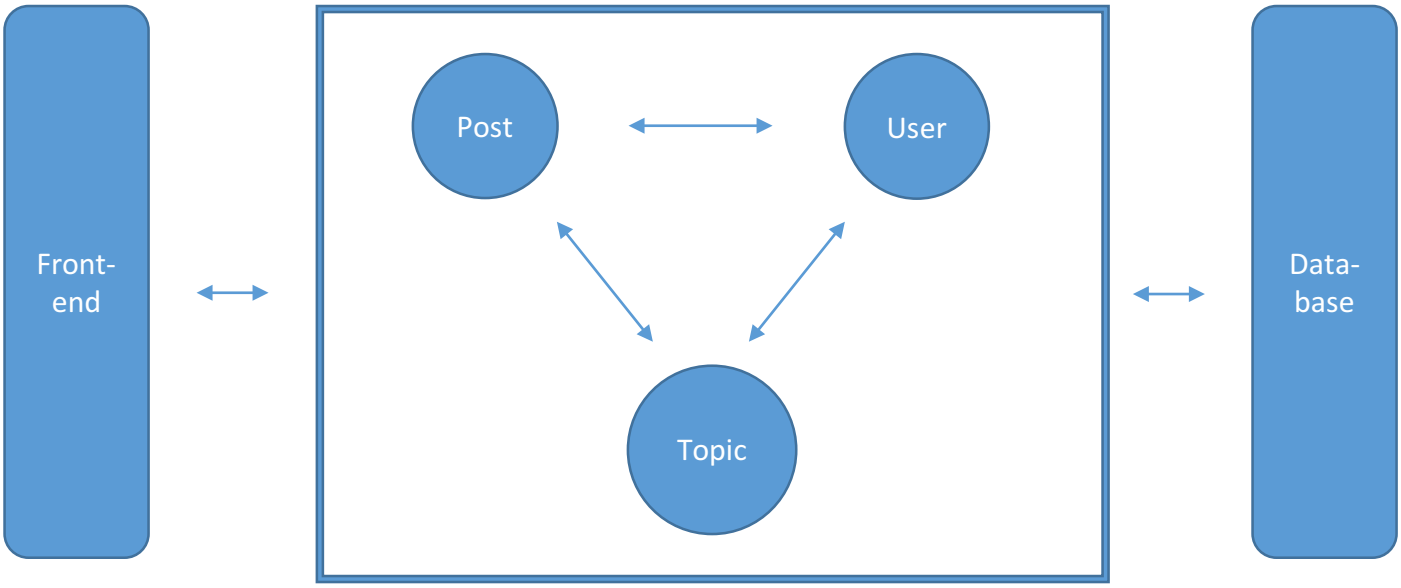
TechHub – Project Report

Design:

Our application revolves around three main components, users, posts, and topics (tags). These components interact with the database both simultaneously and separately, and also the html pages made for them.



Users:

Upon registering, each user is asked for basic information such as username, password, and email, later on they can choose to add extra information such as location, occupation, etc. Once a user is created, this basic information along with any unset information (stored as "Not set") is sent to the database and stored. Among these are fields such as upvotes, topics, posts that are not populated since the user has just registered.

Posts:

Each user can share their expertise on a given subject by posting. There are three types of posts: reviews, questions, and replies. For reviews and questions, the user has to select some topics that are related, if a topic is not suggested, the user can add new topics with the controls on the html page. Through posting, different fields in different collections are updated/added. For example, topics are added in the topics collection (if it doesn't exist previously). In the user collection, the newly created post will be linked with its author. Posts not only interact with topics and users, but other posts as well. The third type of post, replies/comments, are created with only the necessary information that can link to the original post (i.e. no topics, ratings, etc).

Topics:

Related to posts are topics, these act like tags that you see on blogs or Instagram. It provides the user more agility to search or view what they specifically want to view. A topic contains a field that an array of the users currently following that topic, and similar once a user follows a topic, the topic id will be added to the according fields in the user collection.

Security

We have taken two different approaches to lower the security vulnerability. First, all passwords that are created upon registering are first hashed then stored in the database. If a user tries to log in, the entered password is hashed again and checked to see if for matches. All hashing actions are done on the client-side to prevent “man-in-the-middle” attack. If a hacker can successfully intercept the communication between the client and the server, the intercepted message is still not readable since it is already hashed. The hashing function we utilized is called the MD5 hashing function. The second measure to prevent malicious actions is by storing the session id of the logged in user. For example, when a user logs in a session id is generated and stored both in the cookie and the database. When said user attempts to do any high risk actions (such as changing passwords, information, or deleting content), the server checks the session id stored in the cookie with the server, if any mismatch happens, those actions are stopped. This prevents cookie forging and prevents malicious activities when a user forgets to logout on a public computer.

Performance