# Design Document for MongoTickets

Group 4_Rasel_8

Justin Moran:  25% contribution

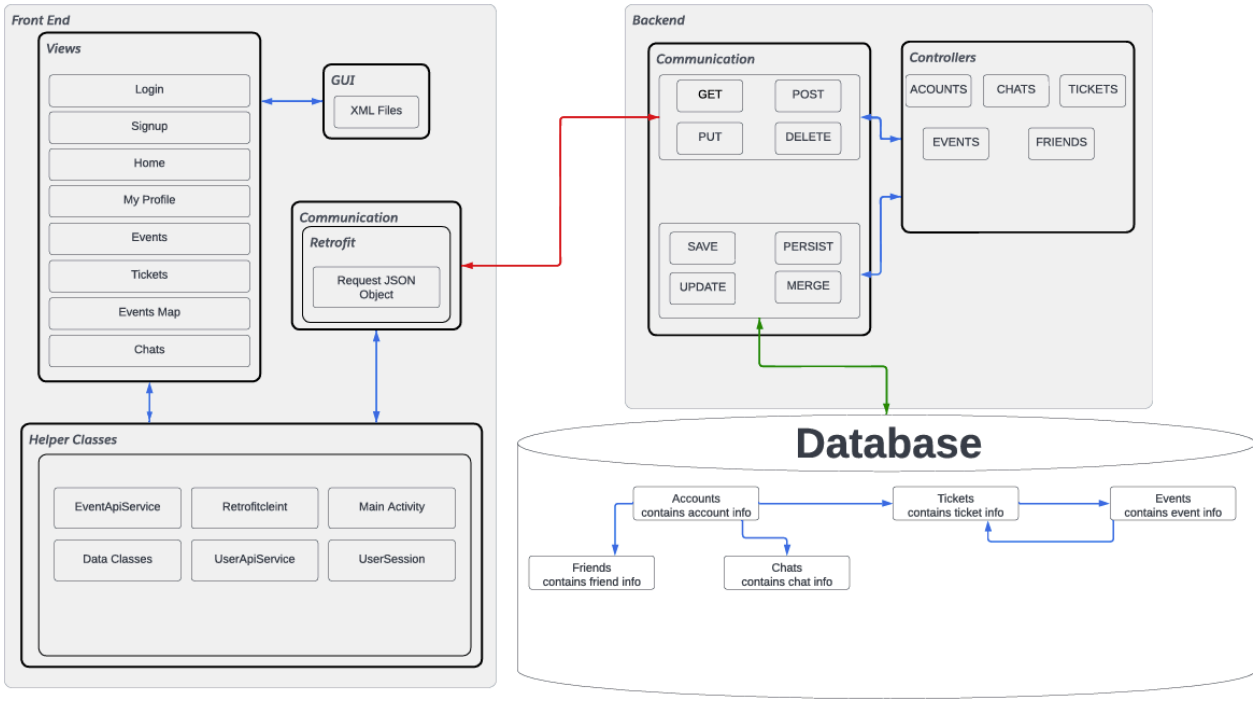
Prerak Jain: 25% contribution

Pablo Leguizamo: 25% contribution

Anoop Boyal: 25% contribution

# Application Architecture Diagram

4_Rasel_8 P03_BlockDiagramsAndAPI



**Frontend (currently implemented)**

- Login:
    - Edit Text: Enter Email
    - Edit Text: Enter password
    - Button: Login
    - Button: Don't have an account? sign up
    - This page has fields to enter your email and password and login to your account. It also has a button to sign up for an account in case you don't have an account.
- Signup:
    - Edit Text: Enter Username
    - Edit Text: Enter E-mail
    - Edit Text: Enter Password
    - Button: Signup
    - This page has fields to enter your username, email, and password which you will use to login to your account. Upon entering the fields and clicking the Signup button, your account will be created and added to the database.
- Home: Displays the user profile along with a graphic of the different events that you can find on the app:
    - TextView: emailTextView
- My Profile: Displays the account information of a given user, with the following elements:
    - ImageView: userImage
    - TextView: emailTextView
    - TextView: usernameTextView
    - TextView: idTextView

- Events: Displays every event in the database, with the most recent events being at the top of the list. You can search for specific events using the search bar, and can also book tickets to see all of the tickets for a given event, and can see the average ticket price of a given event with the button to the right of the book tickets button
    - EditText: search_bar
    - TextView: eventNameText
    - TextView: eventDetailsText
    - Button: bookTicketsButton
    - Button: average_ticket_price_button
- Tickets: Displays the tickets available for every event, showing the event name, date of event, ticket price, and lets you choose the quantity of tickets:
    - TextView: ticketEventsName
    - TextView: ticketCost
    - TextView: ticketDate
    - TextView: ticketQuantity
- Chats / Groupchats:
    - Button: Create Chat
    - Button: Chat
    - Edit text: Type a message
    - Form: select friends to add
    - This page pulls up all your current chats. You can create a chat by clicking the “Create chat button” which then displays a form which contains a list of your friends. You can select one or many friends to make a chat or groupchat. The created groupchat has a “chat” button which pulls up a chat window that uses websockets to communicate. You can enter a message and send it.

- Friends:
    - Edit Text: search friends
    - Button: Add Friend
    - Button: Remove Friend
    - The friends page contains a search bar to search for friends and a button to add friends. On entering a valid ID and pressing enter, the friend will be added to the database and will be displayed as a card. The card will have a “Remove friend” button which will remove the friend from the database.
- Events Map:
    - Button: Center on Ames
    - This page shows a map using the google maps API. We intend to use this to display events close to you

**Backend**
*Communication*

The backend uses mappings to update the database based on information sent to the given mappings’ URLs. These include: • Post: send information on an item to be added to the database. • Get: request information, often with an identifier for the specific item requested from the database • Put: send information to update a specific item in the database • Delete: send an identifier to delete a specific item from the database

*Controllers*

The controllers contain the mappings for communication between frontend and the database. These include: • User: Contains the above mappings to create users, which contain one-to-one relationships with the below foodtruck, customer, and admin type users. • Customer: This adds a few customer-specific variables, including the option to upload a profile picture and is connected by the username to the User table. • Foodtruck: Similar to customers, it adds other variables such as menus and location. • Reviews: This has a one to many relationship with foodtrucks and customers, where customers leave reviews on businesses that are stored in the reviews controller. • Favorites: Same as reviews with a one to many relationship, customers can favorite foodtrucks for quick viewing in the frontend.

PUT THE TABLE RELATIONSHIPS DIAGRAM on this fourth page! (Create the picture using MySQLWorkbench)