

# Evangadi Forum project Task breakdown

Note: The task number does not indicate task precedence.

## ***Backend Task***

### **1. Title: Create MySQL Database and Table Schema for Q&A Evangadi Forum Platform**

Description: Set up the MySQL database structure and schema for the Q&A Evangadi Forum platform to facilitate question and answer interactions among users.

Acceptance Criteria:

- Database Setup:
  - MySQL server should be installed and accessible.
  - Execute `CREATE DATABASE evangadi_forum;` SQL command successfully.
  - Verify database creation by listing databases .
- Table Schema Creation:
  - Design tables: `userTable`, `questionTable` , `answerTable`. Define fields and data types for each table to support Q&A functionalities.
  - Establish relationships between tables using foreign keys where applicable.
  - Include necessary constraints (e.g., primary keys, unique constraints).
- Schema Optimization:(This is optional only if you add more features)
  - Normalize tables to reduce redundancy and improve efficiency.
  - Consider indexing for performance optimization, especially for frequently queried fields.
- Database Configuration::
  - Create `config.js` or equivalent configuration file with MySQL connection details.
  - Ensure the configuration file accurately reflects MySQL connection parameters.
  - Verify successful database connectivity using the configured connection details.

Label: Database, Q&A Platform, MySQL

Dependency:

Task Points: 3days

Assignee: [ ]

## 2. Title: API Documentation Tasks for Evangadi Forum

Description: Document APIs for Evangadi Forum covering authentication, login, signup, and Q&A functionalities.

Acceptance Criteria:

1. Authentication Middleware:
  - Document the endpoint and method for authenticating users.
  - Specify required headers or parameters for authentication.
  - Describe the expected behavior and response status codes (e.g., 200 OK, 401 Unauthorized).
2. Sign-up:
  - Document the endpoint and method for user registration (signup).
  - Specify required fields ( username, first\_name, last\_name, email, password) in the request body.
  - Describe the expected behavior and response status codes (e.g., 201 Created, 400 Bad Request).
3. Login:
  - Document the endpoint and method for user login.
  - Specify required credentials ( email, password) in the request body.
  - Describe the expected behavior and response status codes (e.g., 200 OK, 401 Unauthorized).
4. Get Answers for a Question:
  - Document the endpoint and method to retrieve answers for a specific question.
  - Include query parameters or path variables if applicable.
  - Describe the response format and possible status codes (e.g., 200 OK, 404 Not Found).
5. Post Answers for a Question:
  - Document the endpoint and method to submit an answer for a question.
  - Describe the expected response format and status codes (e.g., 201 Created, 400 Bad Request).
6. Get All Questions:
  - Document the endpoint and method to fetch all questions.
  - Describe the response format and possible status codes (e.g., 200 OK, 404 Not Found).
7. Get Single Question:
  - Document the endpoint and method to retrieve details of a specific question.
  - Specify path variables or query parameters.

- Describe the response format and possible status codes (e.g., 200 OK, 404 Not Found).
8. Post Question:
- Document the endpoint and method to create a new question.
  - Describe the expected response format and status codes (e.g., 201 Created, 400 Bad Request).

Label: API Documentation

Task Points: 3 days

Assignee: [ ]

### 3. Title: Implement Authentication Middleware

Description: Develop authentication middleware to verify user identity and authorize access to protected endpoints.

Endpoint: `/api/checkUser`

Acceptance Criteria:

1. Middleware Implementation:
  - Create middleware to intercept requests and validate session tokens or credentials.
2. Token Verification:
  - Implement logic to verify JWT (JSON Web Token) or session tokens.
3. Error Handling:
  - Implement error handling for invalid or expired tokens (`401 Unauthorized`).
4. Security Considerations:
  - Implement HTTPS to protect token transmission and sensitive user data.

Label: API Development, Authentication, Middleware

Task Points: 3 days

Attachments:

- [Link to API documentation]

Assignee:[ ]

## 4. Title: Implement Sign-up API Endpoint

Description: Develop the Sign-up API endpoint to register new users with username, first name, last name, email, and password.

**Endpoint:** /api/register

Acceptance Criteria:

1. Endpoint Implementation:
  - Create the /api/register endpoint using the HTTP POST method.
2. Request Parameters:
  - Define required fields in the request body:
    - **username**: (string, required) Unique username for the user.
    - **first\_name**: (string, required) First name of the user.
    - **last\_name**: (string, required) Last name of the user.
    - **email**: (string, required) Email address of the user.
    - **password**: (string, required) Password for the user account.
3. User Registration Logic:
  - Implement validation and storage logic for the provided user details.
4. Password Security:
  - Implement secure password hashing (e.g., bcrypt) before storing in the database.
5. Response Handling:
  - Handle responses for successful user creation (**201 Created**) and validation errors (**400 Bad Request**).
6. Error Handling:
  - Implement error handling for missing or invalid request parameters.
  - Check for existing users with the same **username** or **email** to prevent duplicates.

Label: API Development, Authentication, Sign-up

Task Points: 3 days

Attachments:

- [Link to API documentation ]

Assignee: [ ]

## 5.Title: Implement Login API Endpoint

Description: Develop the Login API endpoint to authenticate users and issue session tokens for accessing protected resources.

**Endpoint:** /api/login

Acceptance Criteria:

1. Endpoint Implementation:
  - Create the /api/login endpoint using the HTTP POST method.
2. Authentication Logic:
  - Implement logic to validate user credentials (email and password).
3. Session Token Generation:
  - Generate a session token (JWT) upon successful authentication.
4. Response Handling:
  - Handle responses for successful authentication (200 OK) and invalid credentials (401 Unauthorized).
5. Error Handling:
  - Implement error handling for missing or invalid request parameters (Follow api doc).
6. Security Considerations:
  - Ensure sensitive information (e.g., passwords) is securely handled and transmitted over HTTPS.

Label: API Development, Authentication, Login-in

Task Points: 3 days

Attachments:

- [Link to API documentation ]

Assignee: [ ]

## 6. Title: Implement API Endpoint: Get Answers for a Question

Description: Develop the API endpoint to retrieve answers for a specific question.

Endpoint: `api/answer/:question_id`

Acceptance Criteria:

1. Endpoint Implementation:
  - Create the `/api/answer/:question_id` endpoint using the HTTP GET method.
2. Request Parameters:
  - Define path variable `question_id` to specify the question for which answers are requested.
3. Response Handling:
  - Handle responses with JSON payload containing answers and associated metadata.
4. Error Handling:
  - Implement error handling for invalid `question_id` or other client errors ([Follow api doc](#)).

Label: API Development, Get Answers, Question

Task Points: 3 days

Attachments:

- [\[Link to API documentation \]](#)

Assignee: [ ]

## 7. Title: Implement API Endpoint: Post Answers for a Question

Description: Develop the API endpoint to post an answer for a specific question.

Endpoint: `api/answer`

Acceptance Criteria:

1. Endpoint Implementation:
  - Create the `/api/answer` endpoint using the HTTP POST method.
2. Response Handling:
  - Handle responses for successful answer creation (`201 Created`) and validation errors (`Follow api doc`).
3. Error Handling:
  - Implement error handling for missing answers.

Label: API Development, Post Answer, Question

TaskPoints: 3 days

Attachments:

- [Link to API documentation]

Assignee: [ ]



## 8.Title: Implement API Endpoint: Get All Questions

Description: Develop the API endpoint to fetch all questions.

Endpoint: `api/question`

Acceptance Criteria:

1. Endpoint Implementation:
  - Create the `/api/question` endpoint using the HTTP GET method.
2. Response Handling:
  - Handle responses with JSON payload containing all questions and associated metadata.
3. Error Handling:
  - Implement error handling for client errors (`Follow api doc`) or server errors (`500 Internal Server Error`).

Label: API Development, Get All Questions

TaskPoints: 4 days

Attachments:

- [Link to API documentation]

Assignee: [ ]

## 9. Title: Implement API Endpoint: Get Single Question

Description: Develop the API endpoint to retrieve details of a specific question.

**Endpoint:** `api/question/:question_id`

Acceptance Criteria:

1. Endpoint Implementation:
  - Create the `api/question/:question_id` endpoint using the HTTP GET method.
2. Request Parameters:
  - Define path variable `question_id` to specify the question ID for retrieval.
3. Response Handling:
  - Handle responses with JSON payload containing details of the question.
4. Error Handling:
  - Implement error handling for invalid `question_id` ( **Not Found**) or other client errors (**Follow api doc**).

Label: API Development, Get Single Question

Task Points: 3 days

Attachments:

- [Link to API documentation]

Assignee: [ ]

## 10. Title: Implement API Endpoint: Post Question

Description: Develop the API endpoint to create a new question.

Endpoint: `api/question`

Acceptance Criteria:

1. Endpoint Implementation:
  - Create the `api/question` endpoint using the HTTP POST method.
2. Response Handling:
  - Handle responses for successful question creation (`201 Created`) and validation errors (`Follow api doc`).
3. Error Handling:
  - Implement error handling for missing or invalid request parameters.

Label: API Development, Post Question

TaskPoints: 3 days

Attachments:

- [Link to API documentation]

Assignee: [ ]

## Frontend Task

### 11. Title: Implement Header Component

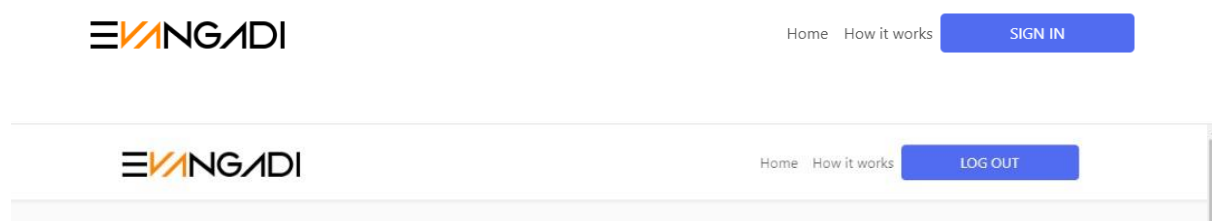
Description: Develop the Header component for the Evangadi Forum application, including navigation links.

Acceptance Criteria:

1. Component Structure:
  - Create the Header component structure with navigation links (Home, LogOut LogIn).
2. Styling:
  - Apply CSS styles to ensure a responsive and visually appealing header.
3. Routing:
  - Implement navigation links to route to corresponding pages.
4. LogOut functionality

Label: Frontend Development, Header

Attachments: use the below screenshot attachment.



TaskPoints: 2 days

Assignee: [ ]

## 12. Title: Implement Footer Component

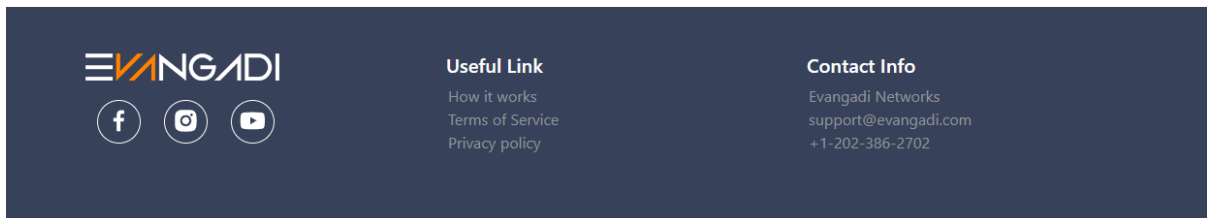
Description: Develop the Footer component for the Evangadi Forum application, including site links and contact information.

Acceptance Criteria:

1. Component Structure:
  - Create the Footer component structure with site links (Useful Link, Contact Info and Socials)
2. Styling:
  - Apply CSS styles to ensure a responsive and visually appealing footer.

Label: Frontend Development, Footer

Attachments: use the below screenshot attachment.



Task Points: 2 days

Assignee: [ ]

### 13. Title: Implement Sign-up Component

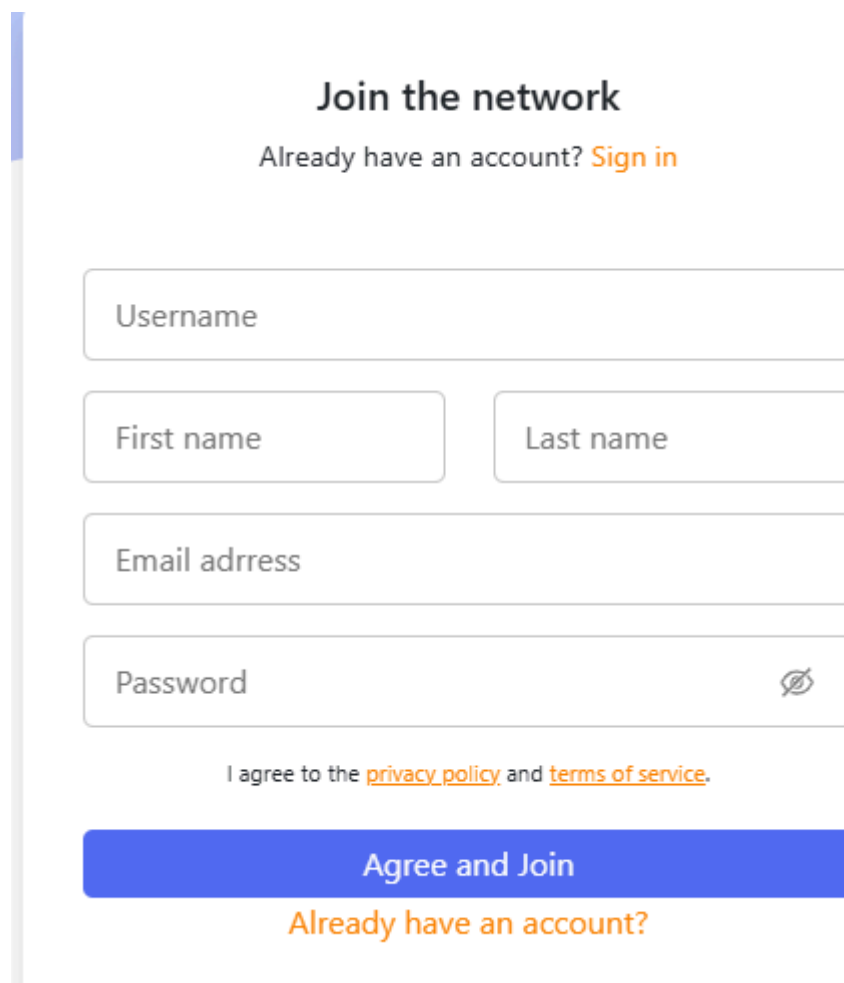
Description: Develop the Sign-up component for the Evangadi Forum application to register new users.

Acceptance Criteria:

1. Component Structure:
  - Create the Sign-up component with form fields (username, first name, last name, email, password).
2. Form Validation:
  - Implement form validation to ensure all fields are correctly filled out.
3. API Integration:
  - Integrate with the Sign-up API to submit user registration details.
4. Styling:
  - Apply CSS styles for a user-friendly and responsive form.

Label: Frontend Development, Sign-up

Attachments: use the below screenshot attachment.



The screenshot shows a web form titled "Join the network". Below the title is a link "Sign in" for users who already have an account. The form contains five input fields: "Username", "First name", "Last name", "Email address", and "Password". The "Password" field has a toggle icon (an eye) to switch between visible and hidden states. Below the input fields is a line of text: "I agree to the [privacy policy](#) and [terms of service](#).". At the bottom of the form is a blue button labeled "Agree and Join". Below the button is another link: "Already have an account?".

TaskPoints: 2days

Assignee: [ ]

## 14. Title: Implement Login-in Component

Description: Develop the Sign-in component for the Evangadi Forum application to authenticate users.

Acceptance Criteria:

1. Component Structure:
  - Create the Sign-in component with form fields (email, password).
2. Form Validation:
  - Implement form validation to ensure all fields are correctly filled out.
3. API Integration:
  - Integrate with the Login API to authenticate users.
4. Styling:
  - Apply CSS styles for a user-friendly and responsive form.

Label: Frontend Development, Sign-in

Attachments: use the below screenshot attachment.

## Login to your account

Don't have an account? [Create a new account](#)

[Forgot password?](#)

Login

TaskPoints: 3 days

Assignee: [ ]



## 15. Title: Implement About Component

Description: Develop the About component for the Evangadi Forum application to provide information about the platform.

Acceptance Criteria:

1. Component Structure:
  - Create the About component with information about Evangadi Network.
2. Styling:
  - Apply CSS styles to ensure a visually appealing and responsive layout.
3. Content Integration:
  - Add content such as text, and links.

Label: Frontend Development, About

Attachments: use the below screenshot attachment.

About

# Evangadi Networks

No matter what stage of life you are in, whether you're just starting elementary school or being promoted to CEO of a Fortune 500 company, you have much to offer to those who are trying to follow in your footsteps.

Whether you are willing to share your knowledge or you are just looking to meet mentors of your own, please start by joining the network here.

HOW IT WORKS

TaskPoints: 2 days

Assignee [ ]

## 16. Title: Implement Sign-up and Sign-in Page

Description: Develop the Sign-up and Sign-in page for the Evangadi Forum application to allow users to register and log in.

Acceptance Criteria:

1. Page Structure:
  - Create the Sign-up and Sign-in page with navigation between the Sign-up and Sign-in components.
2. Component Integration:
  - Integrate the Sign-up and Sign-in components on the page.
3. Styling:
  - Apply CSS styles for a consistent and user-friendly design.

Label: Frontend Development, Sign-up, Sign-in



### Login to your account

Don't have an account? [Create a new account](#)



[Forgot password?](#)

### About

## Evangadi Networks

No matter what stage of life you are in, whether you're just starting elementary school or being promoted to CEO of a Fortune 500 company, you have much to offer to those who are trying to follow in your footsteps.

Whether you are willing to share your knowledge or you are just looking to meet mentors of your own, please start by joining the network here.

[HOW IT WORKS](#)

#### Useful Link

[How it works](#)  
[Terms of Service](#)  
[Privacy policy](#)

#### Contact Info

Evangadi Networks  
[support@evangadi.com](mailto:support@evangadi.com)  
+1-202-386-2702

Task Points: 2 days

Assignee: [ ]

## 17. Title: Implement Home Page

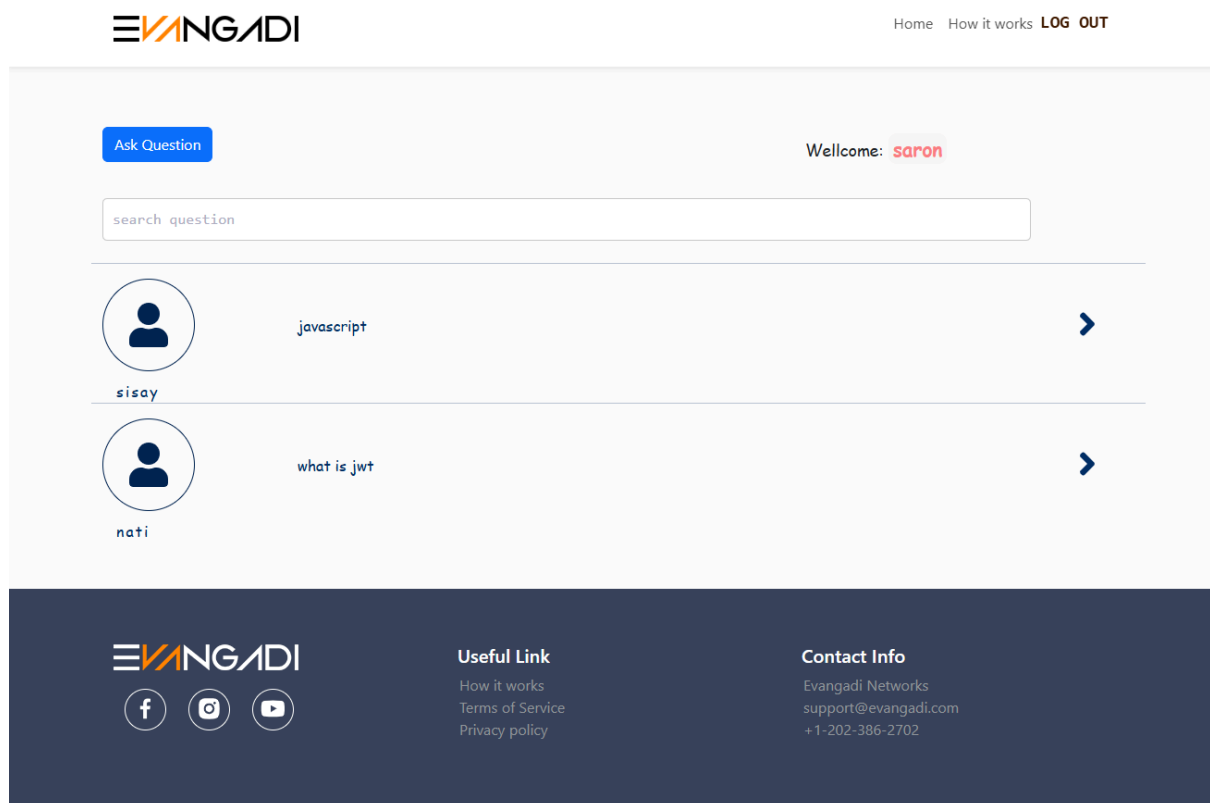
Description: Develop the Home page for the Evangadi Forum application, serving as the landing page for users.

Acceptance Criteria:

1. Page Structure:
  - Create the Home page.
2. API Integration
3. Styling:
  - Apply CSS styles to ensure a visually appealing and responsive layout.

Label: Frontend Development, Home

Attachments: use the below screenshot attachment.



TaskPoints: 3 days

Assignee: [ ]

## 18. Title: Implement Question Page

Description: Develop the Question page for the Evangadi Forum application to display details of a specific question.

Acceptance Criteria:

1. Page Structure:
  - Create the Question page with sections for the question, answers, and answer form.
2. API Integration
3. Content Display:
  - Display question details and list of answers.
4. Answer Submission:
  - Include a form for submitting new answers.
5. Styling:
  - Apply CSS styles to ensure a visually appealing and responsive layout.

Label: Frontend Development, Question

Attachments: use the below screenshot attachment.

## Steps To Write A Good Question.

- ➔ Summarize your problems in a one-line-title.
- ➔ Describe your problem in more detail.
- ➔ Describe what you tried and what you expected to happen.
- ➔ Review your question and post it here.

## Post Your Question

Task Points: 3 days

Assignee: [ ]

## 19. Title: Implement Answer Page

Description: Develop the Answer page for the Evangadi Forum application to display and submit answers to questions.


Acceptance Criteria:

1. Page Structure:
  - Create the Answer page with sections for the question, answers, and answer form.
2. API Integration
3. Content Display:
  - Display the question and existing answers.
4. Answer Submission:
  - Include a form for submitting new answers.
5. Styling:
  - Apply CSS styles to ensure a visually appealing and responsive layout.

Label: Frontend Development, Answer




Attachment: use the below screenshot attachment.




HomeHow it worksLOG OUT

QUESTION

 **JavaScript**

what is javascript

Answer From The Community







programming language

sisay

Your answer ...

Post Answer





Useful Link

[How it works](#)[Terms of Service](#)[Privacy policy](#)

Contact Info

[Evangadi Networks](#)[support@evangadi.com](mailto:support@evangadi.com)[+1-202-386-2702](tel:+1-202-386-2702)

Task Points: 3days

Assignee: [ ]