PO3: Block Diagrams

`BUILT-IT

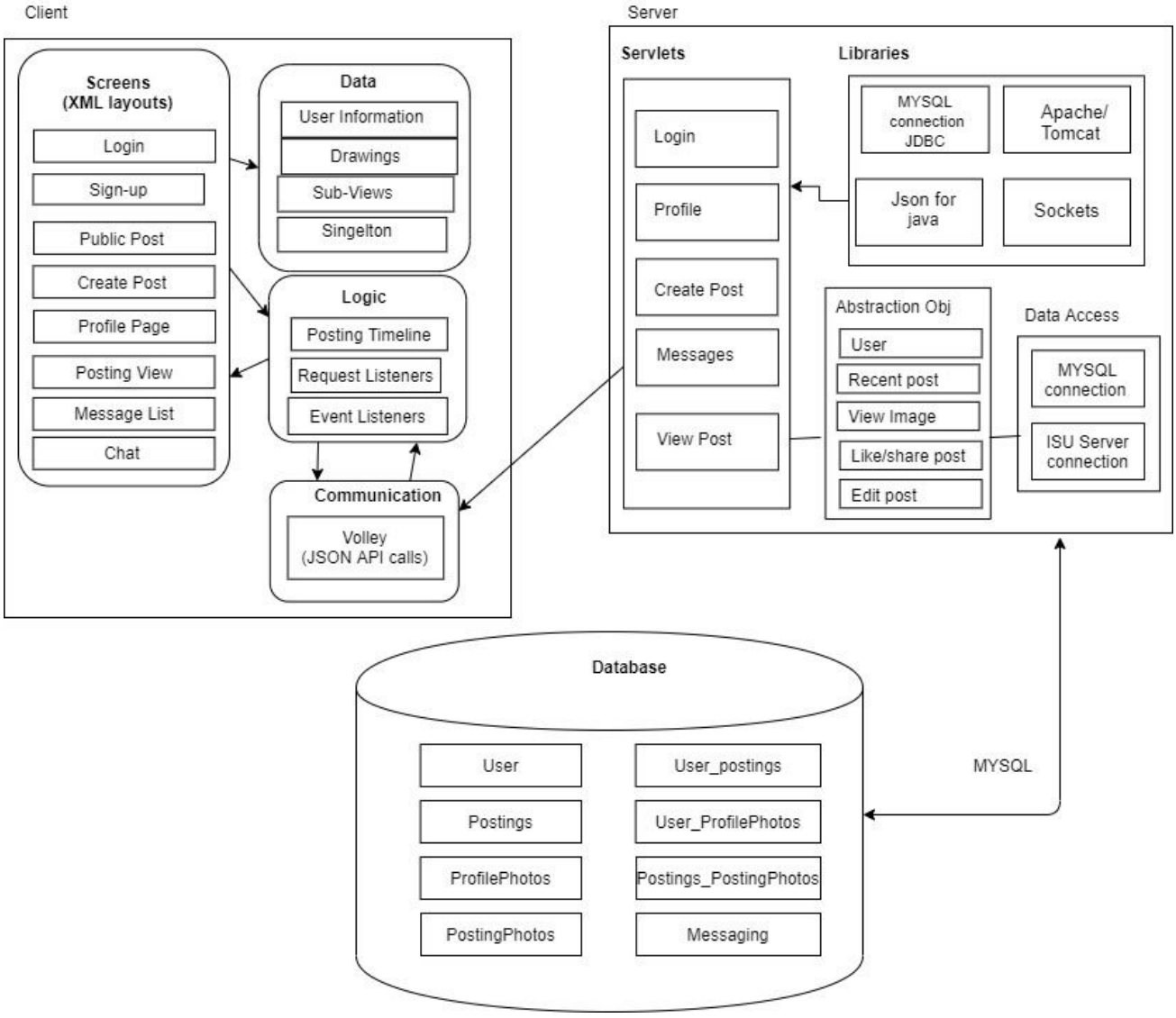
VB_4

Ismael Duran

Kenyon Thomas

Mohit Saroye

Cody Federicks


Client
Screens
(XML layouts)
Login
Sign-up
Public Post
Create Post
Profile Page
Posting View
Message List
Chat
Data
User Information
Drawings
Sub-Views
Singelton
Logic
Posting Timeline
Request Listeners
Event Listeners
Communication
Volley
(JSON API calls)
Server
Servlets
Login
Profile
Create Post
Messages
View Post
Libraries
MYSQL connection JDBC
Apache/ Tomcat
Json for java
Sockets
Abstraction Obj
User
Recent post
View Image
Like/share post
Edit post
Data Access
MYSQL connection
ISU Server connection
Database
User
User_postings
Postings
User_ProfilePhotos
ProfilePhotos
Postings_PostingPhotos
PostingPhotos
Messaging
MYSQL

Android Screens

The screens in our application will develop threads for event listener and request listener. Each screen has its own XML layout that will be what the user sees. To help us display what on the screen we use additional XML layout to display a component for part of the screen. Whenever a request needs to be made it creates a request to the server.

Android Data

The data is used to help the development of the screens. The user's information is stored. Drawings are developed to style components. As well as the implementation of a recycler view to display the most recent post on the feed.

Android Logic

The logic is what shows happen when the page is opened and what shows happen when an event or request happens. The event and request will create separate threads. The threads are listeners waiting for something to happen. Requests will make a JSON call to the server then it will interact with the screen to the given commands. When an event happens it will interact with the screens to do the given commands.

Android Communication

To communicate with the server the Volley library is used to make browser requests to the server. Volley has implemented thread functionality when requests are made. It will give the request listeners the response of the call. All the browser necessary request responses were built in the server.

Server:

The server uses JSON format to make the connections between backend (spring framework) and Frontend (Android Studios). Such that the servlets execute the required functionality of the app working alongside with the abstraction objects. Where, abstraction object allows the client to edit the required changes in the profile/post.

Database:

The server side uses the MYSQL database to store all the required data of the application on the ISU server using the annotations of mapping provided by Spring. Which allows java object classes to change Database via JDBC MYSQL Queries. Spring allows the connection with android studios via datasource url API calls. Database has various tables with all the required data used and saved by the app users.

Table: Postings - Many-To-One Relationship With User

Date – Every Time a post is created the date acts a primary key
Compensation – The stipend given when taking on the project
NeededRoles – Used to identify people who can accomplish project's goal
Picture – Users can post pictures of their ideas or team
ProjectDescription – Describes the project in detail
ProjectName – Self-descriptive
ThumbnailDescription – Shortened version for the description of the project
User_id – Identification of who made the post; also a foreign key

Table: Users

Id – Auto generated upon creation of account
Email – Self-descriptive
FirstName – Self-descriptive
LastName – Self-descriptive
PhoneNumber – Self-descriptive
Password – Self-descriptive

Relationship Table: Users_Postings

UserId – Corresponds to User table
PostingDate – Corresponds to Posting Date

Table: ProfilePhotos - One To One with User

Name – Photo Name
Size – Size of image
Format – Format of the photo

Relationship Table: Users_ProfilePhotos

ProfilePhoto_Name – Connects to ProfilePhotos table
User_id – Connects to User table

Table: PostingPhotos - One To One With Postings

Name – Photo Name
Size – Size of Image
Format – Format of the photo

Relationship Table: Postings_PostingPhotos

Posting_Date - Corresponds to Posting Date
PostingPhoto_Name - Connects to PostingPhotos table

Table: Messaging – Many To Many Relationship with Users

MessageID – Auto Incremented when a message is sent
User1 – Self Descriptive
User2 – Self Descriptive
Message – Message Associated