A Major Project Report on

SOCIAL NETWORKING SYSTEM

Submitted by

NARAYANA POOJITHA (R170604)
DERANGULA NANDINI (R171024)
NEELAM RAMA NIKITHA (R171027)

Submitted to

Mr. N. SATHYANANDARAM

HOD CSE

Computer Science and Engineering
Rajiv Gandhi University of Knowledge Technologies
Idupulapaya, Vempalli, Kadapa-516330
Andhra Pradesh, India

as a part of

Partial fulfilment of the degree of Bachelor of Technology in Computer Science and Engineering

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES
(A.P. Government Act 18 of 2008)
IIIT RK VALLEY, RGUKT-AP
Department of Computer Science and Engineering

CERTIFICATE FOR PROJECT COMPLETION

This is certify that the project entitled "SOCIAL NETWORKING SYSTEM " submitted by N. POOJITHA(R170604), D. NANDINI(R171024) and N.RAMA NIKITHA(R171027) , under our guidance and supervision for the partial fulfillment for the degree Bachelor of Technology in Computer Science and Engineering during the academic semester-II 2022-2023 at IIIT ,RK VALLEY RGUKT-AP. To the best of my knowledge, the result embodied in this dissertation work have not been submitted to any University or Institute for the award of any degree or diploma.

Project Guide	**Head of the Department**
Mr.N. CHANDRA SEKHAR	Mr.N. SATHYANANDARAM
Assistant Professor	HOD Of CSE
IIIT,RGUKT-AP,RK Valley	IIIT,RGUKT-AP,RK Valley

Acknowledgement

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success.

I am grateful to our respected Director, **Prof. K. SANDHYA RANI** for fostering an excellent academic climate in our institution.

I also express my sincere gratitude to our respected Head of the Department **Mr. N.SATHYANANDARAM** for his encouragement, overall guidance in viewing this project a good asset and effort in bringing out this project.

I would like to convey thanks to our guide **Mr. N. CHANDRASHEKAR** for his guidance, encouragement, co-operation and kindness during the entire duration of the course and academics.

My sincere thanks to all the members who helped me directly and indirectly in the completion of project work. I express my profound gratitude to all ourFriends and family members for their encouragement.

ABSTRACT

A social networking application that focuses on question-and-answer interactions is a unique platform that enables users to connect with others based on their shared interests and knowledge. This type of application allows users to ask questions, provide answers, and engage with others who share similar interests.

The application provides a user-friendly interface, personalized feeds, and notification alerts, allowing users to stay up-to-date with the latest content and interact with others easily. Users can also create profiles and follow other users, providing a social aspect to the platform.

This type of social networking application serves as an excellent resource for individuals seeking advice or knowledge on various topics, allowing them to connect with a diverse community of experts and enthusiasts alike.

The application's community-driven approach also encourages collaboration and engagement, making it an invaluable tool for personal and professional development.

TABLE OF CONTENT:

Introduction:

The following sections provide the complete description about Software Requirement Specification Document.

Purpose:

The Software Requirements Specification will provide the detailed description of the requirements for the Online Medical Store Automation System. The SRS will give complete understanding the project and its functionality.

The SRS document describes the functional and non-functional requirements for the Online Medical Store Automation Software. Through this developers will know the correct software to be developed for the end user.The SRS is the foundation of the project.

The SRS can be used by Software Enginners to fully understand the requirements of customer or end user. And end users can use this SRS for testing, so that they check weather the developers meet the customer requirements. If the software didn’t meet their requirements they can specify on which area the error is.

Scope:

A social networking application that centers around question-and-answer interactions has significant potential for various uses and audiences. The application can be used to facilitate the sharing of knowledge on various topics, allowing users to ask and answer questions and share their expertise. The application can serve as a networking platform for individuals in specific industries, allowing them to connect with others in their field and expand their professional network.

The application can be used to bring together people who share common interests and facilitate the formation of communities around these interests. The application can be used as a tool for educational purposes, providing a platform for students and educators to connect and share knowledge. The application can be used as a marketing tool for businesses and brands to engage with their audience and gather feedback on products and services.

Intended Audience:

The intended audience are online users.

Overall Description:

Describes the general factors that affect the product and its requirements.

Product perspective:

Social Networking system is a software, which is being developed to answer questions asked by users.

Product Features:

User Registration:

Any user who want to buy the medicines can register with their details like FirstName, LastName, Email, Profile. Then they can login with the email and password.

User login:

User can login he/she should enter user name and password. The website provides login facility to the users.

Search Topics:

User can search the topics based on their required topics. The website provides any type of topics. For Example: Database Design, Programming, Java, JavaScript etc.

User Characteristics:

User:

Users can set their user name and password and user can search the topics or they ask Question and any User can Answer for that Questions.

Operating Environment:

- Operating System : Ubuntu or Windows
- Technologies used :
  Front-End: JavaScript, CSS, Handlebars.
  Back-End: Nodejs, Mongodb.

REQUIREMENT SPECIFICATION:

This section contains all the software requirements that when combined with the system context diagram, use cases, and use case descriptions, is sufficient to enable designers to design a system to satisfy requirements, and testers to test that the system satisfies those requirements.

Hardware requirement:

The Software will run on all basic configuration systems.

Software requirement:

Front End	JavaScript, CSS, Handlebars
Database Server	Mongodb
Web Browser	Chrome, FireFox
Operating System	Windows, Ubuntu
Web Server	Apache

APACHE:

- The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows.
- The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards.
- The Apache HTTP Server was launched in 1995 and it has been the most popular web server on the Internet since April 1996.

Handlebars:

- Handlebars is a template engine for JavaScript that allows developers to create dynamic HTML templates.

- Handlebars uses a simple syntax with double curly braces ({{}}) to denote variables and built-in helpers for common tasks such as iteration and conditional rendering.

- Handlebars can compile templates into JavaScript functions that can be executed at runtime to generate dynamic content.

Mongodb:

- MongoDB is a popular NoSQL database that stores data in flexible, document-like structures called BSON.

- ➓ BSON is a binary representation of JSON, which allows for more efficient data storage and retrieval.

- ➓ MongoDB supports a variety of data models, including key-value, document, and graph, making it a versatile tool for many different types of applications.

- ➓ MongoDB is designed to scale horizontally, which means it can distribute data across multiple servers to handle large volumes of traffic and data.

- ➓ MongoDB has a rich query language that supports complex queries, including aggregation and geo-spatial queries

- ➓ MongoDB supports ACID transactions, which ensure that database transactions are reliable and consistent.

External Interface Requirements:-

- ➓ User Interface:
  - ➓ Using Social Networking System website
    - ➓ User can register
    - ➓ User can login
    - ➓ User can logout

- ➓ Hardware Interface: No hardware interfaces

- ➓ Software Interface:
  - ➓ Operating system : Windows, Ubuntu

- ➓ Technologies used : JavaScript, CSS, Mongodb , Handlebars, Nodejs
- ➓ Communication :
  - ➓ If any User ask Questions then remaining user can answer for that Question like that the communication will be increase in the Social Networking system.

Functional Requirements:

Functional requirements define the fundamental actions that system must perform. The functional requirements for the Social Networking System are User registration, Asking Questions, Answering Questions.

- User:
  - ➓ All users can create an account that is used to clarify their doubts.
  - ➓ User registration and login shall be mandatory.
- Create an Account:
  - ➓ The system should provide the user with an easy to use GUI to facilitate their creating

an account.

  - ➓ The system shall ask for an email address and password.
  - ➓ The system shall notify the user if incorrect characters are used in the email or password fields.
  - ➓ The system should notify the user if their email has already been used.
  - ➓ The system should notify the user if any required fields are left empty.
  - ➓ The system should explain how the submitted password is unsecure.
  - ➓ The system should prompt the user for their email address and password while login.
- Ask Questions:
  - ➓ The user can ask the Question to clarify their doubts or to gain some informations
  - ➓ Only login users can ask questions.
- Answer Questions:
  - ➓ Any user can give the answers to another users clarify their required information
  - ➓ Only login users can suggest their answers here.

Non-Functional Requirements:-

- ➓ Performance:
  - ➓ High level of performance is required as n number of people can use it at the same time.Anyone who is accessing this application should get the data required quickly from the database based on their request.
  - ➓ The load time for user interface screens shall take no longer than ten seconds.
  - ➓ The log in information shall be verified within five seconds.
- ➓ Reliability:
  - ➓ There should be no errors while user was using or this website, all operations should run smoothly for better performance. User should get appropriate information related to the required topics or any doubts.
- ➓ Security:
  - ➓ This Social Networking System software provide more secured answers to save confidential data it consists of sensitive information related to their knowledge.
- ➓ Availability:
  - ➓ This Social Networking System software should provide 24/7 access to the users.
- ➓ Platform Compatibility:
  - ➓ This software should work on any kind of browser as it is a web application.
- ➓ Maintainability:
  - ➓ The Social Networking System is a website. It shall be easy to maintain.
- ➓ Portability
  - ➓ The Social Networking System shall run in any Linux or Windows environment that contains database

DESIGN INTRODUCTION:

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization. Once the software was requirements have been analyzed and specified the software design involves FOUR technical activities - design, coding, implementation and testing that are required to build and verify the software.

Design is the only way to accurately translate the Users requirements into finished software or a system. Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

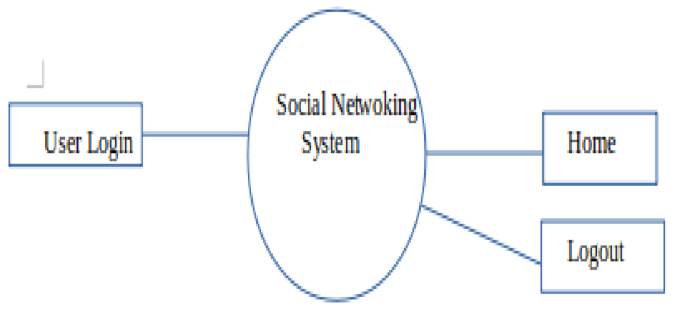
UML Diagrams:

UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

1. CONTEXT DIAGRAM:

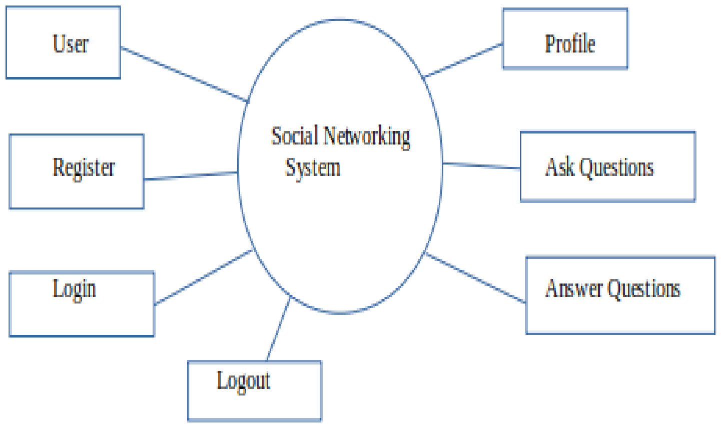
DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities.

DFD LEVEL 0:



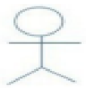
DFD LEVEL-1:

DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its sub processes. And the databases used are also represented.

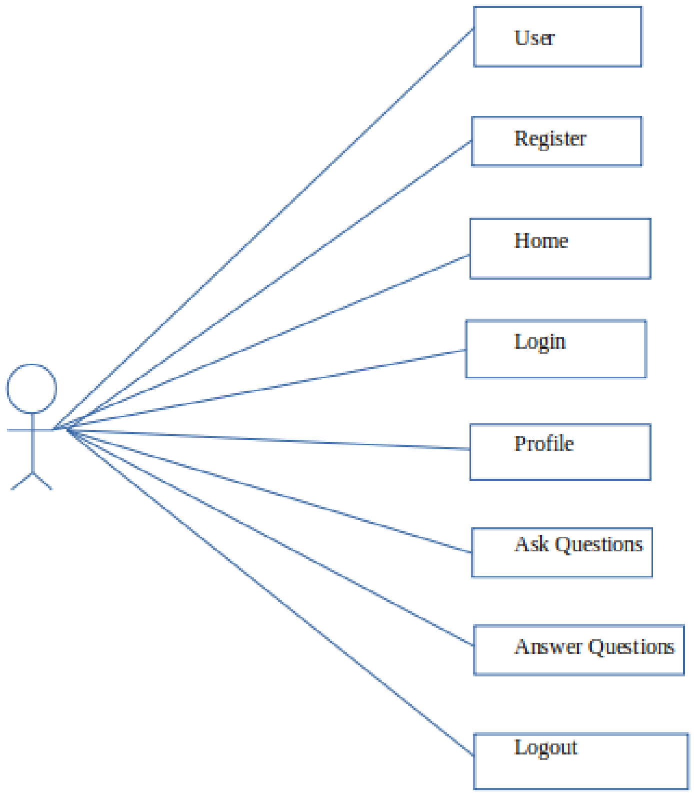


2. USECASE DIAGRAM:

Actor: A coherent set of roles that users of use cases play when interacting with the use cases an observable result of value of an actor.



Action: A description of sequence of actions, including variants that a system performs yields an observable result of value of an actor. Diagram is drawn in an eclipse shape.

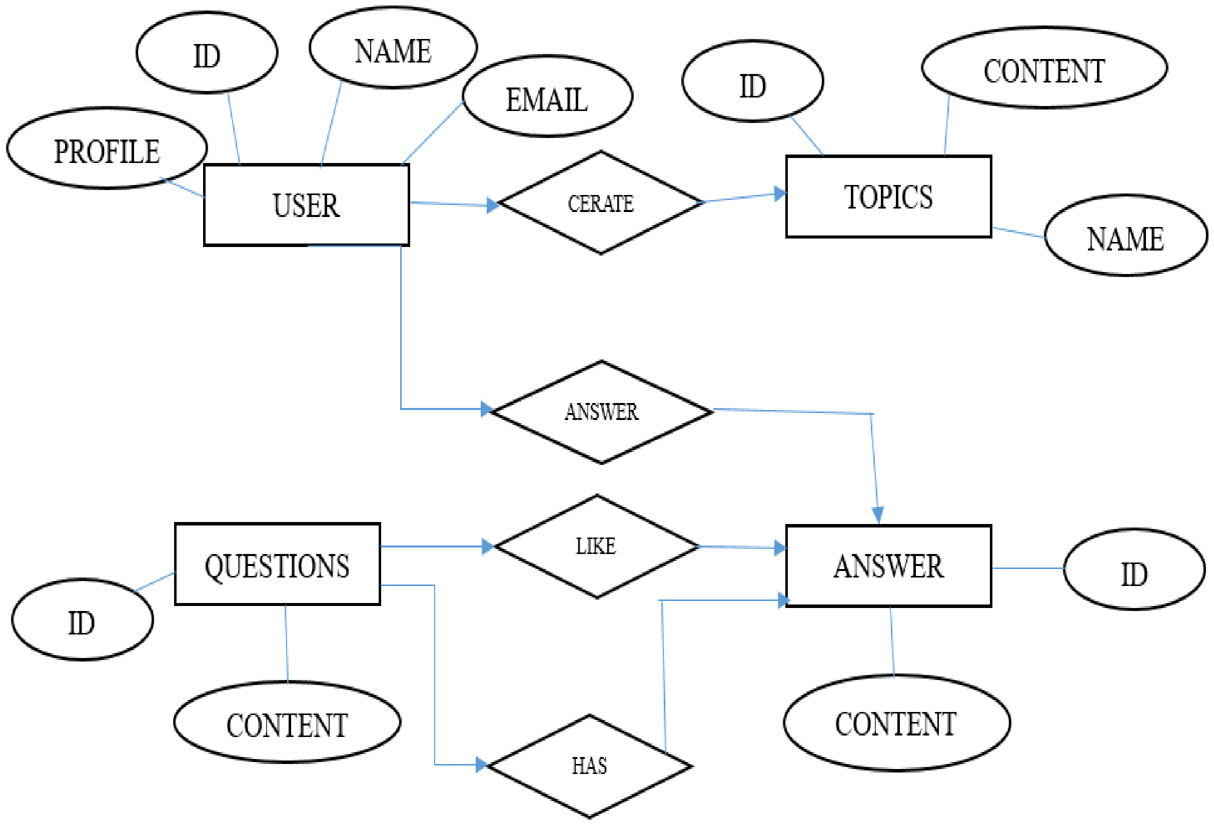


3. ER Diagram:

ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. It maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables. In addition, the model can be used as a design plan by the database developer to implement a data model in specific database management software.

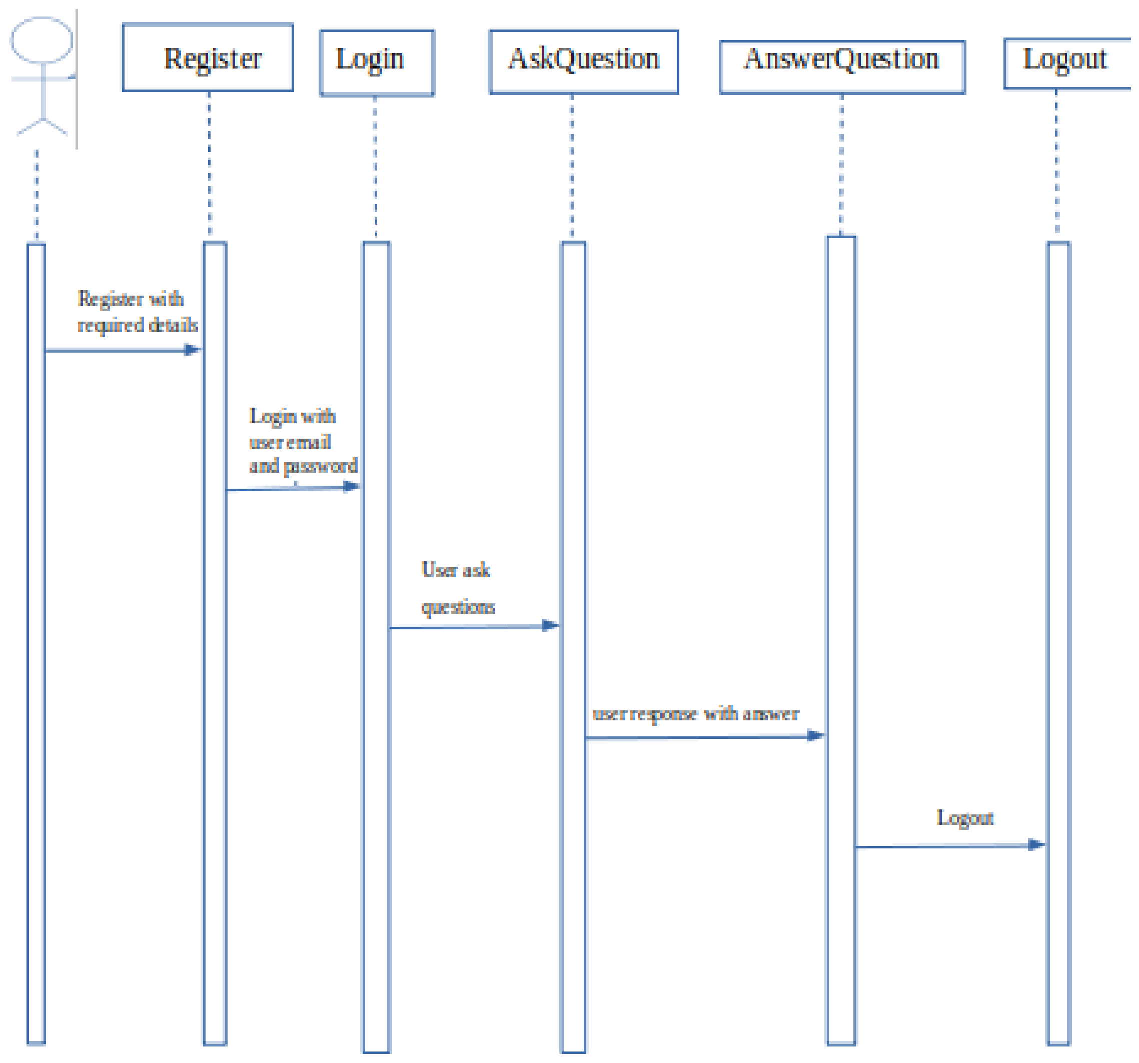
ER Notation: There is no standard for representing data objects in ER diagrams. The original notation used by Chen is widely used in academics texts and journals but rarely seen in either CASE tools or publications by nonacademics. Today, there are a number of notations used; among the more common are Bachman, crow's foot, and IDEFIX. All notational styles represent entities as Rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the cardinality of a connection.

Entities are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.
Relationships are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs 15 Attributes, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
Cardinality of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one. Existence is represented by placing a circle or a perpendicular bar on the line.



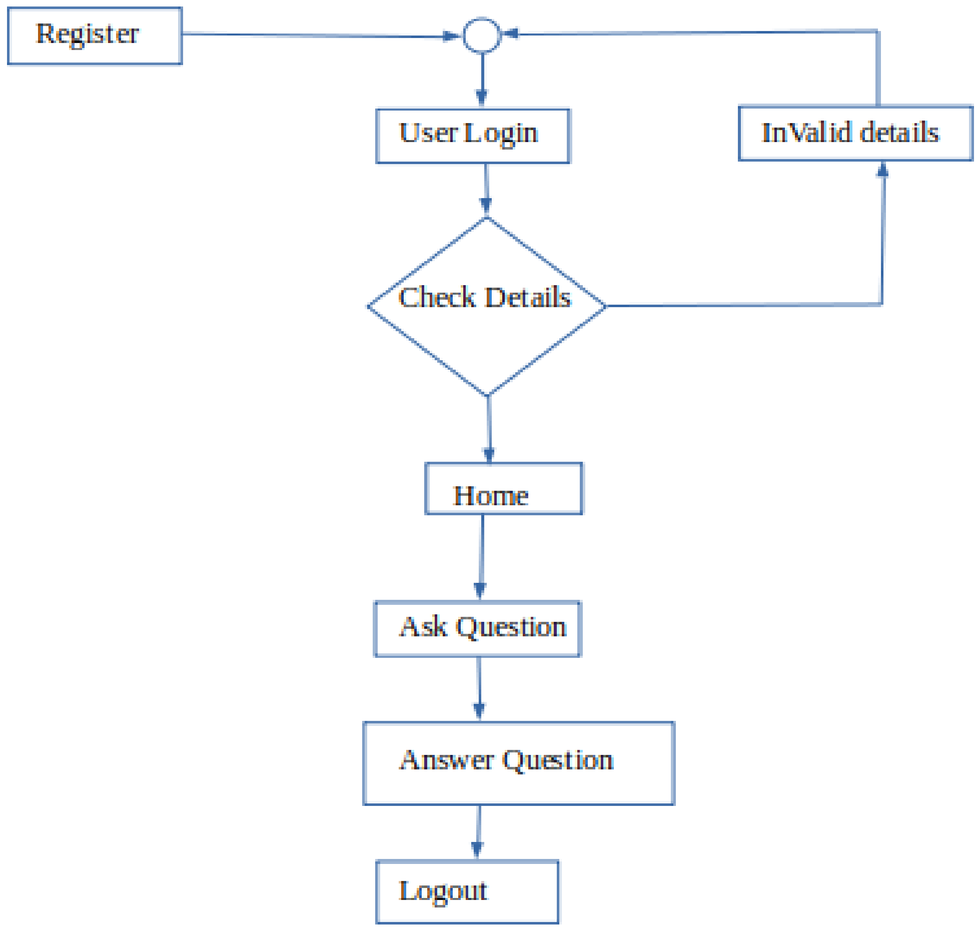
4. SEQUENCE DIAGRAM:

The purpose of sequence diagram is to show the flow of functionality through a usecase. In other words, we call it mapping process in terms of data transfers from the actor through the corresponding objects.


Register
Login
AskQuestion
AnswerQuestion
Logout
Register with required details
Login with user email and password
User ask questions
user response with answer
Logout

5. ACTIVITY DIAGRAM:

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc. Start of a process mentioned with filled circle and activities are represented with rectangles, Condition checks are represented with rhombus and the termination is represented with double lined filled circle.



CODING:

The next process after the Design is Coding. Developers after getting the complete Design document they start Coding as per requirement. The coding phase includes system design in an integrated development environment. It also includes static code analysis and code review for multiple types of devices.

FrontEnd:

Register.hbs:

```
<div class="container container-forum">
<div class="row">
<div class="col-md-7 mx-auto" style="margin-top: 50px;">
<div class="card">
<div class="card-header login-register-card-header" style=" background-color:
#57b846;">Register </div>
<div class="card-body">
{{#if successMessage}}
<div class="alert alert-success text-center">
{{successMessage}} <br>
<a href="/users/login"> Login </a></div>
{{/if}}
{{#if errors}}
{{#each errors}}
<div class="alert alert-danger text-center">
{{this.msg}}
</div>
{{/each}}
{{/if}}
<form action="/users/register" method="POST" enctype="multipart/form-data" class="mx-auto"
style="dipslay:inline-block;width:400px">
<div class="form-group mb-4">
<input type="text" name="firstName" placeholder="firstName" class="form-control login-register-
input" required></div>
<div class="form-group mb-4">
<input type="text" name="lastName" placeholder="lastname" class="form-control login-register-input"
required></div>
<div class="form-group mb-4">
<input type="email" name="email" placeholder="email" class="form-control login-register-input"
required></div>
<div class="form-group mb-4">
<input type="password" name="password" placeholder="password" class="form-control login-register-
input"></div>
<div class="custom-file mb-4">
<input type="file" name="image" class="custom-file-input" id="validatedCustomFile" required>
```

```
<label class="custom-file-label" for="validatedCustomFile">choose profile picture</label>
<button class="btn btn-block text-capitalize mt-4 login-register-btn-submit"
name="registration_submit" style="background-color: #57b846;"> <i class="fas fa-user-plus"></i> sign
in</button></form>
<p class="login-register-have-account-link">have an account?</p>
<p style="text-align: center;"><a href="/users/login" class="login-register-bottom-link" style="color:
#57b846;">log in now</a></p>
</div></div></div></div></div>
```

Login.hbs:

```
<div class="container container-forum">
<div class="row">
<div class="col-md-7 mx-auto" style="margin-top: 50px;">
<div class="card" >
<div class="card-header login-register-card-header" style="background-color: #007bff;">login</div>
<div class="card-body">
{{#if messages.error}}
<div class="alert alert-danger text-center">
{{messages.error}}</div>
{{/if}}
<form action="/users/login" method="POST" class="mx-auto" style="dipslay:inline-
block;width:400px">
<div class="form-group mb-4">
<input type="email" name="email" placeholder="email" rquired class="form-control login-register-
input"></div>
<div class="form-group mb-4">
<input type="password" name="password" required placeholder="password" class="form-control login-
register-input"></div>
<button class="btn btn-block text-capitalize mt-4 login-register-btn-submit" style="background-color:
#007bff;"> <i class="fas fa-sign-in-alt" type="submit"></i> log in</button></form>
<p class="login-register-have-account-link">dont have an account?</p>
<p style="text-align: center;"><a href="/users/register" class="login-register-bottom-link" style="color:
#0d4c87;"> register now </a></p></div></div></div></div></div>
```

Ask_question.hbs:

```
<div class="container container-forum">
<div class="row">
<div class="col-md-10 mx-auto" style="margin-top: 50px;">
{{#if topic}}
<form action="/topics/{{topic._id}}/question" method="POST" class="form-border">
<h3 class="text-center"><a href="/topics/{{topic._id}}" class="text-capitalize text-primary text-
decoration-none"> {{topic.name}} </a> </h3>
{{#if successMessage}}
```

```
<div class="alert alert-success text-center">
{{successMessage}}
</div>
{{/if}}
{{#if errors}}
{{#each errors}}
<div class="alert alert-danger text-center">
{{this.msg}}
</div>
{{/each}}
{{/if}}
<div class="form-group mb-4">
<label for="descritpion">Your question</label>
<textarea class="form-control" name="question" id="description" placeholder="question"
rows="3"></textarea> </div>
<button class="btn btn-block text-capitalize" style="background-color: #324f6c; color: white; font-
weight: 300; font-size: larger;" > ask</button></form>
{{else}}
<div class="alert alert-warning">
<h1 class="text-capitalize text-center">topic does not exist!</h1>
</div>
{{/if}}</div></div></div>
```

Write_answers.hbs:

```
<script src="/js/tinymce/tinymce.min.js"></script>
<script>
tinymce.init({
selector: '#answer',
});
</script>
<div class="container container-forum">
<div class="row">
<div class="col-md-10 mx-auto" style="margin-top: 50px;">
{{#if question}}
<form action="/questions/{{question._id}}/answer" method="POST" class="form-border">
<h3 class="text-center"> <a href="/questions/{{question._id}}" class="text-primary text-capitalize text-
decoration-none"> {{question.content}} </a> </h3>
{{#if successMessage}}
<div class="alert alert-success text-center">
{{successMessage}}
</div>
{{/if}}
<div class="form-group mb-4">
<label for="answer">Your answer</label>
```

```
<textarea class="form-control" name="answer" id="answer" placeholder="answer"
rows="15"></textarea><div>
<button class="btn btn-block text-capitalize"style="background-color: #324f6c; color: white; font-
weight: 300; font-size: larger;"> answer</button></form>
{{else}}
<div class="alert alert-warning">
<h1 class="text-capitalize text-center">question does not exist!</h1>
</div></div></div></div>
```

Profile.hbs:

```
<div class="container container-forum bg-white">
    {{#if user}}
        <div class="row justify-content-center" style="padding-top: 10px;">
            <div class="col-lg-4 col-md-4 col-sm-4 col-xs-12">
<img src="/images/users-images/{{user.imageName}}" class="img-fluid mx-auto d-block rounded">
</div>
<div class="col-lg-8 col-md-8 col-sm-4 col-xs-12">
<h2 class="text-capitalize font-weight-bold font-italic"> {{user.firstName}} {{user.lastName}} </h2>
<h5>registred on {{user.registrationDate}} </h5>    </div></div><hr>
<div class="row">
<div class="col-lg-4 col-md-4">
<div class="row">
<div class="col-lg-10 col-md-10">
<table class="table">
<thead>
<tr><th class="h4 text-capitalize">flux</th></tr></thead>
<tbody>
<tr><td>
<a href="/users/{{user._id}}/questions" class="text-muted text-capitalize"> {{user.questions.length}}
question(<span class="text-lowercase">s</span>)</a></td></tr>
<tr><td>
<a href="/users/{{user._id}}/answers" class="text-muted text-capitalize"> {{user.answers.length}}
answer(<span class="text-lowercase">s</span>)</a></td></tr>
<tr><td>
<a href="/users/{{user._id}}/topics" class="text-muted text-capitalize"> {{user.topics.length}}
topic(<span class="text-lowercase">s</span>) created</a></td></tr>
<tr><td>
<a href="/users/{{user._id}}/topics-followed" class="text-muted text-capitalize">
{{user.topicsFollowed.length}} topic(<span class="text-lowercase">s</span>) followed</a></td></tr>
</tbody></table> </div></div></div>
<div class="col-lg-8 col-md-8">
<div class="row">
<div class="col-lg-10 col-md-10">
```

```
{{#if questions}}
{{>profile-includes/user-questions}}
{{/if}}
{{#if answers}}
{{>profile-includes/user-answers}}
{{/if}}
{{#if topics}}
{{>profile-includes/user-topics-created}}
{{/if}}
{{#if topicsFollowed}}
{{>profile-includes/user-topics-followed}}
{{/if}}</div></div></div></div>{{else}}
<div class="row">
<div class="col-sm-12 alert alert-warning mb-0" >
<h1 class="">User Does Not Exist!</h1></div></div>
{{/if}}</div>
```

BackEnd:
AnswersController.js :

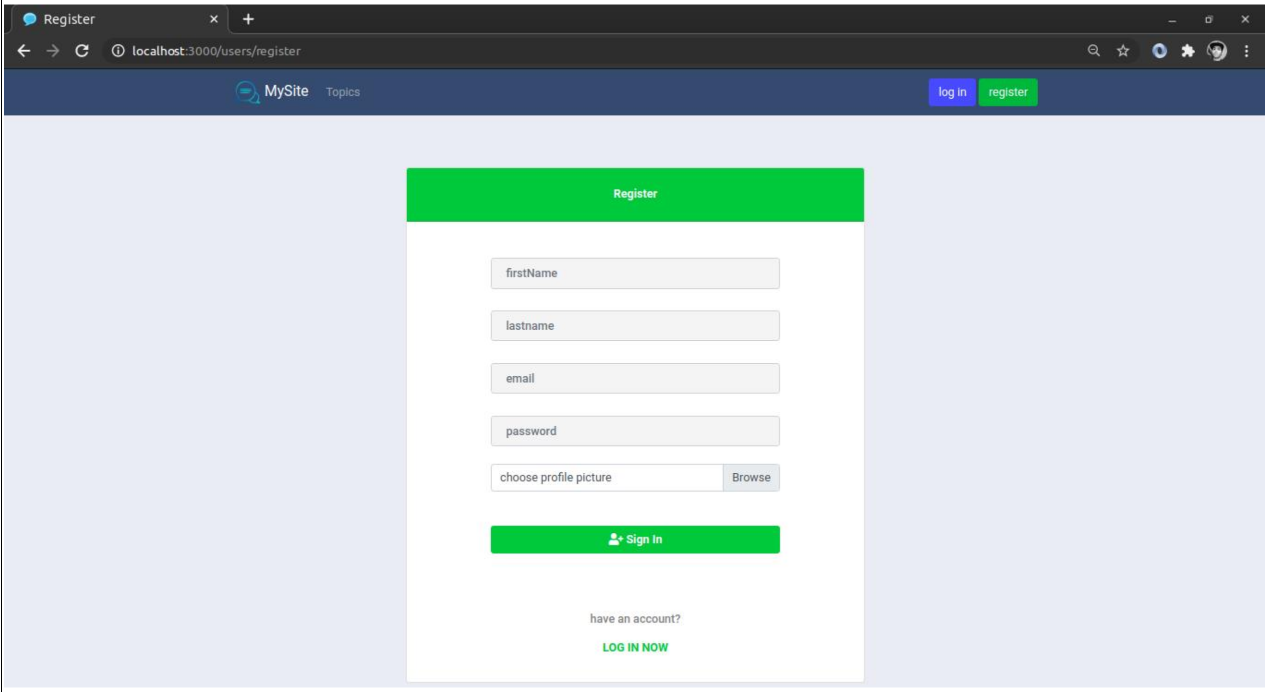
```
const Answer = require("./../models/answer")
const User = require("./../models/user")
const answer = require("./../models/answer")
exports.handleLikingUnliking = async (req, res, next) => {
try {
const userId = req.user._id
const { answerId } = req.body
console.log(userId)
console.log(answerId)
const answer = await Answer.findById(answerId)
const user = await User.findById(userId)
if (user && answer) {
const isLiking = answer.usersWhoLike.includes(userId) && user.answersLiked.includes(answerId)
let updatedUser = null
let updatedAnswer = null
if (isLiking) {
updatedUser = await User.findByIdAndUpdate(
userId,{
$pull: { answersLiked: answerId }
},
{
new: true,
useFindAndModify: false
})
updatedAnswer = await Answer.findByIdAndUpdate(answerId,
{
$pull: { usersWhoLike: userId }
},
{
new: true,
useFindAndModify: false
})
} else {
updatedUser = await User.findByIdAndUpdate(
userId,
{
$push: { answersLiked: answerId }
},
{
new: true,
useFindAndModify: false
})
            updatedAnswer = await Answer.findByIdAndUpdate(
answerId,
{
$push: { usersWhoLike: userId }
},
{
new: true,
useFindAndModify: false
} ) }
res.json({ isLiking: !isLiking, answer: updatedAnswer })}
} catch (error) {next(error)
}}
```

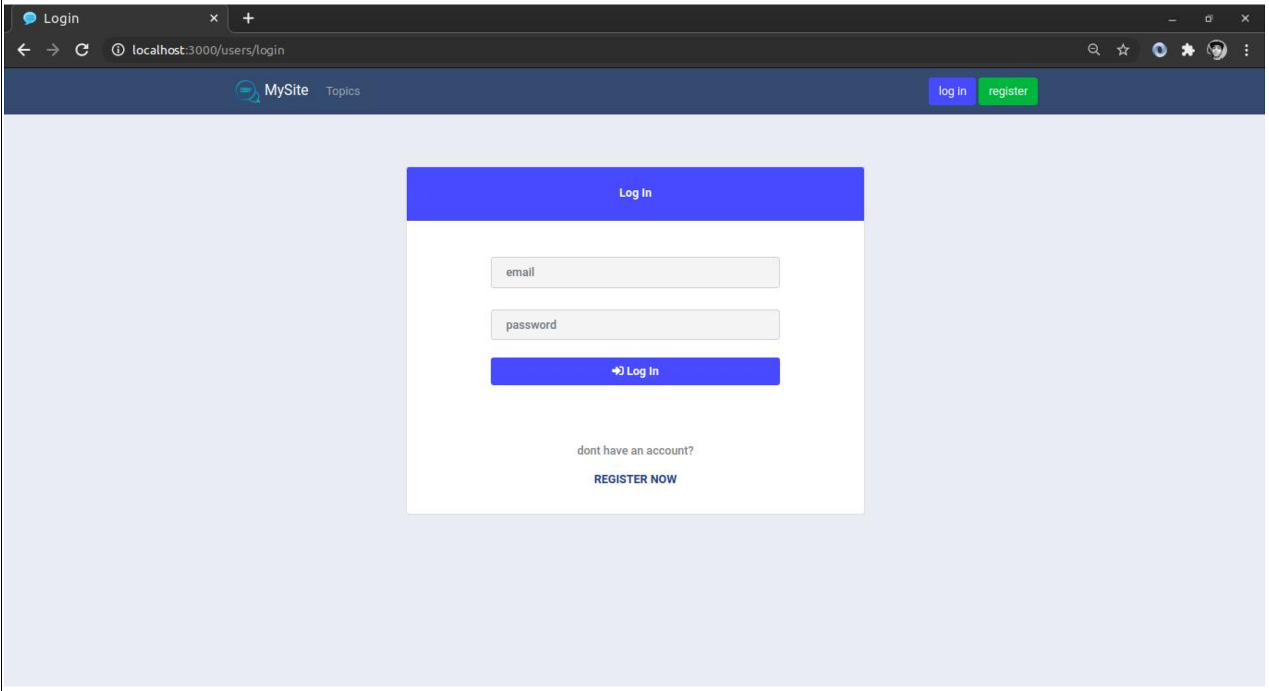
QuestionsController.js :

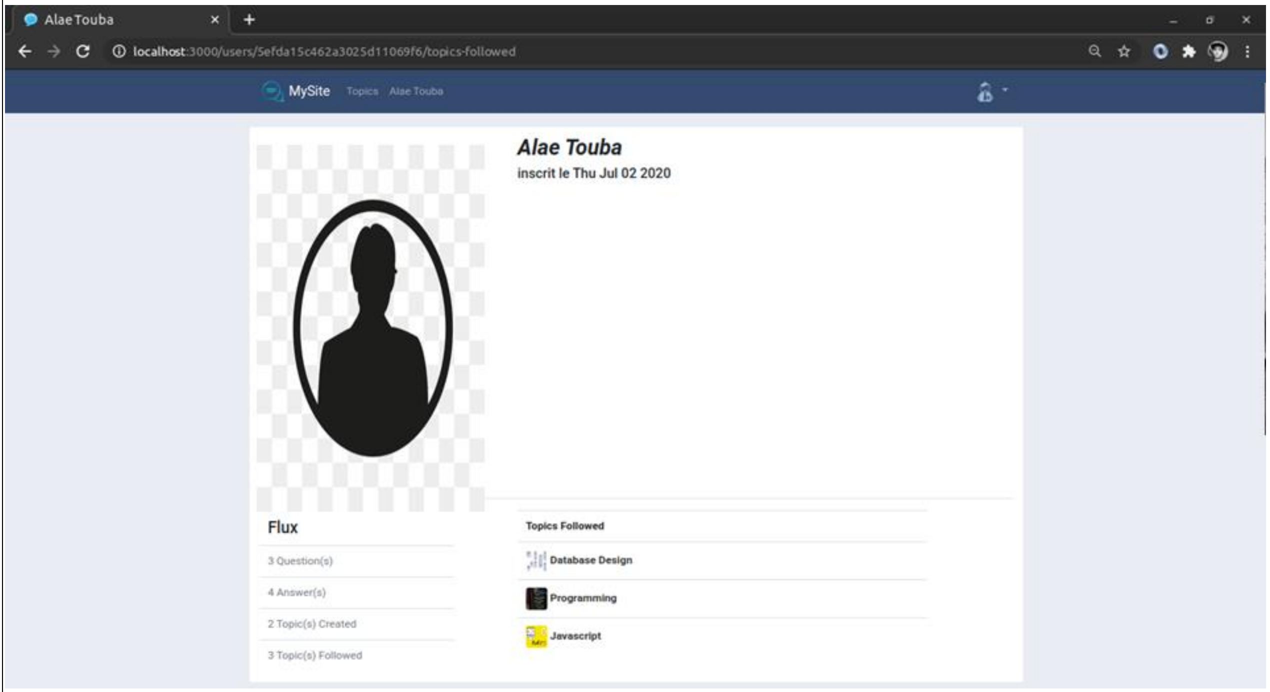
```
const Question = require("./../models/question")
const Topic = require("../models/topic")
const User = require("./../models/user")
const Answer = require("./../models/answer")
exports.getQuestionById = async (req, res, next) => {
const input = {
title: "question not found",
connectedUser: req.user
}
try {
const question = await Question.findById(req.params.id).populate("user").populate({
path: "topic",
populate: {
```

```
path: "questions",
match: { _id: { $ne: req.params.id } }
}})
.populate({
path: "answers",populate: { path: "user" }})
if (question) {
input.title = question.content
question.creationDate = new Date(parseInt(question.creationDate)).toDateString()
input.question = question
input.answersIds = question.answers.map((answer) => answer._id)
}
res.render("question", input)
} catch (error) {
if (error.message && error.message.incldues("Cast to ObjectId failed for value")) {
res.render("question", input)
} else {
next(error)
}}}
exports.getCreateAnswerPage = async (req, res, next) => {
const input = {
title: "answer a question",
connectedUser: req.user
}
try {
const question = await Question.findById(req.params.id)
if (question) {
input.question = question
}
res.render("write-answer", input)
} catch (error) {
if (error.message && error.message.includes("Cast to ObjectId failed for value")) {
res.render("write-answer", input)
} else {
next(error)
}}}
exports.handleCreateAnswer = async (req, res, next) => {
console.log(req.body)
const input = {
title: "write an answer",
connectedUser: req.user
}
try {
const { answer } = req.body
const connectedUser = req.user
const questionId = req.params.id
const question = await Question.findById(questionId)
if (question) {
const newAnswer = await Answer.create({
content: answer,
user: connectedUser._id,
question: questionId})
const updatedUser = await User.findByIdAndUpdate(
connectedUser._id,
{
$push: { answers: newAnswer._id }
},
{
new: true,
useFindAndModify: false
})
const updatedQuestion = await Question.findByIdAndUpdate(
questionId,
{
$push: { answers: newAnswer._id }
},
{ new: true,
useFindAndModify: false
})
input.question = question
input.successMessage = "answer successfull created"}
res.render("write-answer", input)
} catch (error) {
if (error.message && error.message.includes("Cast to ObjectId failed for value")) {
res.render("write-answer", input)
} else {
next(error)}}}
```

OUT PUT:


Register
localhost:3000/users/register
MySite
Topics
log in
register
Register
firstName
lastname
email
password
choose profile picture
Browse
Sign In
have an account?
LOG IN NOW


Login
localhost:3000/users/login
MySite
Topics
log in
register
Log In
email
password
Log In
dont have an account?
REGISTER NOW



Alae Touba

localhost:3000/users/5efda15c462a3025d11069f6/topics-followed

Alae Touba

inscrit le Thu Jul 02 2020

Flux

3 Question(s)

4 Answer(s)

2 Topic(s) Created

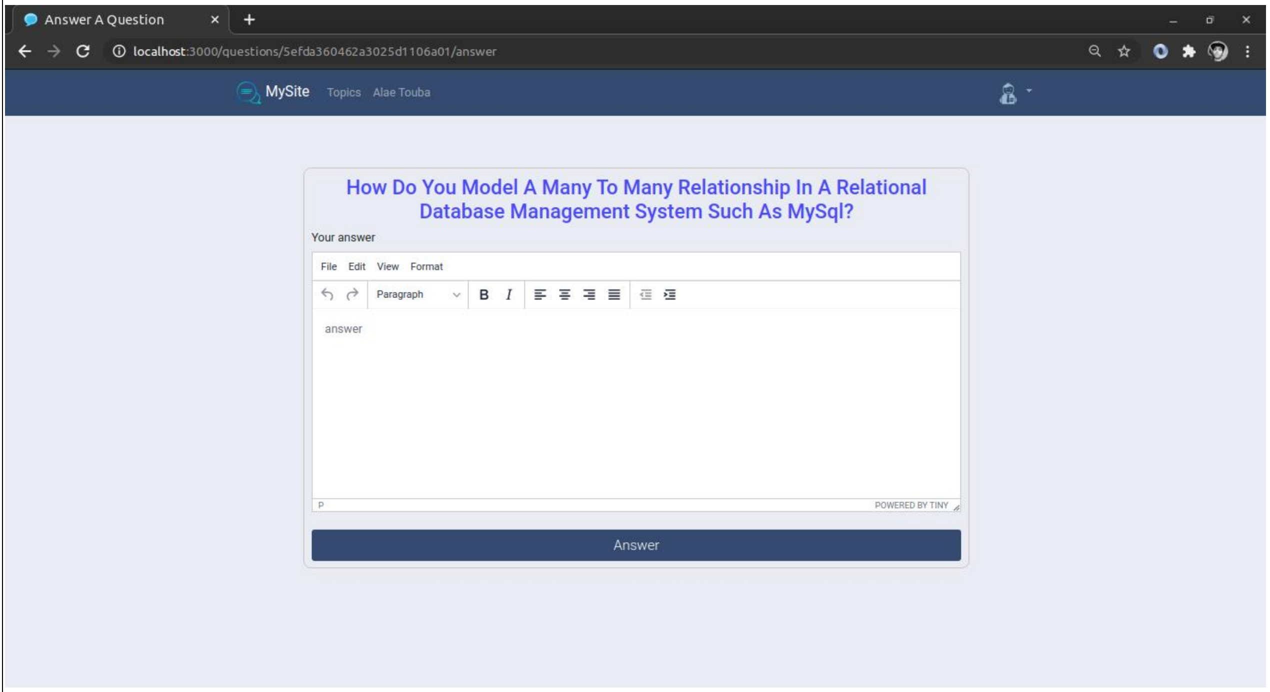
3 Topic(s) Followed

Topics Followed

Database Design

Programming

Javascript



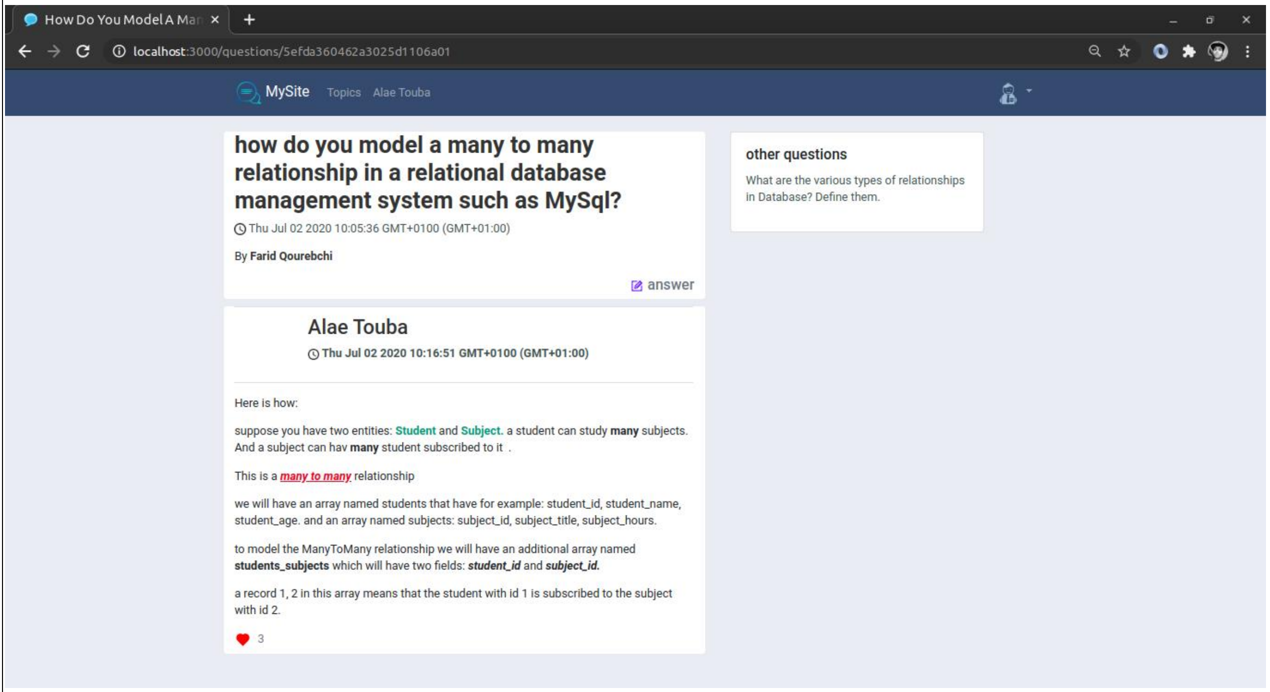
Answer A Question

localhost:3000/questions/5efda360462a3025d1106a01/answer

How Do You Model A Many To Many Relationship In A Relational Database Management System Such As MySql?

Your answer

answer


How Do You Model A Man

localhost:3000/questions/5efda360462a3025d1106a01

MySite Topics Alae Touba

how do you model a many to many relationship in a relational database management system such as MySql?

Thu Jul 02 2020 10:05:36 GMT+0100 (GMT+01:00)

By **Farid Qourebchi**

answer

other questions

What are the various types of relationships in Database? Define them.

Alae Touba

Thu Jul 02 2020 10:16:51 GMT+0100 (GMT+01:00)

Here is how:

suppose you have two entities: **Student** and **Subject.** a student can study **many** subjects. And a subject can hav **many** student subscribed to it .

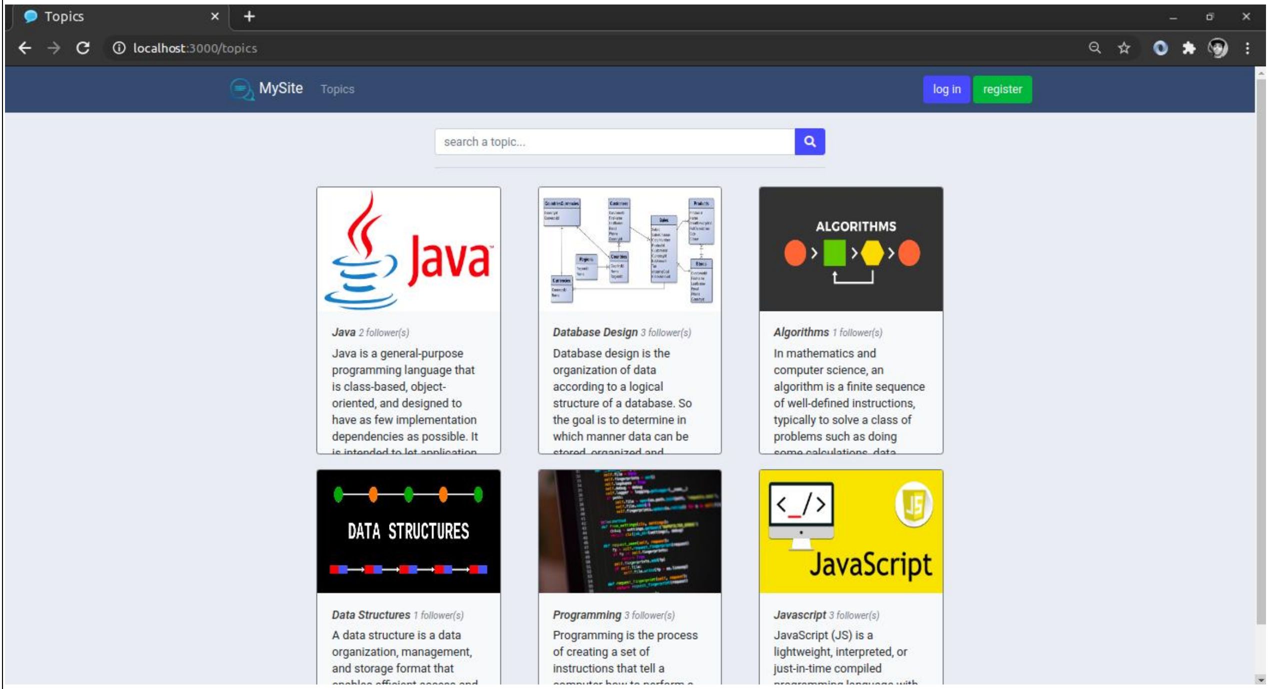
This is a ***many to many*** relationship

we will have an array named students that have for example: student_id, student_name, student_age. and an array named subjects: subject_id, subject_title, subject_hours.

to model the ManyToMany relationship we will have an additional array named **students_subjects** which will have two fields: ***student_id*** and ***subject_id.***

a record 1, 2 in this array means that the student with id 1 is subscribed to the subject with id 2.

3


Topics

localhost:3000/topics

MySite Topics log in register

search a topic...

Java *2 follower(s)*
Java is a general-purpose programming language that is class-based, object-oriented, and designed to have as few implementation dependencies as possible. It

Database Design *3 follower(s)*
Database design is the organization of data according to a logical structure of a database. So the goal is to determine in which manner data can be

Algorithms *1 follower(s)*
In mathematics and computer science, an algorithm is a finite sequence of well-defined instructions, typically to solve a class of problems such as doing

Data Structures *1 follower(s)*
A data structure is a data organization, management, and storage format that

Programming *3 follower(s)*
Programming is the process of creating a set of instructions that tell a

Javascript *3 follower(s)*
JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled

Conclusion:

A Social networking question and answer website can be a valuable resource for users seeking information and knowledge on a variety of topics. By leveraging the power of social networking, such a platform can create a community of users who can share their expertise, engage in discussions and learn from one another. It also helps as a career development resource where people can get answers for their confusion in choosing right career.

Future Scope:

As technology continues to evolve, there are several areas where a social networking question and answer website can continue to grow and improve like Artificial Intelligence, Mobile Optimization and Video Content. This web application can be developed using trending technologies like AI , ML and Cloud.

References:

The references that we have used in developing this Social Networking System are

1. https://chat.openai.com/
2. https://www.quora.com/
3. https://www.wikipedia.org/
4. https://www.npmjs.com/package/handlebars-react/
5. https://nodejs.dev/en/learn/
6. https://w3schools.com/mongodb/
7. https://www.mongodb.com/try/download/community