# **Discussion Forum API Documentation**

# Domains :

### User

* Discussion
* Comment
* Subscription

# 

# 

# 

# User :

#### **Register:**

* **Endpoint**: POST /register
* **Description**: Register a new user by providing name, email, and password.
* **Auth Required**: No

**Field Requirements**:

| **Field** | **Type** | **Required** | **Validation Rules** |
| --- | --- | --- | --- |
| name | string | Yes | 3–50 characters |
| email | string | Yes | Must be a valid email |
| password | string | Yes | Minimum 6 characters |

**Request Body Example**:

{

"name": "John Doe",

"email": "john@example.com",

"password": "secure123"

}

**Response (201 Created)**:

{

"message": "User registered successfully"

}

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 400 | Missing or invalid fields |
| 409 | Email already in use |

#### **Login**

* **Endpoint**: POST /login
* **Description**: Log in an existing user and receive a JWT token.
* **Auth Required**: No

**Field Requirements**:

| Field | Type | Required | Validation Rules |
| --- | --- | --- | --- |
| email | string | Yes | Must be a valid email |
| password | string | Yes | - |

**Request Body Example**:

{

"email": "john@example.com",

"password": "secure123"

}

**Response (200 OK)**:

{

"token": "<JWT\_TOKEN>"

}

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 401 | Invalid email or password |

# Discussion :

## Create Topic

* **Endpoint**: POST /topics
* **Description**: Create a new discussion topic.
* **Auth Required**: Yes

**Field Requirements**:

| Field | Type | Required | Validation Rules |
| --- | --- | --- | --- |
| title | string | Yes | 5–100 characters |
| content | string | Yes | 10–5000 characters |
| tags | array | No | List of strings |
| schedule\_time | datetime | No | RFC3339 format |

**Request Body Example**:

{

"title": "What is Golang?",

"content": "Let's discuss concurrency in Go",

"tags": ["golang", "backend"],

"schedule\_time": "2025-06-14T12:00:00Z"

}

**Response (201 Created)**:

{

"message": "Topic created successfully",

"topic\_id": "abc123"

}

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 400 | Validation errors |
| 401 | Unauthorized |

## Get All Topics

* **Endpoint**: GET /topics
* **Description**: Retrieve all discussion topics.
* **Auth Required**: No

**Response Example**:

[

{

"id": "abc123",

"title": "What is Golang?",

"content": "Let's discuss concurrency in Go",

"tags": ["golang", "backend"],

"created\_at": "2025-06-14T11:00:00Z",

"author": {

"id": "user123",

"name": "John Doe"

}

}

]

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 500 | Server error |

## Get Topics by User

* **Endpoint**: GET /topics/user/{user\_id}
* **Description**: Get all topics created by a specific user.
* **Auth Required**: No

**Response Example**:

[

{

"id": "abc123",

"title": "User's Topic",

"content": "Content...",

"tags": ["sample"],

"created\_at": "2025-06-14T11:00:00Z"

}

]

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 404 | User not found |
| 500 | Server error |

## 4.. Get Topics by Tag

* **Endpoint**: GET /topics/tag/{tag}
* **Description**: Get topics filtered by a specific tag.
* **Auth Required**: No

**Response Example**:

[

{

"id": "topic123",

"title": "Learning Golang",

"tags": ["golang"]

}

]

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 404 | No topics found |
| 500 | Server error |

## 5. Update Topic

* **Endpoint**: PUT /topics/{id}
* **Description**: Update a topic.
* **Auth Required**: Yes

**Field Requirements**: Same as Create Topic.

**Request Body Example:**

{

"title": "Updated Title",

"content": "Updated content...",

"tags": ["go"]

}

**Response Body Example:**

{

“message”: “topic updated Succesfully”

}

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 400 | Validation errors |
| 401 | Unauthorized |
| 404 | Topic not found |

## 5. Delete Topic

* **Endpoint**: DELETE /topics/{id}
* **Description**: Delete a topic.
* **Auth Required**: Yes

**Response Example**:

{

"message": "Topic deleted successfully"

}

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 401 | Unauthorized |
| 404 | Topic not found |

# Comment :

## Add Comment

* **Endpoint**: POST /comments
* **Description**: Add a comment to a topic.
* **Auth Required**: Yes

**Field Requirements**:

| Field | Type | Required | Validation Rules |
| --- | --- | --- | --- |
| topic\_id | string | Yes | Must be valid topic UUID |
| content | string | Yes | 1–2000 characters |

**Request Body Example**:

{

"topic\_id": "abc123",

"content": "This is a great topic!"

}

**Response Example**:

{

"message": "Comment added successfully",

"comment\_id": "cmt456"

}

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 400 | Validation errors |
| 401 | Unauthorized |
| 404 | Topic not found |

# Subscription

## Subscribe to Topic

* **Endpoint**: POST /subscribe
* **Description**: Subscribe to a topic by email to get update notifications.
* **Auth Required**: No

**Field Requirements**:

| Field | Type | Required | Validation Rules |
| --- | --- | --- | --- |
| email | string | Yes | Must be a valid email |
| topic\_id | string | Yes | Must be valid topic UUID |

**Request Body Example**:

{

"email": "user@example.com",

"topic\_id": "abc123"

}

**Response (200 OK)**:

{

"message": "Subscribed successfully"

}

**Error Responses**:

| Status Code | Reason |
| --- | --- |
| 400 | Validation errors |
| 404 | Topic not found |

### 

### 

### 

### **Notes**

* JWT token must be provided in the Authorization: Bearer <token> header.
* Scheduled discussions will be auto-posted at the scheduled time.
* Subscribers receive automated email alerts on topic updates.
* All endpoints return JSON responses.