a name and a short description. You can also view all the organizations you are currently a part of. By clicking on an organization, you can see all the members listed, as well as their current role. If you have an admin role for that organization, you can approve join requests, change users role, or kick members out of the organization.

Join organizations

From this tab you are able to see all the currently available organizations and request to join any that you wish. This will send a join request to the organization owner who can then approve the join request if the want. You will be automatically assigned the base role of member. If you wish to have your role changed, that must be done by a member of your organization.

Navigation Form

In the navigation section on the sidebar is the navigation form tab, where a user can search for a specific person in the search engine and see if the individual already has client data in the system. If not, a new input form will be displayed where the individual's name will be displayed in the name field, and the user will be able to fill in the individual's age, gender, number of kids, veteran status, and select the options they are lacking. The options include food, shelter, mental health help, and substance abuse recovery. At the bottom of the input form is a submit button, which will create the new individual as a client in our database. If the user isn't able to remember a specific individual or wants to separate the clients into categories, there are two filters under the search engine. The filters include toggling the different veteran statuses someone could have and the number of kids a client could have.

Mapping

The second portion of the navigation system is the mapping interface, where keeping in contact with the clients through direct interaction was the focus. At the top of the page is a drop-down of all the clients that exist in the system. Once a client is selected from that dropdown, the user is to click the green 'Place Marker' button right below the dropdown and select a location on the map. From there, a blue marker will show up on the map and will display the client's name and the date/time they were interacted with. If the user needs to clear out the markers for any reason, there is a red button called 'Clear Markers' that will fulfill that functionality.

Service Matching

Service matching is the last page in the navigation section, and this is the crucial page that pulls the streamlined workflow together. When this tab is clicked on, there is a dropdown that displays all the existing clients, age groups, gender, and how much help was wanted from the client. From there, the user will choose the correct demographic information and click on the 'Find Services' button at the bottom. Once the button is clicked, a selection of matched services will pop up and can be selected and submitted using the 'Submit' button. An alert will appear at the top saying that the referral for the client has been created.

Maintenance

The maintenance needed for the full stack application can be split with daily, weekly, and monthly tasks to ensure that everything is running as it should be. The most important maintenance would include monitoring all the logs from all the specific softwares that the project has. The main ones to look at for end frontend errors are the nginx logs as that is the main file where the react app is hosted. Here is a detailed list of what needs to be done on a scheduled basis.

Daily Routine

The biggest daily routine which is also recommended when there is an update to the software would be to check the logs for both pm2 (which hosts the backend) and nginx (which hosts the app as whole). This allows for you to see any errors and where specifically to fix them. Daily health checks on the system are also heavily recommended to ensure that the system is running as it should be.

Weekly Routine

This application depends on many libraries that both the frontend and the backend use that are able to make react so dynamic and user friendly. Every week the react app needs to be updated with any potential updates and bug fixes on these libraries. Simply go to wherever you have stored the backend and frontend of the system and run `npm outdated`. This will show you the apps that need to be updated for the success of the system. From there you can run

It is important to clear your logs on a scheduled routine. This will ensure the health of your EC2 instance. It is important not to overload the instance with unnecessary log files and do a weekly clearing of them or as needed. The log files can be found in `/var/log/syslog` or `/var/log/nginx`

Monthly Routine

Every month, the server should go through its general security practices both for the hosted system and packages, this is as simple as running `sudo apt update && sudo apt upgrade -y`

Code Malfunctions/Version Control

There may be instances where the code goes through an update and that may impact the system as a whole. This may cause the system to be down and the logs to be filled with too many errors to understand. It is important to keep any major change kept

on a version control system of choice so any faulty updates can be discarded and the original system can be restored.

The source code link can be found below:

https://github.com/HelloWorldByMe/HelloWorldByMe/tree/main

Troubleshooting

AWS EC2 Instance: Can't Connect to Instance

1. Make sure that the EC2 instance is running
    a. Log into the AWS EC2 dashboard
    b. Check that the instance state is 'Running', **not 'Stopped' or 'Terminated'**
2. Look at the security group rules
    a. The instance's inbound traffic must allow inbound traffic
        i. **SSH** (port 22) from EC2 public IP
        ii. **HTTP** (port 80) and/or **HTTPS** (port 443)
        iii. Any custom ports that are supposed to be used
    b. Edit the security groups if needed
        i. **Inbound > Add Rule > Choose the correct IP or IP range**
            1. (ex: 0.0.0.0/0 for public access if it's safe)
3. Make sure that the correct SSH command is used
    a. ssh -i /path/to/your-key.pem ec2-user@<ec2-public-ip>
    b. Confirm if the key is correct and has the right permissions
        i. chmod 700 your-key.pem (
            1. This would allow you to have read, write, and execute rights
    c. Double-check that you're using the right user
4. Check that the correct **public IP** is being used by navigating to the EC2 dashboard and selecting the instance

AWS RDS Not allowing a connection

1. Ensure that the EC2 instance is properly configured (see above)
2. Verify that credentials are correctly configured for the purposes you are using it for:
    a. Check username and password
    b. Check the port used (5432)

c. Lastly check the host endpoint (database-1.c9c60ay4es21.us-east-2.rds.amazonaws.com at the time of writing this)

3. On the EC2 dashboard navigate to the "**security groups**" tab on the left side of the screen under "**Network and security**"
    a. Select the security group that pertains to your connection
        i. If the group does not exist:
        ii. Create one
            1. Select the prudent information
            2. "**Type**" should match the purpose that you are using it for
                a. Depends on the reason for connecting but most popular database type of connections are listed and IPv4 or IPv6 should cover other connections
            3. "**Port Range**" should match the port used for the connection you are trying to make. (See that information on the softwares documentation)
            4. "Destination" can be set to "custom" to allow only one specific IP address, however other options like "Anywhere-IPv4/6" to allow any connection to be made
                a. If using a custom IP address, you will then need to enter that
    b. Select the "**VPC ID**" of the target security group
        i. You will be redirected to another page
    c. Select the **preferences** in the top right of the table (right below the "**Create VPC**" button, it will look like a settings button)
        i. Select the "**Block Public Access**" button
        ii. This will allow any inbound requests to enter

Setting up a connection to the database in PostgreSQL

1.) You will first need to install pgAdmin4 (https://www.pgadmin.org/download/)
2.) Once you are on the dashboard you will need to navigate the the "**Servers**" tab in the top left corner of the dashboard
    a.) Right click on "**Servers**"
    b.) Hover over "**Register**" where you will see the option to select "**Server...**"
        i.) You will be met with an interface allowing you to customize your server
        ii.) General

(1) Note: you must put in a name for this database
    (a) This name is purely for you to be able to identify the database, this will not affect the database or its name in any way.
(2) All other options are optional

iii.) Connection

(1) Under "**Host name/ address**" enter the endpoint of the RDS database
(2) "**Port**" should already be set to "5432" for you, but ensure that the port entered under here is the correct port for whatever database you may choose
(3) "**Maintenance database**" should be set to "postgres" or any other database you would like to use as a backup
(4) "**Username**" and "**Password**" needs to be filled in with the correct username for an account with administrative access
(5) All other options are optional depending on how you want to customize the security of the database

iv.) Parameters

(1) This is all purely optional, but may be modified to customize security

v.) SSH tunnel

(1) Only use if you are using SSH Instead of connecting through the RDS endpoint, this is not recommended.
    (a) If you elect to, you will need to select the "**Use SSH tunneling**" slider. Where you will then enter the IP address under the "Tunnel Host" field
    (b) "**Port**" will need to be set according to the SSH connection you are trying to make (default is 22)
    (c) "**Username**" and "**Password**" needs to be filled in with the correct username for an account with administrative access
        (i) You may also use a .pem file instead of a password for this you must upload the permission file

vi.) Advanced

(1) None of these options are necessary for connection, purely for customization, again.

c.) You should then be able to connect

i.) If for any reason you are not able to connect, try connecting again without any of the "optional" items mentioned above

d.) Once connected there are a few specific paths you must follow, to get to the actual data, you must first drop down the different folders in the following sequence: [DATABASE_NAME]/ Databases/ hello_WBM/Schemas/ public/ tables

i.) You should then be able to see the different tables currently being stored and then to query the data right-click on the desired table, then select "**View/ Edit Data**" then select either "All Rows" or the desired amount

ii.) You may also create tables by right clicking on the "**tables**" tab and entering in the information

(1) Another way is to use the "**PSQL Tool**" or "**Query Tool**" in the top left corner in the "**Object Explorer**"

Nginx & PM2

The main troubleshooting for nginx goes into the main configuration file. The easiest place to locate any potential errors that may be affecting the nginx. The most important command to remember is `sudo nginx -t` as that is where any errors may be noted. There could also be potential congestion so it is important to restart nginx once in a while if you notice an issue with the server command. Running `sudo systemctl reload nginx` will reload the everything and `sudo systemctl restart nginx` will rully restart the system.

PM2 carries the same general process where the certains server must be checked if there is no error, running pm2 list is able to show whether it is up properly or not. If for some reason it is labeled as stopped or errored, running `pm2 logs server` (or whatever you decide to name the server), will tell you where the issue may be. If there seems to be some performance issues with the server, a simple restart may be able to do the trick, the command would be `pm2 restart server`.

Conclusion

Thank you for this opportunity to work on this project with you! We are excited to contribute to an amazing mission that you are passionate about and hope to bring many

years of meaningful and efficient service to local service providers. It has been a rewarding experience for us, and we are genuinely proud of what we have accomplished together! As we move forward in our professional careers, we want you to know that we are happy to assist you with any brief questions you may have as you deploy and use our project. Our goal is to ensure a smooth transition and help you get the most out of the project.

With best wishes from your WorldByMe developers:

Elizabeth Knight - eak287@nau.edu

Joey Banaszak - jhb238@nau.edu

Jessica Maldonado - jmo366@nau.edu

Samantha Madderom - sgm257@nau.edu