# API DOCUMENTATION

## Conversation MicroService

```
openapi: 3.0.0
info:
  title: AI Conversation API
  description: API to interact with an AI chat service
  version: 1.0.0
paths:
  /chat:
    post:
      summary: Get AI Chat Response
      description: Sends a chat prompt to the AI service and returns the response.
      requestBody:
        description: The chat prompt to send to the AI service
        required: true
        content:
          application/json:
            schema:
              type: string
              example: "Hello, how are you?"
      responses:
        '200':
          description: AI response received
          content:
            application/json:
              schema:
                type: string
                example: "I'm doing well, thank you!"
        '400':
          description: Invalid request
        '500':
          description: Internal server error
```

## Person MicroService

```
openapi: 3.0.0
info:
  title: User Authentication API
  description: API for user login, registration, and retrieval
  version: 1.0.0
paths:
  /login:
    post:
      summary: User Login
      description: Authenticates the user and returns a JWT token.
      requestBody:
        description: User credentials for login
        required: true
```

```yaml
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/CredentialsDto'
      responses:
        '200':
          description: Successfully authenticated
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/UserDto'
        '400':
          description: Invalid credentials
        '500':
          description: Server error
  /register:
    post:
      summary: User Registration
      description: Registers a new user and returns the user data with a JWT token.
      requestBody:
        description: User details for registration
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SignUpDto'
      responses:
        '201':
          description: Successfully registered
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/UserDto'
        '400':
          description: Invalid input data
        '500':
          description: Server error
  /users/{id}:
    get:
      summary: Get User by ID
      description: Retrieves a user by their ID.
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: integer
          description: ID of the user to retrieve
      responses:
        '200':
          description: User found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/UserDto'
        '404':
          description: User not found
        '500':
          description: Server error

components:
  schemas:
    CredentialsDto:
      type: object
      properties:
        login:
          type: string
          example: "john_doe"
        password:
          type: string
          example: "password123"  # Note: Password should be char[] but displayed as string here for
simplicity
    SignUpDto:
      type: object
      properties:
        firstName:
          type: string
          example: "John"
        lastName:
          type: string
          example: "Doe"
        login:
          type: string
          example: "john_doe"
        password:
          type: string
          example: "password123"  # Password represented as string for simplicity
    UserDto:
      type: object
      properties:
        id:
          type: integer
          example: 1
        login:
          type: string
          example: "john_doe"
        email:
          type: string
          example: "john.doe@example.com"
        token:
          type: string
          example: "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
```

## Emergency MicroService

```
openapi: 3.0.0
info:
  title: Location Sharing API
  description: API to share location information via SMS.
  version: 1.0.0
```

```yaml
paths:
  /api/location/share:
    post:
      summary: Share Location via SMS
      description: Shares a location with a list of phone numbers by sending an SMS.
      requestBody:
        description: Location information and phone numbers to share the location with
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/LocationRequest'
      responses:
        '200':
          description: SMS sent successfully
        '400':
          description: Bad request, invalid location or phone number format
        '500':
          description: Internal server error

components:
  schemas:
    LocationRequest:
      type: object
      properties:
        lat:
          type: number
          format: double
          description: Latitude of the location
          example: 40.730610
        lng:
          type: number
          format: double
          description: Longitude of the location
          example: -73.935242
        phoneNumbers:
          type: array
          items:
            type: string
          description: List of phone numbers to send the location SMS to
          example: ["+1234567890", "+0987654321"]
```

## MOOD Recommendation Micro Service

```yaml
openapi: 3.0.0
info:
  title: Mood Tracker API
  description: API for tracking mood entries, weekly reports, and recommendations.
  version: 1.0.0
paths:
  /api/mood/save:
    post:
      summary: Save Mood Entry
      description: Saves a new mood entry with details like mood score, journal entry, sleep hours, and
water intake.
      requestBody:
        description: Mood entry data to be saved
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/MoodEntry'
      responses:
        '201':
          description: Mood entry created successfully
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/MoodEntry'
        '500':
          description: Internal server error

  /api/mood/weekly-report:
    get:
      summary: Get Weekly Mood Report
      description: Retrieves the weekly mood report for a specific user or all users.
      parameters:
        - name: userId
          in: query
          required: false
          schema:
            type: integer
          description: ID of the user to retrieve the weekly report for
      responses:
        '200':
          description: Weekly mood report retrieved successfully
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/MoodEntry'
        '204':
          description: No content found
        '500':
          description: Internal server error

  /api/mood/recommendations:
    get:
      summary: Get Mood Recommendations
      description: Generates recommendations based on mood entries for a specific user or all users.
      parameters:
        - name: userId
          in: query
          required: false
          schema:
            type: integer
          description: ID of the user to retrieve recommendations for
      responses:
        '200':
```

```yaml
        description: Recommendations retrieved successfully
        content:
          application/json:
            schema:
              type: string
      '500':
        description: Internal server error

  /api/mood/entries:
    get:
      summary: Get All Mood Entries
      description: Retrieves all mood entries for a specific user or all users.
      parameters:
        - name: userId
          in: query
          required: false
          schema:
            type: integer
          description: ID of the user to retrieve entries for
      responses:
        '200':
          description: List of mood entries retrieved successfully
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/MoodEntry'
        '204':
          description: No content found
        '500':
          description: Internal server error

  /api/mood/entries/{id}:
    delete:
      summary: Delete Mood Entry
      description: Deletes a mood entry by its ID.
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: integer
          description: ID of the mood entry to delete
      responses:
        '204':
          description: Mood entry deleted successfully
        '500':
          description: Internal server error

components:
  schemas:
    MoodEntry:
      type: object
      properties:
        id:
          type: integer
          description: Unique ID of the mood entry
          example: 1
        userId:
          type: integer
          description: User ID associated with the mood entry
          example: 123
        moodScore:
          type: integer
          description: Mood score ranging from 1 to 10
          example: 7
        journalEntry:
          type: string
          description: Journal entry for the day
          example: "Today was a productive day!"
        sleepHours:
          type: integer
          description: Number of hours slept
          example: 8
        waterIntake:
          type: integer
          description: Amount of water intake in liters
          example: 2
        entryDate:
          type: string
          format: date
          description: Date of the mood entry
          example: "2024-09-12"
```

## Recommendation System Micro Service

```
openapi: 3.0.0
info:
  title: YouTube Recommendation System API
  description: API for fetching videos from YouTube based on a search query.
  version: 1.0.0
paths:
  /api/videos:
    get:
      summary: Get YouTube Videos
      description: Fetches a list of YouTube videos based on a search query.
      parameters:
        - name: query
          in: query
          required: true
          schema:
            type: string
          description: Search query for YouTube videos
          example: "Java tutorials"
      responses:
        '200':
          description: List of YouTube search results
          content:
```

```yaml
      application/json:
       schema:
        type: array
        items:
         $ref: '#/components/schemas/SearchResult'
    '500':
     description: Internal server error
components:
 schemas:
  SearchResult:
   type: object
   properties:
    kind:
     type: string
     description: Type of the search result
     example: "youtube#searchResult"
    etag:
     type: string
     description: ETag of the search result
     example: "XI7nbFXulYBIpL0ayR_gDh3eu1k/8q"
    id:
     type: object
     description: Information about the video or channel
     properties:
      kind:
       type: string
       description: Type of resource (e.g., video, channel, playlist)
       example: "youtube#video"
      videoId:
       type: string
       description: ID of the video
       example: "Ks-_Mh1QhMc"
    snippet:
     type: object
     description: Metadata of the search result
     properties:
      publishedAt:
       type: string
       format: date-time
       description: Date when the video was published
       example: "2023-01-15T00:00:00Z"
      title:
       type: string
       description: Title of the video
       example: "Learn Java in 30 minutes"
      description:
       type: string
       description: Description of the video
       example: "This video will give you a quick overview of Java programming."
      channelTitle:
       type: string
       description: The channel title the video belongs to
       example: "Java Coding"
```

## Task MicroService

```
openapi: 3.0.0
info:
  title: Task Manager API
  description: API for managing tasks, including CRUD operations, priority management, notifications,
and reminders.
  version: 1.0.0
paths:
  /api/tasks:
    get:
      summary: Get all tasks
      description: Retrieve a list of all tasks.
      responses:
        '200':
          description: List of tasks
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Task'
        '500':
          description: Internal server error
    post:
      summary: Create a new task
      description: Add a new task to the system.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Task'
      responses:
        '201':
          description: Task created successfully
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Task'
        '400':
          description: Bad request

  /api/tasks/{id}:
    get:
      summary: Get task by ID
      description: Retrieve a task by its ID.
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: integer
          description: ID of the task
      responses:
        '200':
```

```yaml
          description: Task found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Task'
        '404':
          description: Task not found
    delete:
      summary: Delete task by ID
      description: Delete a task by its ID.
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: integer
          description: ID of the task
      responses:
        '204':
          description: Task deleted successfully
        '404':
          description: Task not found

  /api/tasks/priority/{priority}:
    get:
      summary: Get tasks by priority
      description: Retrieve tasks filtered by their priority.
      parameters:
        - name: priority
          in: path
          required: true
          schema:
            type: string
          description: Priority level of tasks (e.g., "high", "medium", "low")
      responses:
        '200':
          description: List of tasks with specified priority
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Task'
        '404':
          description: Tasks not found

  /api/tasks/{id}/text:
    patch:
      summary: Update task text
      description: Update the text of an existing task.
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: integer
          description: ID of the task
    requestBody:
      required: true
      content:
        application/json:
          schema:
            type: object
            properties:
              text:
                type: string
                example: "New task description"
    responses:
      '200':
        description: Task updated successfully
      '400':
        description: Invalid input data
      '404':
        description: Task not found

/api/tasks/{id}/priority:
  patch:
    summary: Update task priority
    description: Update the priority of an existing task.
    parameters:
      - name: id
        in: path
        required: true
        schema:
          type: integer
        description: ID of the task
    requestBody:
      required: true
      content:
        application/json:
          schema:
            type: object
            properties:
              priority:
                type: string
                example: "high"
    responses:
      '200':
        description: Task priority updated successfully
      '400':
        description: Invalid input data
      '404':
        description: Task not found

/api/tasks/{id}/reminder:
  patch:
    summary: Update task reminder time
    description: Update the reminder time for a task.
    parameters:
      - name: id
        in: path
        required: true
        schema:
          type: integer
      description: ID of the task
    requestBody:
     required: true
     content:
      application/json:
       schema:
        type: object
        properties:
         reminderTime:
          type: string
          format: date-time
          example: "2024-09-15T10:00:00"
    responses:
     '200':
      description: Task reminder time updated successfully
     '400':
      description: Invalid reminder time format
     '404':
      description: Task not found

  /api/tasks/user/{userId}:
   get:
    summary: Get tasks by user ID
    description: Retrieve tasks assigned to a specific user.
    parameters:
     - name: userId
      in: path
      required: true
      schema:
       type: integer
      description: ID of the user
    responses:
     '200':
      description: List of tasks for the user
      content:
       application/json:
        schema:
         type: array
         items:
          $ref: '#/components/schemas/Task'
     '404':
      description: No tasks found for the user

  /api/tasks/notifications/{userId}:
   get:
    summary: Get tasks with notifications for a user
    description: Retrieve tasks for a user that have notifications or reminders.
    parameters:
     - name: userId
      in: path
      required: true
      schema:
       type: integer
      description: ID of the user
    responses:
     '200':
      description: List of tasks with notifications for the user
```

```
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Task'
        '404':
          description: No tasks with notifications found

components:
  schemas:
    Task:
      type: object
      properties:
        id:
          type: integer
          description: Task ID
          example: 1
        text:
          type: string
          description: Task description
          example: "Finish the project report"
        priority:
          type: string
          description: Task priority
          example: "high"
        reminderTime:
          type: string
          format: date-time
          description: Task reminder time
          example: "2024-09-15T10:00:00"
        notified:
          type: boolean
          description: Whether the user was notified
          example: false
        completed:
          type: boolean
          description: Whether the task is completed
          example: false
        userId:
          type: integer
          description: ID of the user who created the task
          example: 101
```

## Test Micro Service

```
openapi: 3.0.0
info:
  title: Test Score API
  description: API for submitting and retrieving user test scores.
  version: 1.0.0
servers:
  - url: http://localhost:8080
```

```yaml
    description: Local server

paths:
  /test-scores:
    post:
      summary: Submit a test score
      description: Save the test score for a user.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/TestScoreDto'
      responses:
        '200':
          description: Test score saved successfully
          content:
            application/json:
              schema:
                type: string
                example: Test score saved successfully

  /testHistory:
    get:
      summary: Get test history
      description: Retrieve the test history for a user based on their userId.
      parameters:
        - name: userId
          in: query
          required: true
          schema:
            type: string
          description: The ID of the user to retrieve test history for.
      responses:
        '200':
          description: A list of test scores
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/TestScore'
        '404':
          description: No test scores found for the given userId

components:
  schemas:
    TestScore:
      type: object
      properties:
        id:
          type: integer
          format: int64
        date:
          type: string
          format: date
        userId:
```

```yaml
      type: string
    score:
      type: integer
    depressionLevel:
      type: string
  example:
    id: 1
    date: "2024-09-01"
    userId: "12345"
    score: 75
    depressionLevel: "Moderate"

TestScoreDto:
  type: object
  properties:
    id:
      type: integer
      format: int32
    userId:
      type: string
    date:
      type: string
      format: date
    score:
      type: integer
    depressionLevel:
      type: string
  example:
    id: 1
    userId: "12345"
    date: "2024-09-01"
    score: 75
    depressionLevel: "Moderate"

UserDto:
  type: object
  properties:
    id:
      type: integer
    firstName:
      type: string
    lastName:
      type: string
    login:
      type: string
    token:
      type: string
  example:
    id: 123
    firstName: "John"
    lastName: "Doe"
    login: "johndoe"
    token: "jwt-token"
```