# API Documentation: QnA Chat Session Management

## Base URL

http://your-domain.com/qna/api/chat/session

## Authentication

This API requires authentication via **Token Authentication**. Include the token in the Authorization header:

Authorization: Token <your-token>

## 1. Create a New Chat Session

### ****POST**** /qna/api/chat/session

Creates a new chat session without any documents initially.

### Request Body (JSON)

None

### Response

#### ****Success (201)****

{

"message": "Session created successfully.",

"sessionUUID": "123e4567-e89b-12d3-a456-426614174000"

}

#### ****Error Responses****

* **500** - Internal server error

## 2. Add Documents to a Session

### ****PUT**** /qna/api/chat/session

Adds new documentUUIDs to an existing chat session.

### Request Body (JSON)

{

"sessionUUID": "123e4567-e89b-12d3-a456-426614174000",

"documentUUIDs": ["doc-uuid-1", "doc-uuid-2"]

}

### Response

#### ****Success (200)****

{

"message": "Documents added to session successfully.",

"sessionUUID": "123e4567-e89b-12d3-a456-426614174000",

"documentUUIDs": ["doc-uuid-1", "doc-uuid-2"]

}

#### ****Error Responses****

* **400** - No sessionUUID or documentUUIDs provided
* **404** - Session not found or unauthorized
* **400** - Invalid documentUUIDs
* **500** - Internal server error

## OpenAPI Specification (Swagger)

openapi: 3.0.0

info:

title: QnA Chat Session API

version: 1.0.0

description: API for managing chat sessions and associated documents.

servers:

- url: http://your-domain.com

description: Production server

paths:

/qna/api/chat/session:

post:

summary: Create a new chat session

security:

- TokenAuth: []

responses:

'201':

description: Session successfully created

content:

application/json:

schema:

type: object

properties:

message:

type: string

sessionUUID:

type: string

format: uuid

'500':

description: Internal server error

put:

summary: Add documents to an existing session

security:

- TokenAuth: []

requestBody:

required: true

content:

application/json:

schema:

type: object

properties:

sessionUUID:

type: string

format: uuid

documentUUIDs:

type: array

items:

type: string

responses:

'200':

description: Documents added to session successfully

content:

application/json:

schema:

type: object

properties:

message:

type: string

sessionUUID:

type: string

format: uuid

documentUUIDs:

type: array

items:

type: string

'400':

description: Bad request (missing parameters or invalid documentUUIDs)

'404':

description: Session not found or unauthorized

'500':

description: Internal server error

components:

securitySchemes:

TokenAuth:

type: http

scheme: bearer

## Logical Steps Behind the API

### 1. Authentication

* The API uses **Token Authentication** (Token <your-token>).
* The server validates the token before processing any request.

### 2. Creating a Chat Session (POST /qna/api/chat/session)

1. User sends a **POST** request to /qna/api/chat/session.
2. The server validates the authentication token.
3. A new sessionUUID is generated and stored in the database.
4. The response contains the new sessionUUID.

### 3. Adding Documents to a Session (PUT /qna/api/chat/session)

1. User sends a **PUT** request with sessionUUID and documentUUIDs.
2. The server validates:
   * Authentication token
   * If sessionUUID exists and belongs to the user
   * If provided documentUUIDs belong to the user
3. If valid, the documents are linked to the session.
4. The response confirms the successful association.

### 4. Security Measures

* **Authentication**: Users must send a valid token for access.
* **Authorization**: Only the session owner can modify its documents.
* **Data Validation**:
  + Ensures required fields are present.
  + Checks document ownership.
* **Error Handling**: Returns meaningful error messages.