Design Document for Chirrup

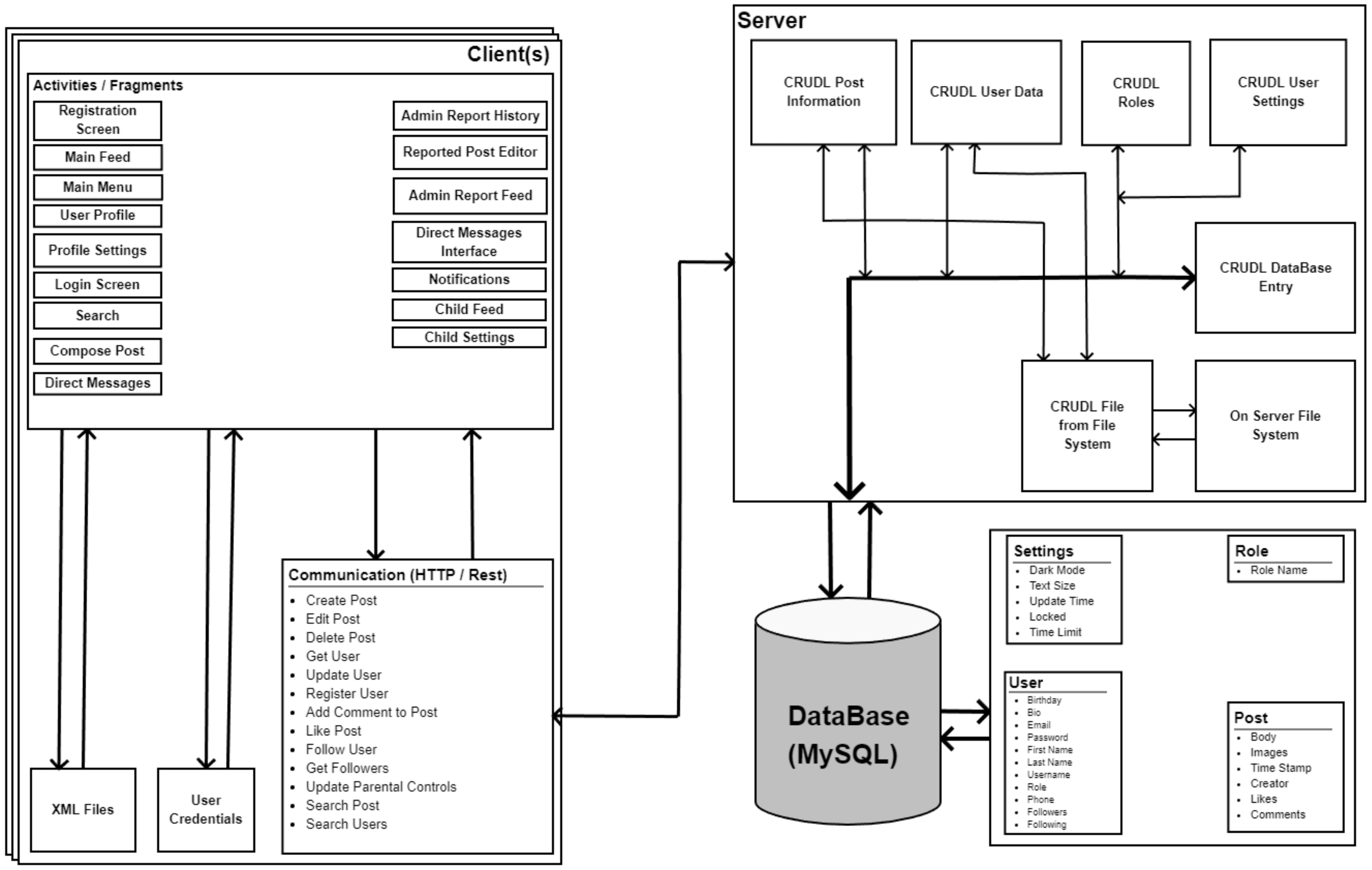
Group SR-2

Tyler Green: 25% contribution

William Zogg: 25% contribution

Jacob Boicken: 25% contribution

Jeremy Noesen: 25% contribution


Client(s)
Activities / Fragments
Registration Screen
Main Feed
Main Menu
User Profile
Profile Settings
Login Screen
Search
Compose Post
Direct Messages
Admin Report History
Reported Post Editor
Admin Report Feed
Direct Messages Interface
Notifications
Child Feed
Child Settings
XML Files
User Credentials
Communication (HTTP / Rest)
Create Post
Edit Post
Delete Post
Get User
Update User
Register User
Add Comment to Post
Like Post
Follow User
Get Followers
Update Parental Controls
Search Post
Search Users
Server
CRUDL Post Information
CRUDL User Data
CRUDL Roles
CRUDL User Settings
CRUDL DataBase Entry
CRUDL File from File System
On Server File System
DataBase (MySQL)
Settings
Dark Mode
Text Size
Update Time
Locked
Time Limit
Role
Role Name
User
Birthday
Bio
Email
Password
First Name
Last Name
Username
Role
Phone
Followers
Following
Post
Body
Images
Time Stamp
Creator
Likes
Comments

Activities / Fragments

Our application will have a large amount of activities and fragments to make up the entirety of the graphical user interface. The first screen new users will see is the registration screen, and returning users will see the login screen. Standard, parent, and child users, upon logging in, will be presented with the main feed of posts from people they follow. On this screen, there will be a button to compose a new post, as well as the main menu button. From the main menu, users can navigate to other pages, including their profile, settings, search, direct messages, and notifications. Parent users will also have the ability to navigate to their child feed and settings.

The profile page will show basic user info at the top and their own posts in a feed at the bottom. Settings and child settings will show full user profile data at the top, and a list of settings or permissions at the bottom. Search will show a search box and allow users to search for posts, users, or messages. Direct messages will show a list of people you have a conversation with, and clicking one will open that conversation. Notifications will show a feed of new updates on your posts or people you follow. Child feed will be identical to the main feed, but accessible to the parent user.

Admin users will have a very different layout to their overall app. They will only have a main screen that shows reported posts and previously reported posts, and another screen that allows them to respond to reports.

Communication (HTTP / Rest)

In our application, HTTP and Rest are the backbone or all server communication that we use in the front end. A majority of the server communication we do on the front end involves sending some sort of request to the server. Tasks such as updating users will utilize PATCH requests, creating users with PUT requests, and receiving users with GET requests. We take a user id, which is then added to the end of a URL for a set of data such as settings or user data. The server handles these requests, and sends us back data in returning. When asking for a user, this will be a JSON containing data for the specified user, but for sending it will be a confirmation that data was uploaded.

Local Files (XML Files and User credentials)

Locally stored files will hold data that is meant to allow the front end to connect and authenticate with the backend, such as user credentials. The XML files within the app handle UI layouts, string values, bitmap elements, and other visual elements within the application.

Server

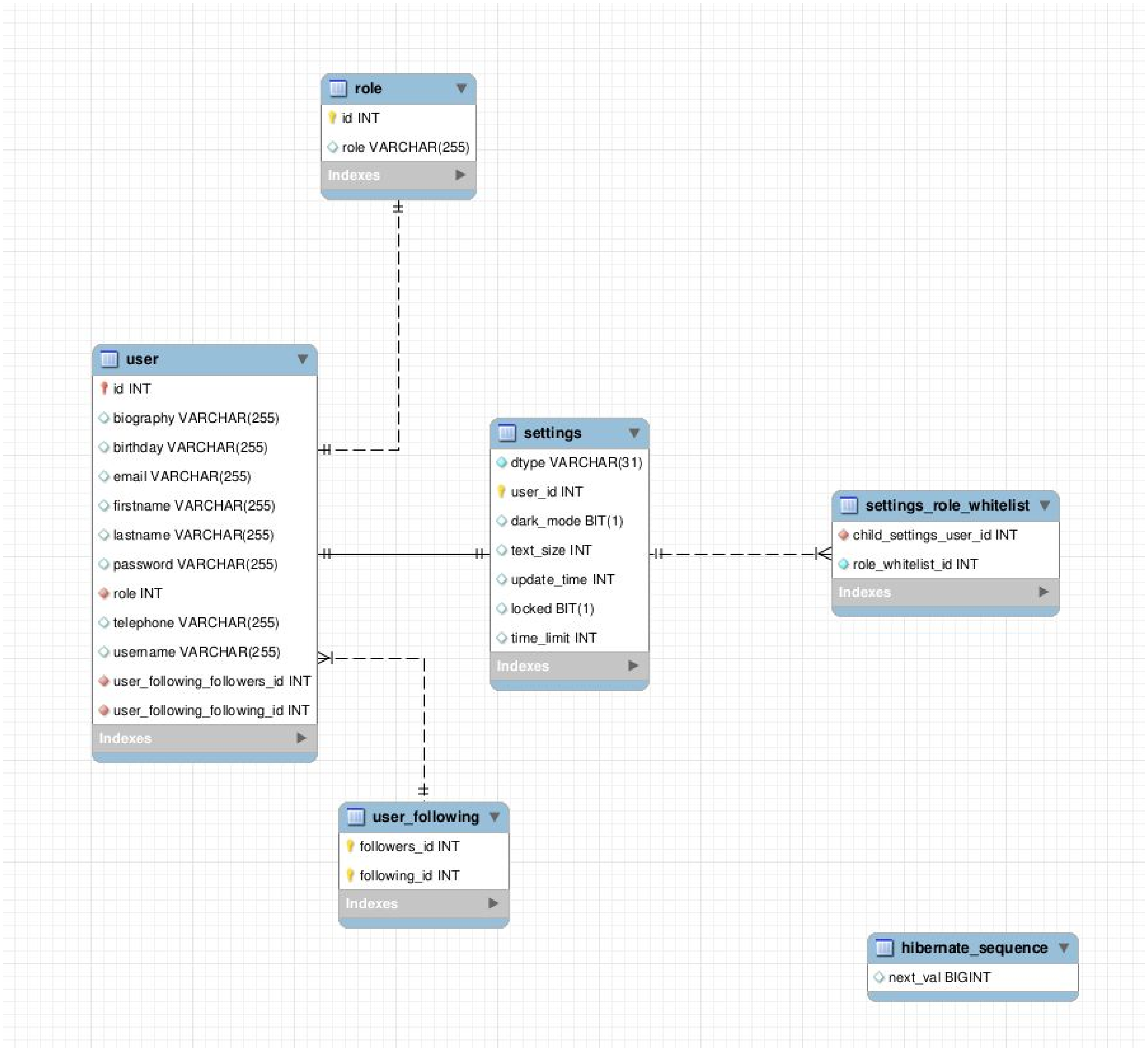
The server allows for CRUDL requests to be made in order to manipulate entries in the following database tables: user, settings, role, follower, and post. Internal services handle connections to the server to interact with the database. The connecting client submits an id in the URL or sends JSON data in the body of a request. The request type and data is parsed by Springboot to perform different gets and saves on the database. For instance, a client can partially update a user's data entry without needing to send an entire user JSON object using a PATCH request.

Server files

Files stored in the server will mostly include images, which will be uploaded to the server with a post request and can be located by a file location stored in the database.

Database (MySQL)

The database used in our backend will have the following tables: user, settings, role, follower, posts, likes, and comments. The user table will hold all the information relevant to a specific user, such as username, email, and password, as well as some other info which is used in the user's profile such as biography and profile picture file location. The settings table will be used to store settings for a specific user, including preferences for dark mode, font size, and other parental controls like a time limit for child accounts. The role table is used to list all the roles used in the application. In order to allow users to follow other users, a follower table stores a user-user relationship. The post table is used to store all posts made by any user. They will store a body, extra multimedia (file locations), creator, and timestamp information. In order to keep track of likes and comments on a post, other tables will be used as a many to many relationship. Many comments can be made on many posts, and the same for likes. The database is connected to the client side via SpringBoot (REST API), which then uses hibernate in order to create and update the database based on entities (user, role, settings, etc.) within the server.


role
id INT
role VARCHAR(255)
Indexes
user
id INT
biography VARCHAR(255)
birthday VARCHAR(255)
email VARCHAR(255)
firstname VARCHAR(255)
lastname VARCHAR(255)
password VARCHAR(255)
role INT
telephone VARCHAR(255)
username VARCHAR(255)
user_following_followers_id INT
user_following_following_id INT
Indexes
settings
dtype VARCHAR(31)
user_id INT
dark_mode BIT(1)
text_size INT
update_time INT
locked BIT(1)
time_limit INT
Indexes
settings_role_whitelist
child_settings_user_id INT
role_whitelist_id INT
Indexes
user_following
followers_id INT
following_id INT
Indexes
hibernate_sequence
next_val BIGINT