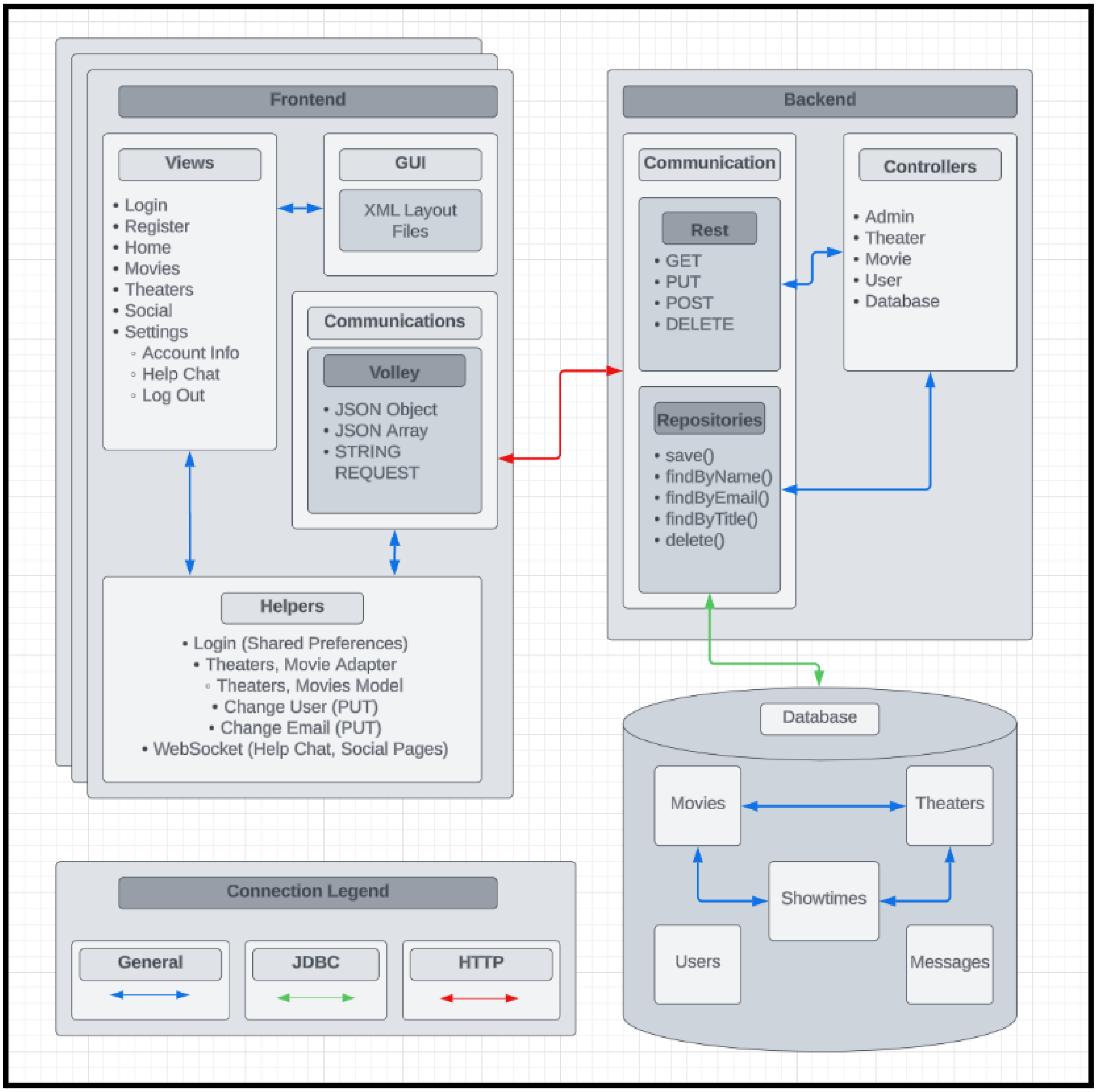
BLOCK DIAGRAM – COM S 309

Team LM_303
Movie Magnet

Nick McCullough
Lev Schaul
Tyler Orman
Ethan Griesman

Block Diagram



Block Diagram Description

Frontend:

Registration (User)

- Register a new account within the app
  - EditText: Username, Date of Birth, Password, Email
  - Button: Register
- "Register" button sends POST request to database to append new account to the total list of users.

Login (Admin, User)

- Takes user input for email and password entered during account Registration
  - EditText: Email, Password
  - Button: Sign In, Sign Up
- "Sign In" button (POST request to validate user login, GET to store username/email for later use in app)
- "Sign Up" button (redirects user to Registration)

Home Page (Admin, User)

- Landing page where users can swipe or tap button to navigate pages

Movies Page (Admin, User)

- Users search for movies (age appropriate, found with POST request) via search bar and are able to find where they can be shown.

Theaters Page (Admin, User)

- Users search for theaters via search bar (GET request) and can see what movies are available (age appropriate found with POST request).

Social Page

- Users chat with other users (divided by age group) via DMs or broadcasting for all to see.

Settings Page

- Users able to navigate to help chat room DM with admin user, edit account user and email (PUT requests), delete account (DELETE request) and log out

Backend:

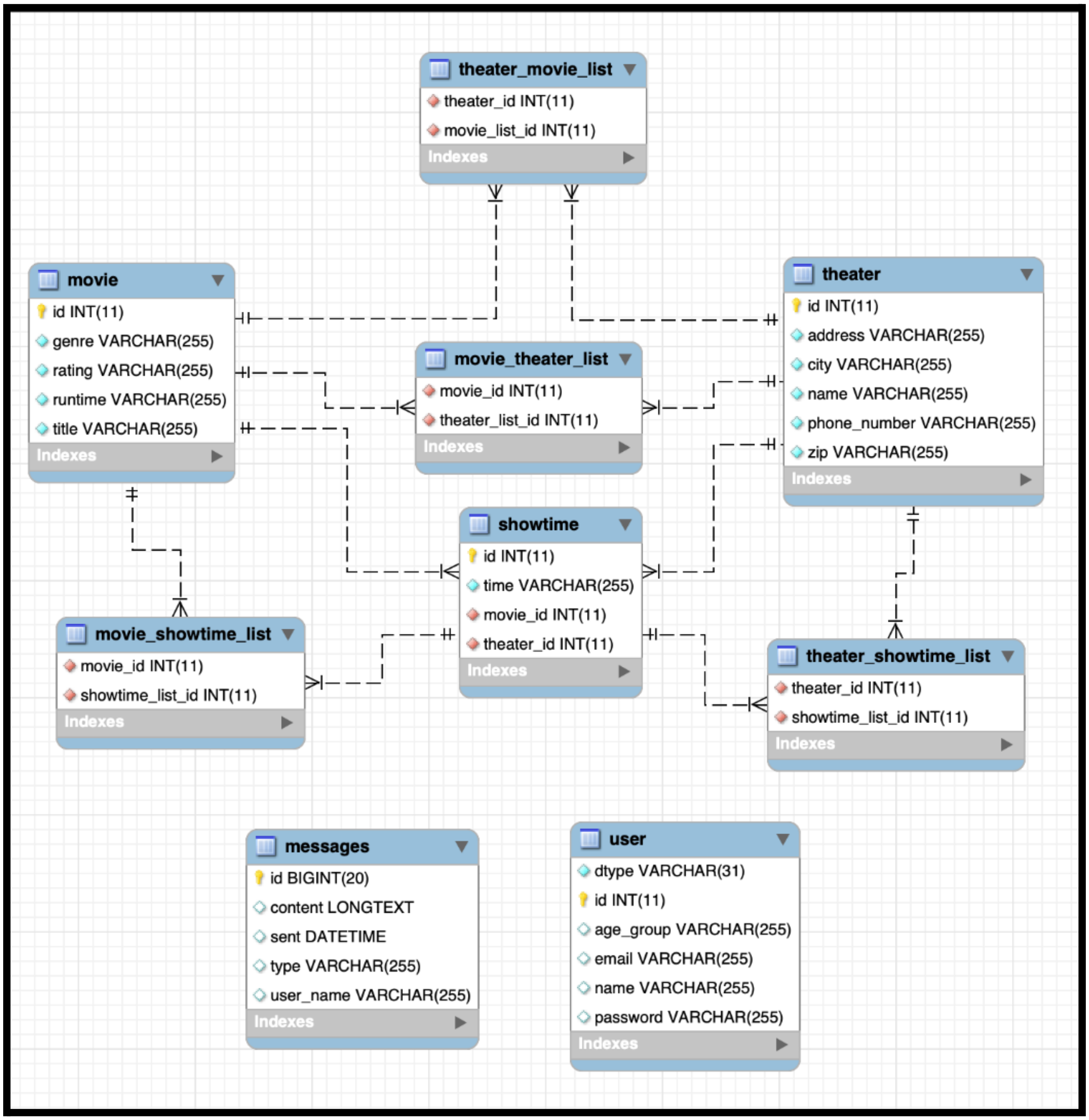
Communication - There are multiple mapping functions that update our database:

- POST: sends information for a new user to be added to the database, their ID is the email they used to login/create account with.
- GET: retrieving info such as an ageGroup, username, email, movie, theater.
- PUT: updates user's email and/or username, and movies/theaters/showtimes.
- DELETE: deletes a user/theater/movie from the database.

Controllers - We have several controllers to handle different backend functions.

- *DatabaseController.java* : includes POST mappings for initializing theater, user, databases, and DELETE mappings for deleting users from the database,
- *UserController.java* (passwords/auth): contains GET, PUT, POST, DELETE mappings for retrieving, updating, and deleting user information such as a user itself, a user's username, an email, and even password authentication.
- *TheaterController.java* : contains GET, PUT, POST, DELETE mappings to list all available theaters, update the database with current showtimes of movies playing at those theaters, creating and deleting theaters from our database.
- *MovieController.java* : contains GET, POST and DELETE mappings, retrieves a list of all movies from our database, search movies from our database, creating/initializing a movie from our database, and deleting a movie.
- *AdminController.java* : Contains GET, PUT, POST, DELETE mappings for a updating a user's username and email, email, creating users, viewing all users, retrieving a single user by email and then deleting a specific user.

Block Design Tables and Fields:



theater_movie_list
- theater_id INT(11)
- movie_list_id INT(11)

movie
- id INT(11)
- genre VARCHAR(255)
- rating VARCHAR(255)
- runtime VARCHAR(255)
- title VARCHAR(255)

theater
- id INT(11)
- address VARCHAR(255)
- city VARCHAR(255)
- name VARCHAR(255)
- phone_number VARCHAR(255)
- zip VARCHAR(255)

movie_theater_list
- movie_id INT(11)
- theater_list_id INT(11)

showtime
- id INT(11)
- time VARCHAR(255)
- movie_id INT(11)
- theater_id INT(11)

movie_showtime_list
- movie_id INT(11)
- showtime_list_id INT(11)

theater_showtime_list
- theater_id INT(11)
- showtime_list_id INT(11)

messages
- id BIGINT(20)
- content LONGTEXT
- sent DATETIME
- type VARCHAR(255)
- user_name VARCHAR(255)

user
- dtype VARCHAR(31)
- id INT(11)
- age_group VARCHAR(255)
- email VARCHAR(255)
- name VARCHAR(255)
- password VARCHAR(255)