1. Introduction

This Software Requirements Specification (SRS) outlines the requirements for the Complaint / Helpdesk Management System. The system enables users to register, authenticate, create complaint tickets, upload images, comment on tickets, and manage ticket workflows. It includes role-based access control across three roles: **customer**, **agent**, and **admin**.

2. Objective and Scope

2.1 Objective

The system aims to:

- Provide secure and reliable complaint management.
- Allow customers to create tickets with text and images.
- Enable agents and admins to manage and update ticket statuses.
- Facilitate communication between users through comments.
- Provide image retrieval functionality for uploaded files.
- Ensure proper access control and security using JWT authentication.

2.2 Scope

The system includes:

- User registration & login (JWT-based authentication).
- Role-based access control: **customer**, **agent**, **admin**.
- Ticket creation, viewing, image upload, and status updates.
- Comment creation and viewing per ticket.
- Secure retrieval of stored images.
- Backend API services according to the OpenAPI specification.

3. Functional Requirements

3.1 Authentication

FR1.1 — User Registration

- Users shall be able to register using:
  - username
  - email
  - password
- Input validation shall follow the schema `UserRegister`.

FR1.2 — User Login

- Users shall be able to authenticate using:
  - email
  - password
- Successful login returns a **JWT token**, as defined in the API.
- Unauthenticated users cannot access protected endpoints.

3.2 User Management

FR2.1 — Role-Based Access

The system shall support three roles:

- **Customer**
  - Create tickets
  - View their own tickets
  - Add comments to their own tickets
- **Agent**
  - View all tickets
  - Update ticket status
  - Comment on any ticket
- **Admin**
  - Full access to all tickets and users
  - Same ticket privileges as agents

Unauthorized and forbidden responses are returned based on API rules (`401` and `403`).

3.3 Ticket Management

FR3.1 — Get All Tickets

- Endpoint: `GET /api/tickets`
- Accessible by:
  - **Agents**
  - **Admins**
  - **Customers** (restricted to their own tickets, enforced by backend logic)

FR3.2 — Create Ticket

- Endpoint: `POST /api/tickets`
- Customers can create a new ticket with:
  - title (required)
  - description (required)
  - optional image uploads (multipart/form-data)

Uploaded images are stored and referenced through `image_ids` .

FR3.3 — Get Ticket by ID

- Endpoint: `GET /api/tickets/{ticket_id}`
- Users can view:
  - Their own tickets (customers)
  - All tickets (agents, admins)

Forbidden access is returned when customer tries to access others' tickets.

FR3.4 — Update Ticket (Status)

- Endpoint: `PUT /api/tickets/{ticket_id}`
- Only **agents** and **admins** may update a ticket's status.
- Allowed status values:
  - `open`
  - `in_progress`
  - `closed`

3.4 Comment Management

FR4.1 — Get Comments for a Ticket

- Endpoint: `GET /api/tickets/{ticket_id}/comments`
- Any authenticated user with permission to view the ticket may view comments.

FR4.2 — Add Comment to Ticket

- Endpoint: `POST /api/tickets/{ticket_id}/comments`
- Request body: `{ "text": "comment message" }`
- All roles (customer, agent, admin) may comment on tickets they have access to.

3.5 Image Retrieval

FR5.1 — Get Image by ID

- Endpoint: `GET /api/images/{image_id}`
- Returns binary JPEG data.
- If the image does not exist, the API returns `404`.

4. Non-Functional Requirements

4.1 Performance

- **NFR1.1 Response Time**
  All API requests should respond within **2 seconds** under normal load.
- **NFR1.2 Scalability**
  The backend shall scale to support growth in:
  - User base
  - Ticket volume
  - Image uploads

4.2 Security

- **NFR2.1 Authentication**
  All protected endpoints require JWT (`BearerAuth`).
- **NFR2.2 Password Protection**
  Passwords must be securely hashed and never stored in plain text.
- **NFR2.3 Authorization**
  Access restrictions:
  - Unauthorized → `401`
  - Forbidden → `403`
- **NFR2.4 Data Protection**
  Uploaded images must not be publicly accessible without permission.

4.3 Usability

- **NFR3.1 UI Simplicity**
  The frontend (outside API scope) must present an intuitive interface for:
  - Submitting tickets
  - Uploading images
  - Viewing comments and statuses
- **NFR3.2 Accessibility**
  Interfaces should follow basic accessibility guidelines.

4.4 Maintainability

- **NFR4.1 Code Quality**
  The system codebase should follow clean architecture and be well-documented.
- **NFR4.2 Testability**
  API endpoints must be structured to support unit and integration testing.

5. System Overview Diagram