A picture containing text

Description automatically generated

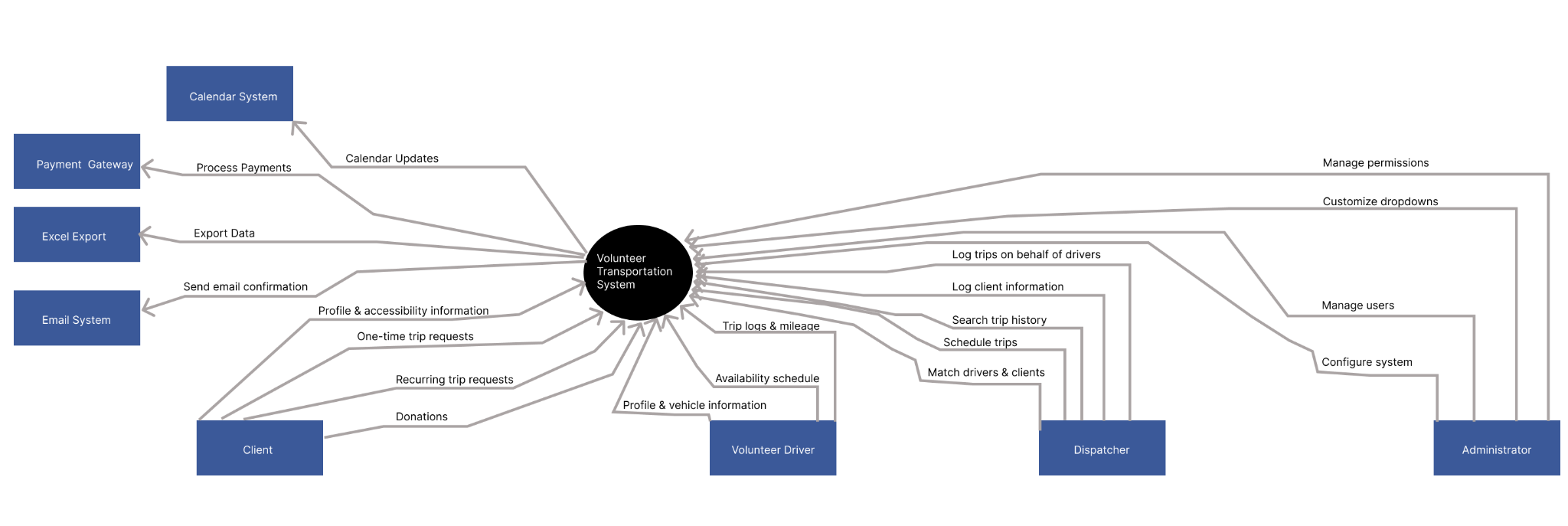
# Team 1 BAAAAM - System Architecture

# Overview

This document outlines the eight key sections that define our system architecture. Our goal is to develop a fully functional product that enhances the ride request and assignment system for our four clients as well as other transit agencies. This document in its current form is meant to serve as an idea of the components, interactions, and dependencies required for this product.

# Context Diagram:

[Figma](https://www.figma.com/design/Megmf9GJcdK48YfFoU5ODD/ISTE-500-BAAM-Context-Diagram?node-id=0-1&p=f)



# Technology Platform

### Frontend

The frontend application will be a React 19 + Vite application.

#### Packages (production):

* Tanstack Router
* Tanstack Query
* Tanstack Table
* Zod
* Ky
* Tailwindcss v4
* Zustand (?)
* Recharts
* Shadcn

#### Packages (dev):

* Typescript
* Eslint

### Backend

#### Backend (Api)

* Express.js
* Zod
* Postgres connector (pg)
* Jwt
* Typescript
* Node mailer
* (telephone gateway)

# Hardware

The application will be contained within deployable docker containers. There will be a dedicated server that will be able to run these containers without issue. The server should have a minimum of 250gb storage and 4gb of RAM with reasonable pricing.

# Security Model

### Application Authentication:

The application will utilize JWT web tokens as a method of Authorization for both the API and frontend application.

There will be two tokens:

* Refresh Token
* Authorization Token

The Authorization Token will have a TTL of 15 minutes, while the refresh token will have a TTL of 30 days. A username and password will be used to authenticate against the database, and then the Authorization and Refresh tokens will be generated.

Tokens will be generated with the users username and separate salt values for the Authorization and Refresh tokens. The username will act as a lookup key along with the server side salt values. These salt values should not be exposed anywhere and should be contained only to the server.

The Authorization token will be sent to the client and sent with all client side requests. Once this token expires, the refresh endpoint will be requested and if the refresh token has not expired, a new authorization token will be issued.

Both the user's username and password must be checked against the database. The database will store encrypted versions of passwords. Passwords being checked will be encrypted and checked against the database password value.

### Production SSL Encryption

Upon production deployment, a SSL certificate should be automated to issue valid certificates using LetsEncrypt. This will enable HTTPS.

### Production Reverse Proxy

Within the production environment, nginx will act as a proxy between the outside world and the local host ports

### Production UFW Firewall Rules

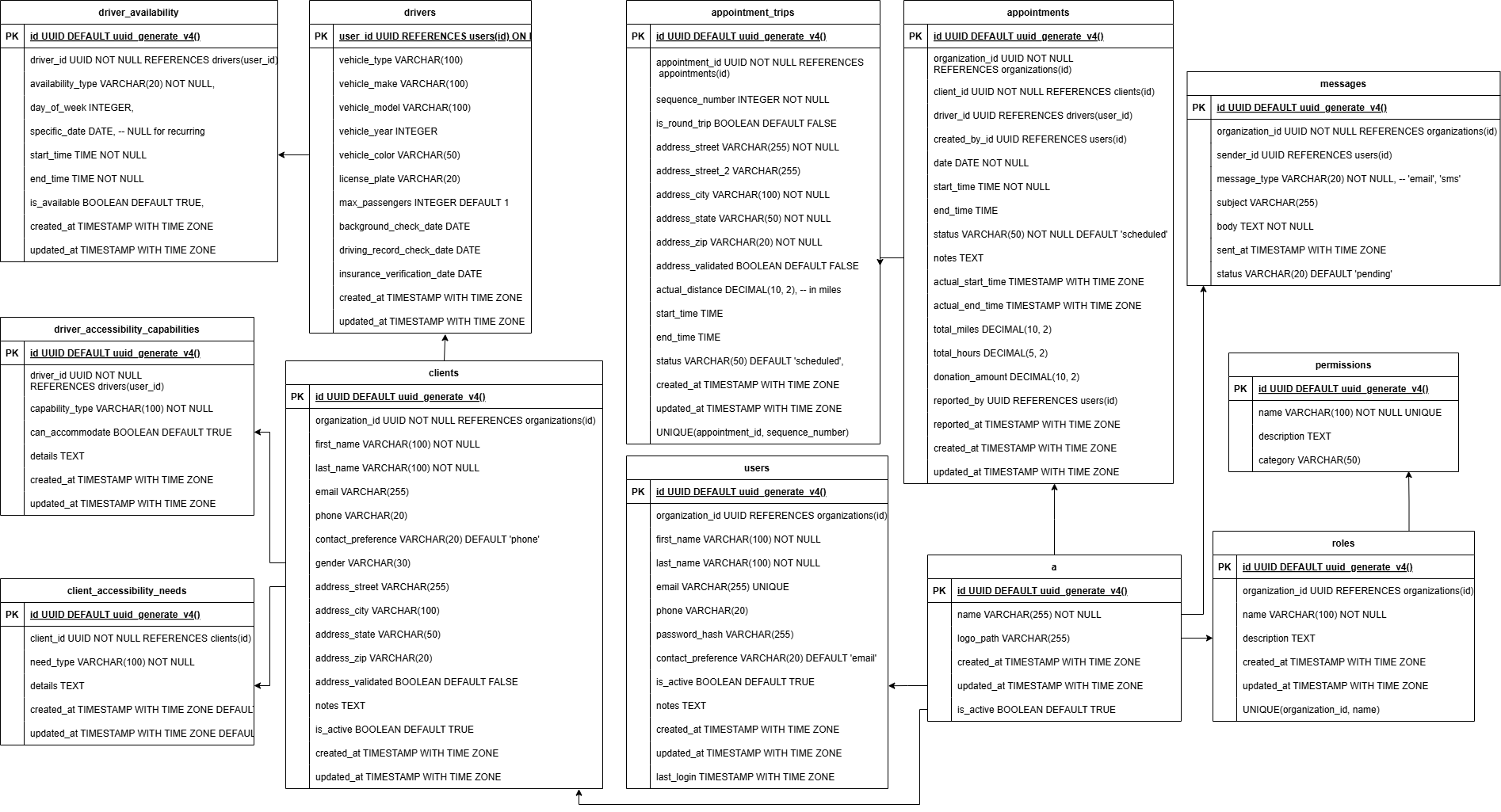
The production environment will have a

# Business Rules

* All data will be separated between organizations
* Superuser will be able to create organizations
* Organizational admin will be able to create and manage organization users
* Organizational admin will be able to create and manage roles and permissions, and assign permissions to each role
* Dispatchers will be able to manage the schedule of rides and assign drivers to rides
* Dispatchers will be able to record the hours and mileage of volunteer drivers, as well as donations received
* Drivers will be able to provide a schedule of their availability and report their hours and mileage to organizational staff
* Drivers will be able to specify their accessibility preferences and car information
* Drivers will be able to report their donations received

# SQL ER Diagram

Below represents a simplified diagram of our PostgreSQL database schema. This version is designed in such a way that it fits within this document. We also have a baaaam-schema.pgerd file of this diagram which allows us to view it in pgAdmin4 with more detail. Further details of this diagram can be found within the [SQL Data Dictionary](#_56x2l6rqlwdu).



# SQL Data Dictionary

[contains the same information as the transit-tables.sql file]

# 

# 

# 

# 

# 