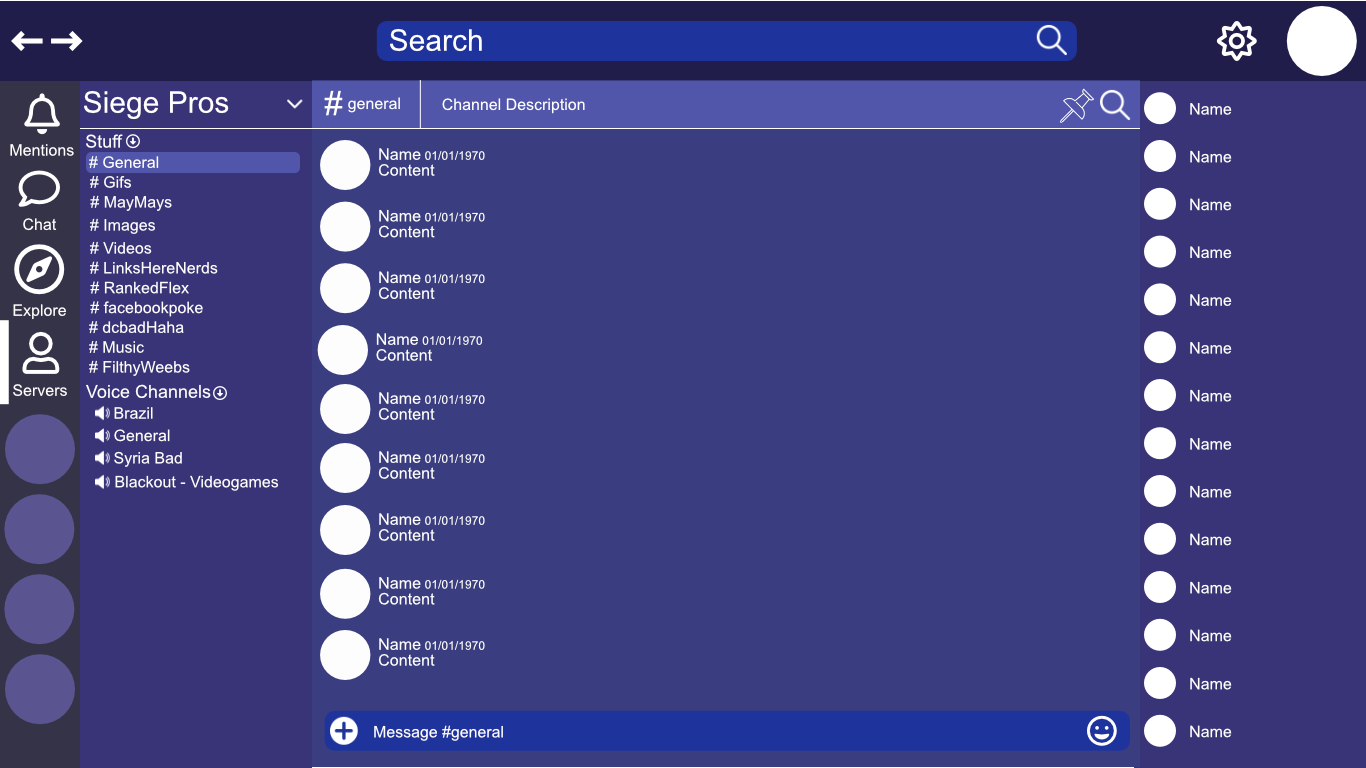
Database Setup for a Server



**Basic Setup**

Parlantos will exchange data through a variety of back-end web services that provide rest endpoints with data being stored in a Postgresql database alongside the filesystem for binary blobs. Unique ids will be generated by a snowflake id web service that will be queried to retrieve each id for newly created content. All content will have a created\_at timestamp alongside the unique id that can be used for sorting purposes. Server information will store the users that are in the server and will pull users information from the related user tables. All server information will utilize SQL design best practices and use different tables for distinct data.

**UI Data Access**

Database calls will be orchestrated by a rest endpoint every time the user clicks on a channel and a websocket connection will send the client info if they have new messages in different channels or will automatically update the current channel. Multiple rest calls will be made when scrolling through content and paging will be implemented in the orchestration endpoints to send the data to the client, with potential that the client can cache content to prevent further unnecessary lookups.

**Server Table**

Id: snowflake(bigint)(PK)

Created\_at: Timestamp

name: VARCHAR

description: VARCHAR

image: VARCHAR (Path to image)

The server table will be used to store each server and information for the server will be fetched using joins based on the server id with other tables that use the id as a foreign key to identify the server they belong to. This will be used to fetch relevant server settings, channels, and members.

**Text Channel Table**

Id: Snowflake (bigint)(PK)

Server\_Id: bigint Foreign Key

Title: VARCHAR

Description: VARCHAR

The text channel table is used to store the title and description of a text channel that is connected to a server id. This allows the text channels to be populated when a user requests a server.

**Messages Table**

Id: Snowflake (bigint)(Pk)

Created\_at: Timestamp

Content: VARCHAR

member\_Id: bigint FK

Text\_channel\_id: bigint FK

The messages table stores all of the messages that are sent and associates them with a text channel and all messages can be sorted based on the timestamp to retrieve messages for a channel.

**Reply Table**

Id: Snowflake (bigint)(Pk)

Created\_at: Timestamp

Content: VARCHAR

member\_Id: VARCHAR

Message\_Id: (bigint)(FK)

The reply table is to store each reply associated with a specific message, this can be used to search if there is a reply for each message and populate the set of associated replies to messages.

**Reactions Table**

Id: Snowflake (bigint)(Pk)

Created\_at: Timestamp

Content: VARCHAR

Message\_id: (bigint)(FK)

The reactions table is used to store the different reactions that have been made to a specific message and is associated with that message.

**Pinned Messages**

Id: Snowflake (bigint)(PK)

Created\_at: Timestamp

Message\_Id: bigint FK

Server\_id: bigint FK

Text\_channel\_id: bigint FK

The pinned message table is used to store the id of the messages that have been pinned to a certain text channel and it checks the pinned messages for the id of the text channel to populate the pins based on the associated message\_id.

**Voice Channel Table**

Id: Snowflake (bigint)(PK)

Created\_at: Timestamp

Server\_Id: bigint FK

Title: VARCHAR

The voice channel table is used to store the information about the voice channels in the server so that they can be populated in the client.

**Member Table**

Id: Snowflake(bigint)(PK)

Username: VARCHAR

Password: VARCHAR

Salt: VARCHAR

Display\_Name: VARCHAR

Name\_Tag: VARCHAR

The member table stores the basic information of a user, their username, password, salt, display name and name tag. The display name is the name that is displayed to other users and the name\_tag is generated to provide a unique identifier along with the display name. The username is what is used to identify the user for login purposes.

**Friends Table**

Id: Snowflake(bigint)(PK)

Member1Id (bigint foreign key)

Member2Id (bigint foreign key)

Primary Key (Member1Id, Member2Id)

The friends table is a bridge table used to identify the users that are friends with each other.

**Server Users Table**

Id: Snowflake(bigint)(PK)

MemberId(bigint foreign key)

ServerId(bigint foreign key)

The server users table is a bridge table to associate the members that are in a server.

**Server Roles Bridge Table**

Id: Snowflake(bigint)(PK)

Server\_member\_id(bigint)(FK)

Role\_Id: (bigint)(FK)

The server roles bridge table is used to store the roles that belong to a particular server.

**Roles Table**

id: (bigint)(PK)

name: VARCHAR

The roles table stores a role and its name.

**Permissions Table**

Permission\_id: (bigint)(PK)

name: VARCHAR

description: VARCHAR

The permissions table contains a permission name and description which binds to different privileges.

**Role Permissions**

Id: (bigint)(PK)

Role\_id: bigint FK,

Permission\_id: bigint FK

Role permissions is a bridge table that is used to tie a set of permissions to a specific role.