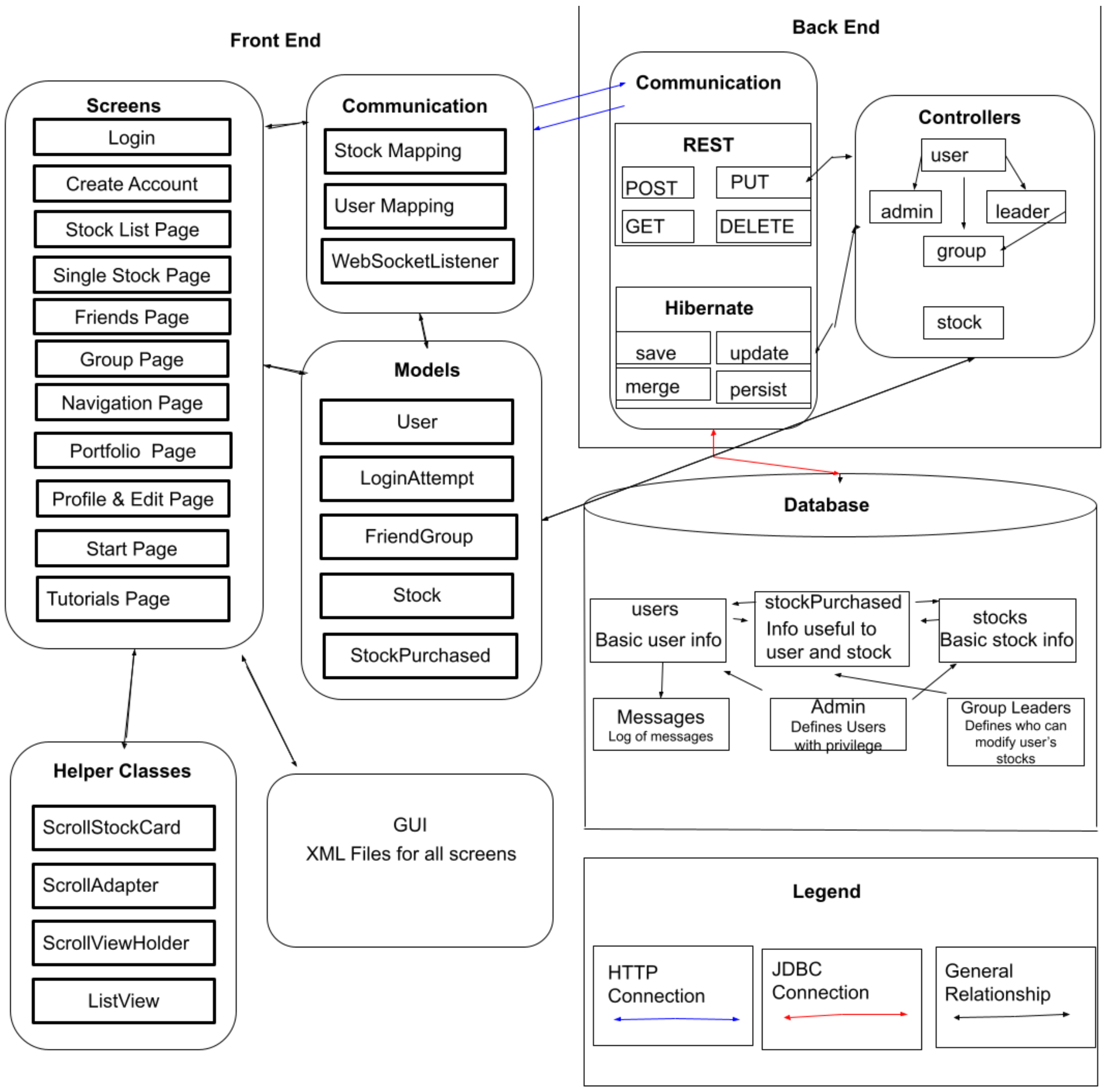
Block Diagram Report

MS_316

Nicholas Kokott, Shivansh Patel, Skyler Kutsch,
Joshua Whittington


Front End
Screens
Login
Create Account
Stock List Page
Single Stock Page
Friends Page
Group Page
Navigation Page
Portfolio Page
Profile & Edit Page
Start Page
Tutorials Page
Communication
Stock Mapping
User Mapping
WebSocketListener
Models
User
LoginAttempt
FriendGroup
Stock
StockPurchased
Helper Classes
ScrollStockCard
ScrollAdapter
ScrollViewHolder
ListView
GUI
XML Files for all screens
Back End
Communication
REST
POST
PUT
GET
DELETE
Hibernate
save
update
merge
persist
Controllers
user
admin
leader
group
stock
Database
users
Basic user info
stockPurchased
Info useful to user and stock
stocks
Basic stock info
Messages
Log of messages
Admin
Defines Users with privilege
Group Leaders
Defines who can modify user's stocks
Legend
HTTP Connection
JDBC Connection
General Relationship

Backend (All currently implemented)

Communication

The backend uses different mapping techniques to update the database based on the information sent to the given mapping's URLs. The techniques are:

- Get: requests information, takes an identifier for an item requested from the database and displays either a change or current information to the user.
- Post: sends new information to add an item to the database using a specific identifier
- Put: sends information to update an item in the database using a specific identifier
- Delete: deletes an item from the database with an identifier.

Controllers

The controllers give the mappings for communication between the frontend and the database. The controllers we will have are as follows:

- User: Uses all the mappings above to create, maintain, and delete themselves. Users have one to many mappings with itself and StocksPurchased. It has a many to one mapping with admin and group leaders.
- Group Leaders: Uses all of the above mappings to do the same as the users. The group leaders can manage the available stocks to purchase for the groups that they control.
- Administrator: Uses all of the same mappings as the users. However, the administrators can add and remove stocks and delete other users from the database.
- Stock: Uses all of the communication mappings to have stocks created, maintained, and deleted from the database. Stocks have one to many mappings with StocksPurchased objects. And a many to one mapping with administrator and group leaders.

Frontend (All currently implemented)

Models

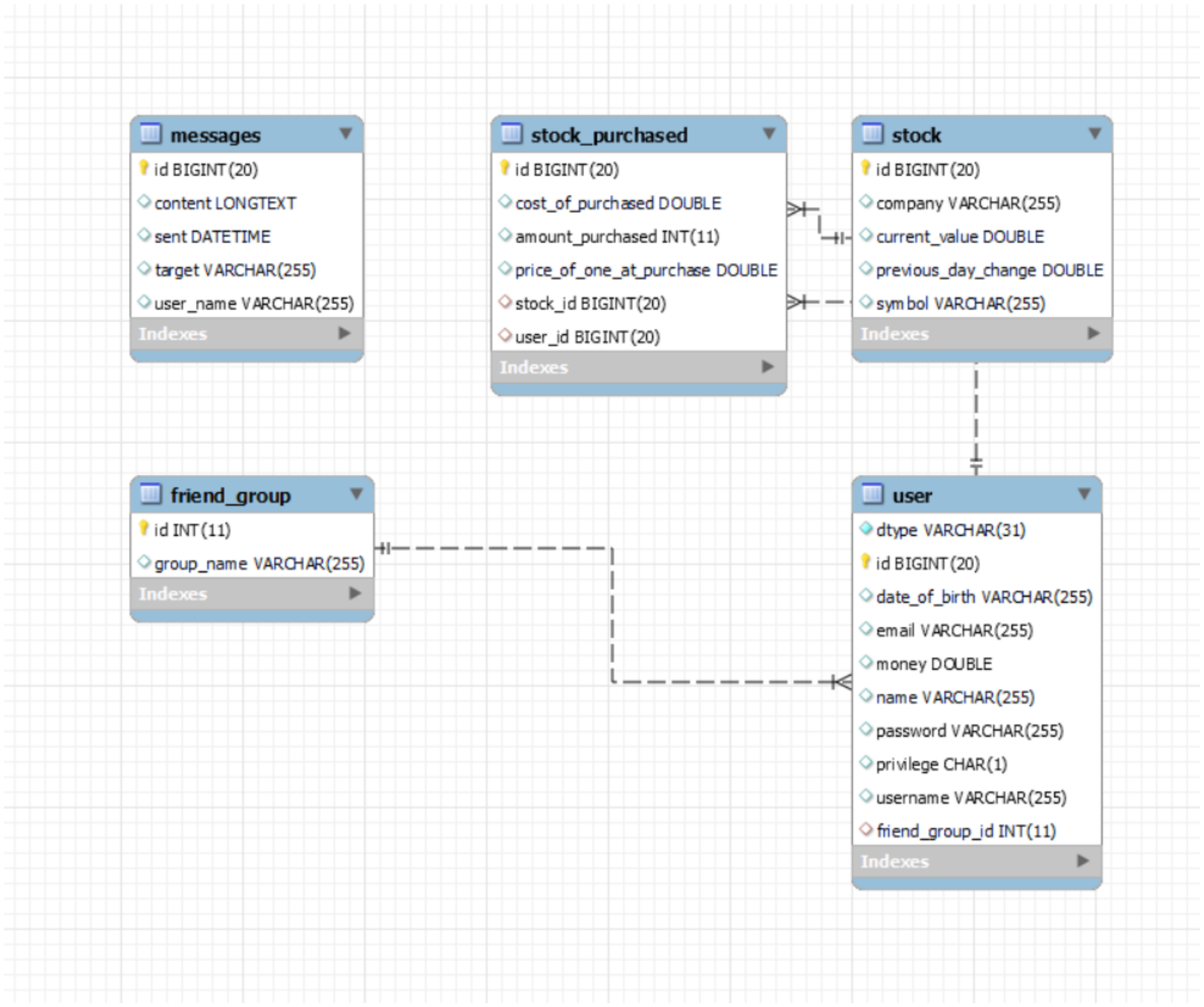
- Stock and StockPurchased: These models use the communication mapping to update, create, and delete stocks available in the stock list or stocks the user has purchased.
- User: Mirrors the database's user controller to send and receive updates on user data in a user object.
- FriendGroup: Mirrors the database's group controller to send and receive updates on group data in a group object.
- LoginAttempt: This model instantiates the global user after a successful login. It is also used to send a login request to the server in the expected format.

Communication

- UserMapping: Maps to the server endpoints created to access the user's stored data
- StockMapping: Maps to the server endpoints created to access a stock's stored data
- WebsocketListener: Maps to the server endpoints that instantiate and maintain the WebSocket server and connections

Helper Classes

- All classes in the Helper Classes container are used to display server responses.


messages
id BIGINT(20)
content LONGTEXT
sent DATETIME
target VARCHAR(255)
user_name VARCHAR(255)
Indexes
stock_purchased
id BIGINT(20)
cost_of_purchased DOUBLE
amount_purchased INT(11)
price_of_one_at_purchase DOUBLE
stock_id BIGINT(20)
user_id BIGINT(20)
Indexes
stock
id BIGINT(20)
company VARCHAR(255)
current_value DOUBLE
previous_day_change DOUBLE
symbol VARCHAR(255)
Indexes
friend_group
id INT(11)
group_name VARCHAR(255)
Indexes
user
dtype VARCHAR(31)
id BIGINT(20)
date_of_birth VARCHAR(255)
email VARCHAR(255)
money DOUBLE
name VARCHAR(255)
password VARCHAR(255)
privilege CHAR(1)
username VARCHAR(255)
friend_group_id INT(11)
Indexes