CMPE 273
Enterprise Distributed Systems

Quora

By Team 10

Koushik Kumar Kamala (013766571)

Madhusudhan Redddy Shagam (013707187)

Naveen Ravipati (013756028)

Sai Krishna Reddy Jali (013752440)

Sanith Kumar (013312273)

Project Contributions:

Koushik Kumar Kamala:

Partial implementation of the database schema. Implemented the backend Part.

Madhusudhan Reddy Shagam:

Implemented the user feed, Answers, Bookmark, Answer Anonymously.

Naveen Ravipati:

Implemented the messages, notifications, and search.

Sai Krishna Reddy Jali

Implemented profile and Questions and other frontend parts.

Sanith Kumar :

Partial implementation of the database schema. Implemented backend part

Objective:

The Objective of this project is to build a website that is very much likely as Quora in terms of functionality and look or UI. Quora is Questions and answers website, where a user asks a question and someone who knows the answer that question. Users can edit, suggest, upvote, downvote a particular answer.

The User can log in, register, deactivate, delete and edit user details. They can ask a question and answer them edit their answer.

Frontend:

The Frontend is built react.js which is a javascript library. The redux is on the top of the react for state management. Redux makes the state management really handly. The user makes the calls from the frontend and it requested data is sent using API calls to the backend. The middleware of the project is discussed in the next section of the report.

Middleware:

Middleware plays an important role. The Middleware composes of two parts, one is node and other is Kafka-backend. The node takes the request from the frontend and then to the Kafka queue. From the Queue, it is sent to the Database for fetching the Data.

Database:

The Database Tier consists of the data tables of the user.

The Answers Schema:

```
const answers = new mongoose.Schema({
    question_id:{
        type: String
    },
    answer:{
        type:String
    },
```

```
    user_id:{
        type:String
    },
    user_id_upvoted:{
        type:Array
    },
    user_name:{
        type:String
    },
    profile_credential:{
        type:String
    },
    owner_status:{
        type:String
    },
    images:{
        type:Array
    },
    is_anonymous:{
        type:Boolean
    },
    upvotes:{
        type:Number
    },

downvotes:{
    type:Number
},
comments:{
    type:Array
},
    date_time:{
        type:String
    }
```

The Questions Schema

```
var mongoose = require('mongoose')
```

```javascript
const questions = new mongoose.Schema({

// question_id:{
//     type:String
// },
count:{
   type:String
},
views:{
   type:Array
},
question:{
   type:String,
   text: true
},
user_id:{
   type:String
},
owner_status:{
   type:String
},
topics:{
   type:Array
},
followers:{
   type:Array
},
// answers:{
//     type:Array
// },
date_time:{
   type: String
}

},{autoIndex:false})
var Questions = mongoose.model('Questions',questions);
module.exports = Questions;
```

The Topics Schema

```
var mongoose = require('mongoose')

const topics = new mongoose.Schema({topic_id:{
   type:String
},
   topic_name:{
       type:String
   },
   questions:{
       type:Array
   }
   ,
   users:{
       type:Array
   },
   followers:{
       type:Array
   },

},{strict:"false"})
var Topics = mongoose.model('topics',topics);
module.exports = Topics;
```

## The Users Schema

```
var mongoose = require('mongoose')

const userDetail = new mongoose.Schema({
email_id:{
   type:String
},
password:{
   type:String
},
first_name:{
   type:String
},
last_name:{
```

```
        type:String
    },
    status:{
        type:String
    },
    city:{
    type:String
    },
    state:{
        type:String
    },
    zip_code:{
    type:String,
    },
    profile_image:{
        type:String
    },
    education:{
        type:String
    },
    career_info:{
        type:String
    },
    about:{
        type:String
    },
    credentials:{
        type:Array
    },
    activity:{
        type:Array
    },
    messages:{
        type:Array
    },

    questions_followed:{
        type:Array
```

```
},
answers_bookmarked:{
    type:Array
},
topics:{
    type:Array
},
followers:{
    type:Array
},
following:{
    type:Array
},
profile_views:{
    type:Array
},
questions_answered:{
    type:Array
},
notification_list:{
    type:Array
},
},{strict:"false"})
var Users = mongoose.model('Users',userDetail);
module.exports = Users;
```

System Architecture
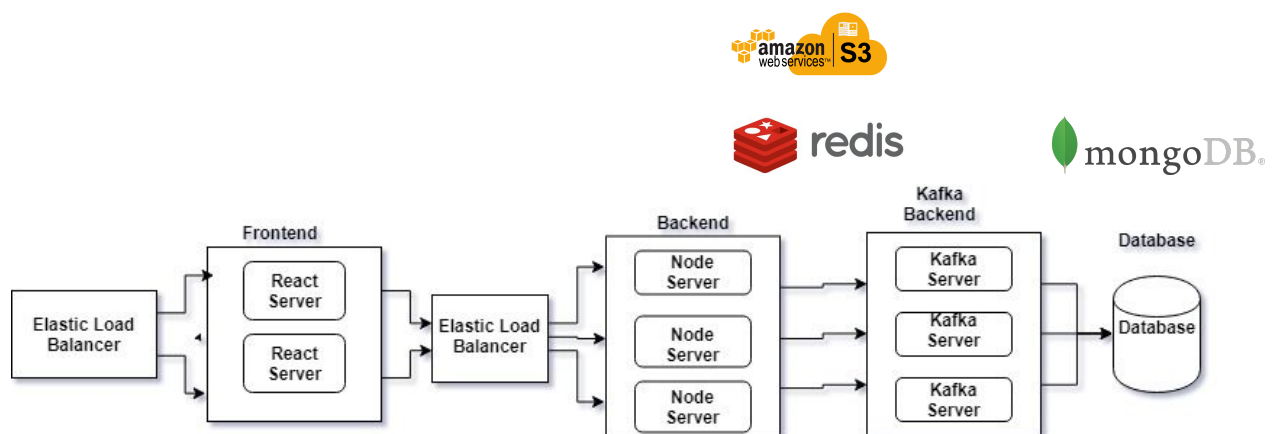
Operational Flow:

• User visits Quora, hosted at AWS and can use the frontend built using both ReactJS and Redux persistent. React is usually created by npm manager. It works by creating virtual DOM and only renders and updates only parts updated by the user.

• When User sends a particular request to backend NodeJS, which works using Event Loop single thread, receives the request using ExpressJS and authenticates the route using PassportJS and then re-route to Kafka Backend server.

• Here Kafka receives the signal and sends it to database MongoDB to validate and receive the data. We have used Mongoose as schema to update and retrieve data from MongoDB.

• And then the Kafka server sends back the date to Node Backend and from there to Frontend and the student can view expected results. Restful Services are used for communication between two systems.

Our Object Management system:

We used react that is best for the states but the states go to the default state whenever the page gets rendered and cannot send the state of one page to other. This can be solved using redux. We also used redux too. In redux, we have a store where we have all the states of various pages by this one page / route can access other pages state variables and use them.

**How to handle "Heavy Weight" Resources?**

The frontend servers are connected using g elastic load balences so the traffic at the frontend will be served, then we are maintaining 4 backend servers which are also connected using elastic load balncer then we have used redis cache for faster access of the most frequenctly accessed data and images and profile pics are stored in s3 bucket. We Have used atlas MongoDB to perform database sharding.

**The policy that we decided when to write into the database.**

Every user has access to write but only for particular operations and particular situations. For example, a user can write question and the user who wrote the question has acess to edit his question, no other user has access to the edit someone's question. In similarlly someone writes answer can be edited by him, no other can modify his answer. User can edit his personal profile details, no one cannot view some other profile details by this others cannot edit his personal details. This varies from route to route. But overall the person who wrote something can be edited by that person only, no others can edit, change, delete it.

Screenshots of Client Application:

- **Quora Login Page:**
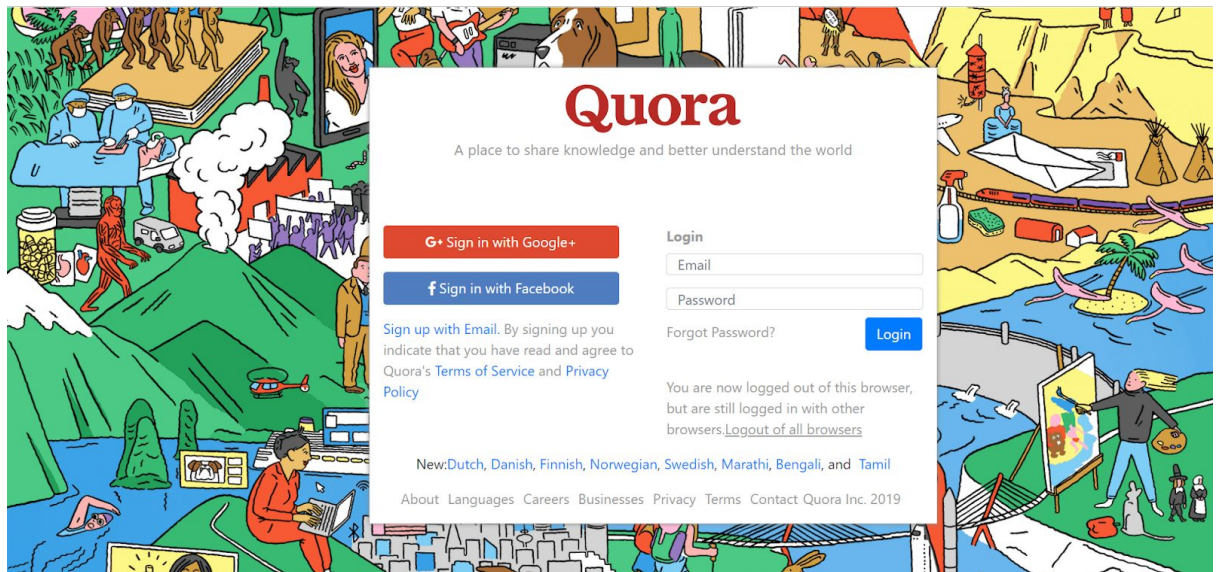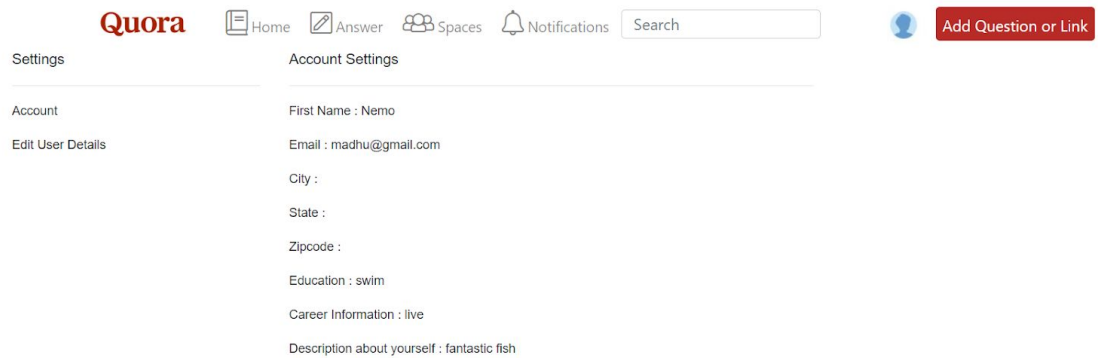
- **Signup Page:**
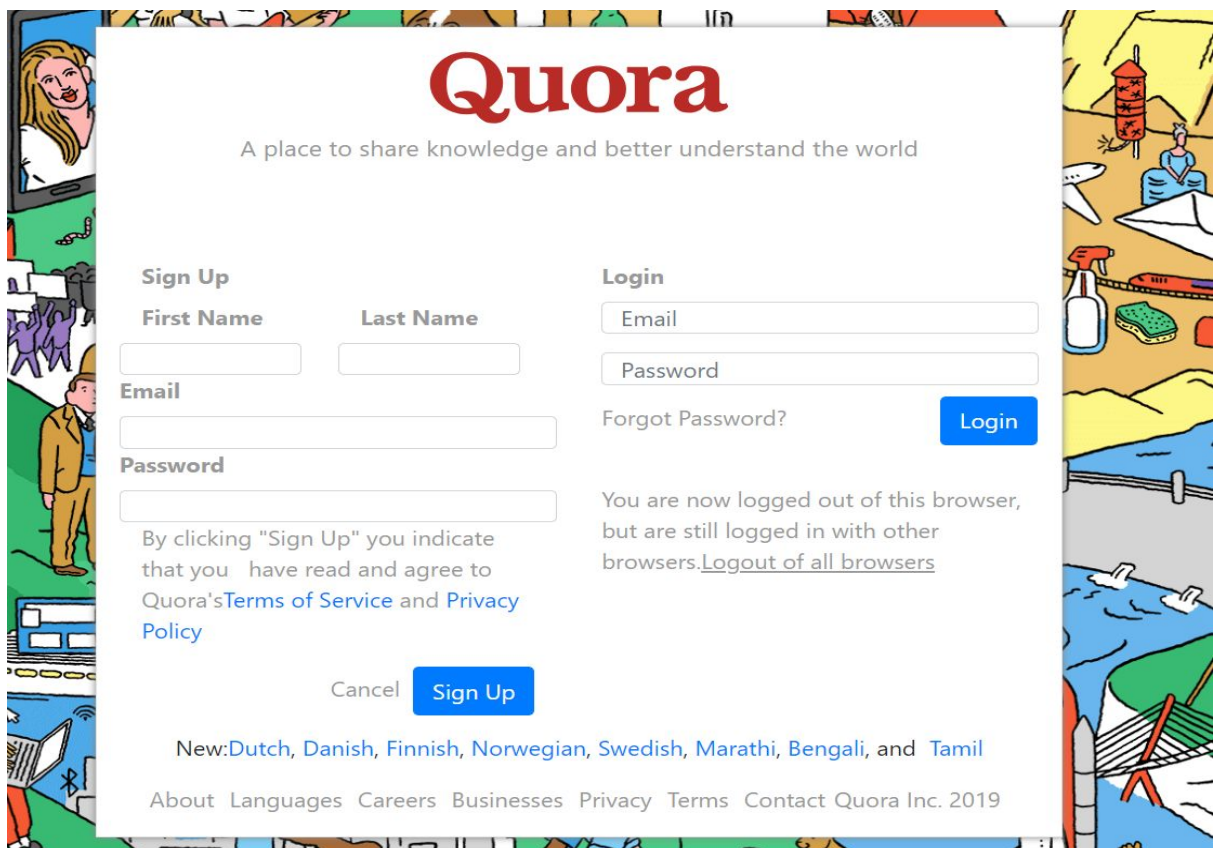
- **Change User Profile information:**
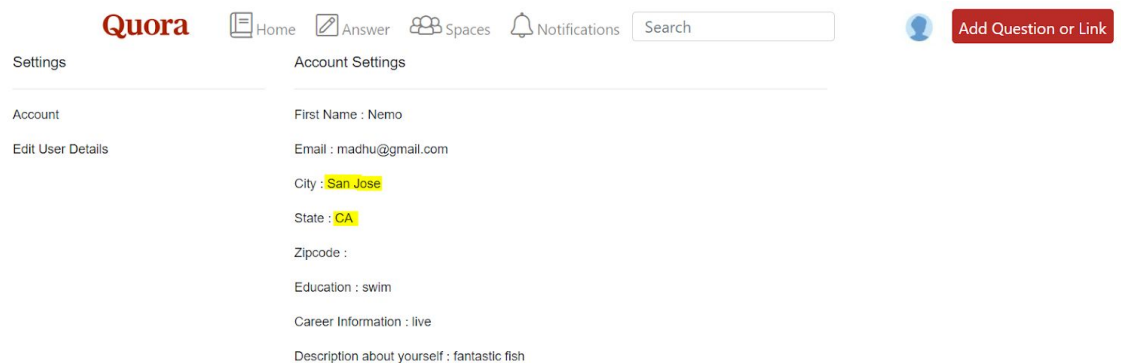
  This is the snapshot before editing the details.

  This is the snapshot after changing details.

- **Search**

  Question and Answer.
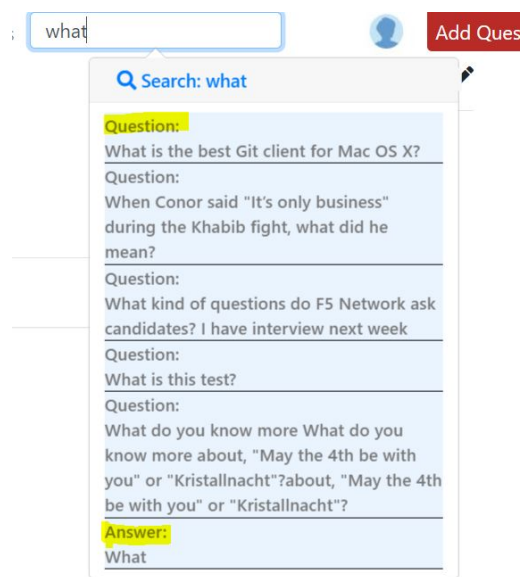
Search for Person.
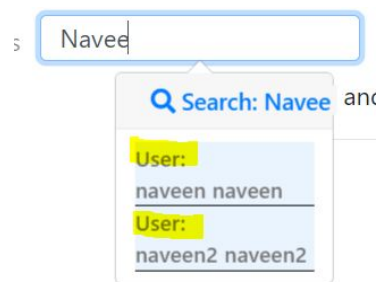
- **Read an Answer:**

- **Bookmark Answers:**

- **Upvote and Downvote Answers:**

    Upvote:

Downvote:

- **Comment a Answer:**

When Conor said "It's only business" during the Khabib fight, what did he mean?

Edit   Follow   Request

Madhusudhan Shagam
Answered 2019-05-06

This is happened

⬆ Upvote 1   ⤳ Share   ⬇ Downvote   Bookmark

Add Comment   Add Comment

Nemo fish
Answered 019-05-07

Not the Relevent answer

● **Answer a Question:**

When Conor said "It's only business" during the Khabib fight, what did he mean?

Edit   Follow   Request

Nemo fish

B   I   ≣   ≣   ⬜   🖼   🔗   •••   ○Anonymous

This Question has a perfect answer

Submit Answer

When Conor said "It's only business" during the Khabib fight, what did he mean?

✎ Edit   ⋙ Follow   ⋔ Request   💬   ⬇

Nemo fish
Answered 2019-05-07

This Question has a perfect answer

- **Messages:**

Conversation with naveen naveen   ✖

                                    Nemo fish
                                          hey

naveen naveen
hi
naveen naveen
Hi

                                    Nemo fish
                                      working

naveen naveen
hi

                                    Nemo fish
                                        hello

Type a message...   Send
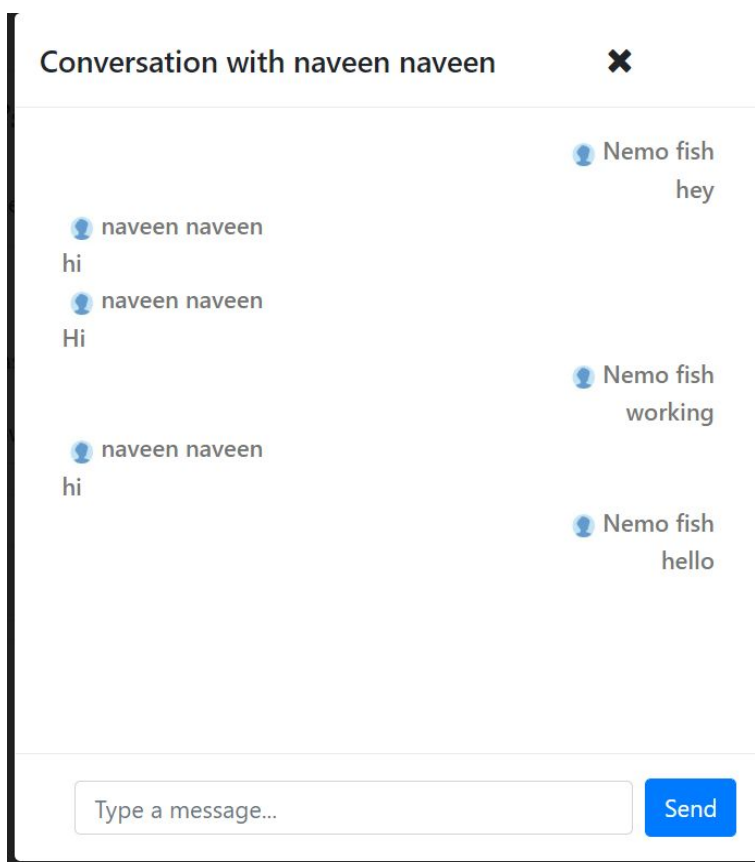
- **Following and Followers:**

Following:

Feeds   Following

Profile
Answers
Questions   naveen naveen
Shares
Spaces
Posts
Blogs
Followers
Following
Edits
Activity

Followers:

Quora   Home   Answer   Spaces   Notifications   Search

naveen naveen

Credential and Highlights

Follow

Feeds   Followers

Profile
Answers
Questions   Madhusudhan Shagam
Shares
Spaces
Posts
Blogs
Followers
Following
Edits
Activity

- **Activity:**

- **Answer anonymously:**

- **Graphs:**

- **Code Listing of Your Client Application:**

  Login

  Register

  Userfeed

  Profile

  Question

  Edit User Details

  Answers

- **Code listing of your server implementations for the entity objects**

  Conversation_list

  Get_conversations

  Send_messages

  Get_notificationss

  View_notifications

  Get_serach_content

  Activity

  Add_answer

  Add_comment_to_answer

  Add_question

  Answers_bookmarked

  Delete_user

  Follow_question

  Follow_topic

  Get_answers

  Get_bookmarked_answer

  Get_feed

  Modifying_details

  Upvotes_downvotes

  User_deactivate

Userprofileupdate

Followers

Following

Get_followers

Get_following

- **code listing of your server implementation for the session object**

Login

- **code listing of your main server code**

Index

Main

Server

- **code listing of your database access class**

MongoDB

- **A code listing of your database creation class**

Answers

Questions

Topics

Users

- **Mocha**

● **Database Schema snapshot**

```
 1    var mongoose = require('mongoose')
 2
 3    const answers = new mongoose.Schema({
 4        question_id:{
 5            type:String
 6        },
 7        answer:{
 8            type:String
 9        },
10        user_id:{
11            type:String
12        },
13        user_id_upvoted:{
14            type:Array
15        },
16        user_name:{
17            type:String
18        },
19        profile_credential:{
20            type:String
21        },
22        owner_status:{
23            type:String
24        },
25        images:{
26            type:Array
27        },
28        is_anonymous:{
29            type:Boolean
30        },
31        upvotes:{
```

```
 1    var mongoose = require('mongoose')
 2
 3    const questions = new mongoose.Schema({
 4    count:{
 5        type:String
 6    },
 7    views:{
 8        type:Array
 9    },
10    question:{
11        type:String,
12        text: true
13    },
14    user_id:{
15        type:String
16    },
17    owner_status:{
18        type:String
19    },
20    topics:{
21        type:Array
22    },
23    followers:{
24        type:Array
```

```
1    var mongoose = require('mongoose')
2
3    const topics = new mongoose.Schema({topic_id:{
4        type:String
5    },
6        topic_name:{
7            type:String
8        },
9        questions:{
10            type:Array
11        }
12        ,
13        users:{
14            type:Array
15        },
16        followers:{
17            type:Array
18        },
19
20    },{strict:"false"})
21    var Topics = mongoose.model('topics',topics);
22    module.exports = Topics;
23
```

```
1    var mongoose = require('mongoose')
2
3    const userDetail = new mongoose.Schema({
4    |
5    email_id:{
6        type:String
7    },
8    password:{
9        type:String
10   },
11   first_name:{
12       type:String
13   },
14   last_name:{
15       type:String
16   },
17   status:{
18       type:String
19   },
20   city:{
21   type:String
22   },
23   state:{
24       type:String
25   },
26   zip_code:{
27   type:String,
28   },
29   profile_image:{
30       type:String
```

- **Observations and lessons learned:**

By the project, we have learnt how to properly handle the code. We reviewed every others code for proper functionality and how to handle the errors generated by the code and our communication effectively to form lasting relationships and contribute to futher projects as well. As per technology-wise, we learned the most by doing this project. This projects helped us to understand how the real-time industry project goes on.

The quora is some website seems to be very simple but implementing this simulation is not so easy. This has may big functionalities that we worked hard to solve them. As a team, we all worked hard and put all our efforts to complete the project on time. We did everythng in a structured form, by this project it was habituated to do everything the structured form.

Initially, we came with some ideas we built the project over it, later on, we need to modify and learn from our mistakes and modified our plan and implemented it in a different way and coded it.