**API Documentation**

**GET /api/ping**

Simple ping endpoint to check if the server is responsive.

**Headers:**

N/A

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"status": "pong",

"message": "Server is running"

}

**Error (500):**

JSON

{

"status": "error",

"message": "Server error"

}

**POST /api/users/register**

Create a new user. Requires admin privileges.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| first\_name | string | Required | User's first name. |
| last\_name | string | Required | User's last name. |
| email | string | Required | User's email address (must be unique). |
| contact\_number | string | Required | User's contact phone number. |
| username | string | Required | Username (must be unique, min 4 chars). |
| password | string | Required | User's password (min 8 chars). |
| role\_id | integer | Required | ID of the role to assign. |
| environment\_id | integer | Required | ID of the environment to assign. |

**Example:**

JSON

{

"first\_name": "John",

"last\_name": "Doe",

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

"contact\_number": "1234567890",

"username": "johndoe",

"password": "securePassword123",

"role\_id": 2,

"environment\_id": 1

}

**Responses:**

**Success (201):**

JSON

{

"message": "User created successfully",

"user": {

"id": 1,

"username": "johndoe",

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

"first\_name": "John",

"last\_name": "Doe",

"contact\_number": "1234567890",

"role": {

"id": 2,

"name": "Technician",

"description": "Technical staff",

"is\_super\_user": false

},

"environment": {

"id": 1,

"name": "Default",

"description": "Default environment"

},

"is\_deleted": false,

"created\_at": "2025-04-27T20:05:29Z",

"updated\_at": "2025-04-27T20:05:29Z"

}

}

**Error (400):**

JSON

{

"error": "Missing required fields | Username already exists | Email already exists | Password must be at least 8 characters long | ..."

}

**POST /api/users/login**

Authenticate a user and receive a JWT.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| username | string | Required | Registered username. |
| password | string | Required | User's password. |

**Example:**

JSON

{

"username": "johndoe",

"password": "securePassword123"

}

**Responses:**

**Success (200):**

JSON

{

"access\_token": "<JWT\_TOKEN>"

}

**Error (401):**

JSON

{

"error": "Invalid credentials"

}

**POST /api/users/logout**

Logs out the user. Invalidates the token if possible (depends on implementation, often relies on client-side token removal).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token (optional, aids logging) | Optional | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"message": "Successfully logged out",

"status": "success"

}

**Error:** N/A (Typically designed to succeed even with invalid tokens)

**GET /api/users**

Get a list of users. Filters based on user role (Admin sees all, others see users in their environment).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| include\_deleted | boolean | Optional | Include soft-deleted users (Admin only). Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

[

{

"id": 1,

"username": "johndoe",

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

"first\_name": "John",

"last\_name": "Doe",

"contact\_number": "1234567890",

"role": {

"id": 2,

"name": "Technician",

"description": "Technical staff",

"is\_super\_user": false

},

"environment": {

"id": 1,

"name": "Default",

"description": "Default environment"

},

"is\_deleted": false,

"created\_at": "2025-04-27T20:05:29Z",

"updated\_at": "2025-04-27T20:05:29Z"

},

{ ... }

]

**Error (500):**

JSON

{

"error": "Internal server error | Database error: <details>"

}

**GET /api/users/batch**

Get a paginated list of users in a compact format.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| page | integer | Optional | Page number. Default: 1. |
| per\_page | integer | Optional | Items per page. Default: 50. |
| include\_deleted | boolean | Optional | Include soft-deleted users (Admin only). Default: false. |
| role\_id | integer | Optional | Filter by role ID. |
| environment\_id | integer | Optional | Filter by environment ID (Admin only, otherwise uses user's environment). |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"metadata": {

"total\_items": 5,

"total\_pages": 1,

"current\_page": 1,

"per\_page": 50

},

"items": [

{

"id": 1,

"username": "johndoe",

"first\_name": "John",

"last\_name": "Doe",

"full\_name": "John Doe",

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

"contact\_number": "1234567890",

"role": {

"id": 2,

"name": "Technician",

"description": "Technical staff",

"is\_super\_user": false

},

"environment": {

"id": 1,

"name": "Default",

"description": "Default environment"

}

},

{ ... }

]

}

**Error (500):**

JSON

{

"error": "Internal server error",

"details": "<error\_details>"

}

**GET /api/users/compact-list**

Get a list of all users with essential details (no permissions).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| include\_deleted | boolean | Optional | Include soft-deleted users (Admin only). Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

[

{

"id": 1,

"username": "johndoe",

"first\_name": "John",

"last\_name": "Doe",

"full\_name": "John Doe",

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

"contact\_number": "1234567890",

"role": {

"id": 2,

"name": "Technician",

"description": "Technical staff",

"is\_super\_user": false

},

"environment": {

"id": 1,

"name": "Default",

"description": "Default environment"

}

},

{ ... }

]

**Error (500):**

JSON

{

"error": "Internal server error | Database error: <details>"

}

**GET /api/users/byRole/{role\_id}**

Get users belonging to a specific role.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| role\_id | integer | Required | ID of the role. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

[

{

"id": 1,

"username": "johndoe",

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

"first\_name": "John",

"last\_name": "Doe",

"contact\_number": "1234567890",

"role": { ... },

"environment": { ... }

},

{ ... }

]

**Error (500):**

JSON

{

"error": "Internal server error"

}

**GET /api/users/byEnvironment/{environment\_id}**

Get users belonging to a specific environment.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| environment\_id | integer | Required | ID of the environment. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

[

{

"id": 1,

"username": "johndoe",

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

...

},

{ ... }

]

**Error (403):**

JSON

{

"error": "Unauthorized access to environment"

}

**GET /api/users/search**

Search for users based on criteria (Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| id | integer | Optional | Search by user ID. |
| username | string | Optional | Search by username (partial match). |
| role\_id | integer | Optional | Search by role ID. |
| environment\_id | integer | Optional | Search by environment ID. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

[

{

"id": 1,

"username": "johndoe",

... (full user details)

},

{ ... }

]

**Error (403):**

JSON

{

"error": "Unauthorized"

}

**GET /api/users/{user\_id}**

Get details for a specific user.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| user\_id | integer | Required | ID of the user. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"id": 1,

"username": "johndoe",

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

"first\_name": "John",

"last\_name": "Doe",

"contact\_number": "1234567890",

"role": { ... },

"environment": { ... },

"permissions": [ "perm1", "perm2", ...],

"is\_deleted": false,

"created\_at": "...",

"updated\_at": "..."

}

**Error (404):**

JSON

{

"error": "User not found"

}

**PUT /api/users/{user\_id}**

Update details for a specific user.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| user\_id | integer | Required | ID of the user to update. |

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| first\_name | string | Optional | User's first name. |
| last\_name | string | Optional | User's last name. |
| email | string | Optional | User's email address (must be unique). |
| contact\_number | string | Optional | User's contact phone number. |
| password | string | Optional | New password (min 8 chars). |
| username | string | Optional | Username (Admin only, must be unique, min 4 chars). |
| role\_id | integer | Optional | ID of the role to assign (Admin only). |
| environment\_id | integer | Optional | ID of the environment to assign (Admin only). |

**Example:**

JSON

{

"first\_name": "Jonathan",

"email": "jonathan.doe@example.com",

"password": "newSecurePassword456"

}

**Responses:**

**Success (200):**

JSON

{

"message": "User updated successfully",

"user": { ... (updated user details) }

}

**Error (400):**

JSON

{

"error": "Username already exists | Email already exists | Invalid role\_id | ..."

}

**DELETE /api/users/{user\_id}**

Soft delete a user and associated data.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| user\_id | integer | Required | ID of the user to delete. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"message": "User and all associated data deleted successfully",

"deleted\_items": {

"user": 1,

"forms\_created": 0,

"form\_submissions": 5,

"answers\_submitted": 25,

"attachments": 3

}

}

**Error (404):**

JSON

{

"error": "User not found"

}

**GET /api/users/current**

Get details of the currently authenticated user.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"id": 1,

"username": "johndoe",

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

... (full user details including permissions)

}

**Error (404):**

JSON

{

"error": "User not found"

}

**POST /api/roles**

Create a new role (Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| name | string | Required | Name of the role (must be unique). |
| description | string | Optional | Description of the role. |
| is\_super\_user | boolean | Optional | Whether this is a super user role (default: false). |

**Example:**

JSON

{

"name": "Editor",

"description": "Can edit content",

"is\_super\_user": false

}

**Responses:**

**Success (201):**

JSON

{

"message": "Role created successfully",

"role": {

"id": 3,

"name": "Editor",

"description": "Can edit content",

"is\_super\_user": false,

"is\_deleted": false,

"created\_at": "...",

"updated\_at": "..."

}

}

**Error (400):**

JSON

{

"error": "Name is required | Role name already exists"

}

**GET /api/roles**

Get a list of roles. Non-admins cannot see super user roles.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

[

{

"id": 1,

"name": "Admin",

"description": "Full system administrator",

"is\_super\_user": true,

...

},

{

"id": 2,

"name": "Technician",

"description": "Technical staff",

"is\_super\_user": false,

...

}

]

**Error (500):**

JSON

{

"error": "Internal server error"

}

**GET /api/roles/batch**

Get a paginated list of roles.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| page | integer | Optional | Page number. Default: 1. |
| per\_page | integer | Optional | Items per page. Default: 50. |
| include\_deleted | boolean | Optional | Include soft-deleted roles (Admin only). Default: false. |
| is\_super\_user | boolean | Optional | Filter by super user status. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"metadata": { ... },

"items": [

{

"id": 1,

"name": "Admin",

...

},

{ ... }

]

}

**Error (500):**

JSON

{

"error": "Internal server error",

"details": "<error\_details>"

}

**GET /api/roles/{role\_id}**

Get details for a specific role.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| role\_id | integer | Required | ID of the role. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"id": 2,

"name": "Technician",

"description": "Technical staff",

"is\_super\_user": false,

...

}

**Error (404):**

JSON

{

"error": "Role not found"

}

**PUT /api/roles/{role\_id}**

Update a role (Admin only). Cannot modify the main Admin role.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| role\_id | integer | Required | ID of the role to update. |

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| name | string | Optional | New name for the role (must be unique). |
| description | string | Optional | New description for the role. |
| is\_super\_user | boolean | Optional | Set super user status. |

**Example:**

JSON

{

"description": "Staff performing technical duties"

}

**Responses:**

**Success (200):**

JSON

{

"message": "Role updated successfully",

"role": { ... (updated role details) }

}

**Error (403):**

JSON

{

"error": "Cannot modify the main administrator role"

}

**DELETE /api/roles/{role\_id}**

Soft delete a role (Admin only). Cannot delete the main Admin role or roles with active users.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| role\_id | integer | Required | ID of the role to delete. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"message": "Role and associated permissions deleted successfully"

}

**Error (400):**

JSON

{

"error": "Cannot delete role with active users",

"role": { ... },

"active\_users": {

"count": 2,

"users": [ { ... }, { ... } ]

},

"suggestion": "Please reassign or deactivate these users before deleting this role"

}

**DELETE /api/roles/{role\_id}/permissions/{permission\_id}**

Remove a specific permission from a role (Admin only). Cannot modify the main Admin role.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| role\_id | integer | Required | ID of the role. |
| permission\_id | integer | Required | ID of the permission to remove. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"message": "Permission removed from role successfully"

}

**Error (400):**

JSON

{

"error": "Failed to remove permission from role"

}

**POST /api/forms**

Create a new form.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| title | string | Required | Title of the form. |
| description | string | Optional | Description of the form. |
| is\_public | boolean | Optional | Whether the form is public (default: false). |
| user\_id | integer | Optional | ID of the user to assign as creator (Admin only). Defaults to current user. |

**Example:**

JSON

{

"title": "Daily Safety Check",

"description": "Checklist for daily equipment safety",

"is\_public": false

}

**Responses:**

**Success (201):**

JSON

{

"message": "Form created successfully",

"form": {

"id": 1,

"title": "Daily Safety Check",

"description": "Checklist for daily equipment safety",

"is\_public": false,

"created\_by": { ... },

"questions": [],

"is\_deleted": false,

"created\_at": "...",

"updated\_at": "..."

},

"form\_creator": "current\_username"

}

**Error (400):**

JSON

{

"error": "Missing required fields | Form title already exists | ..."

}

**GET /api/forms**

Get a list of forms accessible to the user.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

[

{

"id": 1,

"title": "Daily Safety Check",

"description": "...",

"is\_public": false,

"created\_by": { ... },

"questions": [ { ... }, ... ],

...

},

{ ... }

]

**Error (500):**

JSON

{

"error": "Internal server error"

}

**GET /api/forms/batch**

Get a paginated list of forms with filtering options.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| page | integer | Optional | Page number. Default: 1. |
| per\_page | integer | Optional | Items per page. Default: 50. |
| include\_deleted | boolean | Optional | Include soft-deleted forms (Admin only). Default: false. |
| is\_public | boolean | Optional | Filter by public status. |
| user\_id | integer | Optional | Filter by creator user ID. |
| environment\_id | integer | Optional | Filter by environment ID (Admin only). |
| only\_editable | boolean | Optional | Return only forms editable by the user. Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"metadata": { ... },

"items": [

{

"id": 1,

"title": "Daily Safety Check",

... (compact form details)

},

{ ... }

]

}

**Error (500):**

JSON

{

"error": "Internal server error",

"details": "<error\_details>"

}

**GET /api/forms/{form\_id}**

Get details for a specific form.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_id | integer | Required | ID of the form. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"id": 1,

"title": "Daily Safety Check",

"description": "...",

"is\_public": false,

"created\_by": { ... },

"questions": [

{

"id": 1,

"form\_question\_id": 101,

"text": "Is equipment powered off?",

"type": "checkbox",

"order\_number": 1,

"remarks": null,

"possible\_answers": [ { "id": 1, "form\_answer\_id": 201, "value": "Yes" }, { "id": 2, "form\_answer\_id": 202, "value": "No" } ]

},

{ ... }

],

"is\_deleted": false,

"created\_at": "...",

"updated\_at": "..."

}

**Error (404):**

JSON

{

"error": "Form not found"

}

**POST /api/forms/{form\_id}/questions**

Add multiple questions to an existing form.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_id | integer | Required | ID of the form to add questions to. |

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| questions | array | Required | List of question objects to add. |

**Example:**

JSON

{

"questions": [

{ "question\_id": 15, "order\_number": 3 },

{ "question\_id": 22, "order\_number": 4 }

]

}

**Responses:**

**Success (200):**

JSON

{

"message": "Questions added successfully",

"form": { ... (updated form details including new questions) }

}

**Error (400):**

JSON

{

"error": "Questions are required | Invalid question ID | ..."

}

**POST /api/form-submissions**

Submit answers for a specific form.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json or multipart/form-data if signatures are included |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON (potentially multipart/form-data) **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_id | integer | Required | ID of the form being submitted. |
| submitted\_at | string | Optional | Timestamp of submission in ISO format (YYYY-MM-DDTHH:MM:SS). Defaults to server time. |
| answers | array | Required | List of answers submitted. |

**Example:**

JSON

{

"form\_id": 1,

"answers": [

{

"question\_id": 5,

"answer\_text": "Yes",

"question\_type\_text": "checkbox"

},

{

"question\_id": 6,

"answer\_text": "Checked all safety guards.",

"question\_type\_text": "text"

},

{

"question\_id": 7,

"answer\_text": "Signature data or reference",

"question\_type\_text": "signature",

"is\_signature": true

}

]

}

**Responses:**

**Success (201):**

JSON

{

"message": "Form submitted successfully",

"submission": {

"id": 101,

"form\_id": 1,

"submitted\_by": "current\_username",

"submitted\_at": "...",

"answers\_submitted": [ ... ],

"attachments": [ ... ],

...

}

}

**Error (400):**

JSON

{

"error": "form\_id is required | Invalid answer data | ..."

}

**POST /api/answers-submitted**

Create a single submitted answer record. Typically used internally by form submission.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_submission\_id | integer | Required | ID of the parent form submission. |
| question\_text | string | Required | Text of the question at the time of submission. |
| question\_type\_text | string | Required | Type of the question (e.g., 'text', 'table'). |
| answer\_text | string | Required | Answer provided by the user. |
| column | integer | Conditional | Column index (required if question\_type\_text is 'table'). |
| row | integer | Conditional | Row index (required if question\_type\_text is 'table'). |
| cell\_content | string | Optional | Content for table cells. |

**Example:**

JSON

{

"form\_submission\_id": 101,

"question\_text": "Is equipment powered off?",

"question\_type\_text": "checkbox",

"answer\_text": "Yes"

}

**Responses:**

**Success (201):**

JSON

{

"message": "Answer submitted successfully",

"answer\_submitted": {

"id": 501,

"form\_submission\_id": 101,

"question": "Is equipment powered off?",

"question\_type": "checkbox",

"answer": "Yes",

"column": null,

"row": null,

"cell\_content": null,

...

}

}

**Error (400):**

JSON

{

"error": "Missing required fields | Column and row are required for table-type questions | ..."

}

**POST /api/answers-submitted/bulk**

Create multiple submitted answer records for a single form submission. Typically used internally.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_submission\_id | integer | Required | ID of the parent form submission. |
| submissions | array | Required | List of submitted answers. |

**Example:**

JSON

{

"form\_submission\_id": 101,

"submissions": [

{ "question\_text": "Q1?", "question\_type\_text": "text", "answer\_text": "Ans1" },

{ "question\_text": "Q2?", "question\_type\_text": "table", "answer\_text": "CellVal", "column": 0, "row": 1, "cell\_content": "Cell Value" }

]

}

**Responses:**

**Success (201):**

JSON

{

"message": "Answers submitted successfully",

"submissions": [ { ... }, { ... } ]

}

**Error (400):**

JSON

{

"error": "Missing required fields: form\_submission\_id and submissions | Each submission must contain required fields | ..."

}

**POST /api/attachments**

Upload an attachment (file or signature) linked to a form submission.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | multipart/form-data |

**URL Parameters:**

N/A

**Request Body:**

Format: Form Data **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_submission\_id | integer | Required | ID of the form submission to link to. |
| file | file | Required | The file to upload. |
| is\_signature | boolean | Optional | Set to 'true' if the file is a signature image. Default: false. |
| answer\_submitted\_id | integer | Conditional | ID of the AnswerSubmitted record this signature relates to (if is\_signature=true). |
| signature\_position | string | Optional | Position/identifier for the signature (e.g., 'Completed By'). |
| signature\_author | string | Optional | Name of the person who signed. |

**Example:** N/A (Form data)

**Responses:**

**Success (201):**

JSON

{

"message": "Attachment created successfully",

"attachment": {

"id": 301,

"form\_submission\_id": 101,

"filename": "signature\_john\_doe.png",

"file\_path": "uploads/submission\_101/signature\_john\_doe\_timestamp.png",

"file\_type": "image/png",

"is\_signature": true,

"answer\_submitted\_id": 505,

"signature\_position": "Completed By",

"signature\_author": "John Doe",

...

}

}

**Error (400):**

JSON

{

"error": "form\_submission\_id is required | No file provided | Invalid file type | ..."

}

**POST /api/attachments/bulk**

Upload multiple attachments for a single form submission.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | multipart/form-data |

**URL Parameters:**

N/A

**Request Body:**

Format: Form Data **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_submission\_id | integer | Required | ID of the form submission. |
| file&lt;N&gt; | file | Required | File upload field (e.g., file1, file2). |
| is\_signature&lt;N&gt; | boolean | Optional | Indicates if file&lt;N&gt; is a signature. |
| answer\_submitted\_id&lt;N&gt; | integer | Conditional | Answer ID for signature&lt;N&gt;. |
| signature\_position&lt;N&gt; | string | Optional | Position for signature&lt;N&gt;. |
| signature\_author&lt;N&gt; | string | Optional | Author for signature&lt;N&gt;. |

**Example:** N/A (Form data with multiple file fields like file1, file2, etc.)

**Responses:**

**Success (201):**

JSON

{

"message": "Attachments created successfully",

"attachments": [ { ... }, { ... } ]

}

**Error (400):**

JSON

{

"error": "form\_submission\_id is required | No files provided | ..."

}

**GET /api/attachments/{attachment\_id}**

Download a specific attachment file.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| attachment\_id | integer | Required | ID of the attachment to download. |

**Request Body:**

N/A

**Responses:**

**Success (200):** File content (e.g., image/png, application/pdf) **Error (404):**

JSON

{

"error": "File not found"

}

**GET /api/export/form/{form\_id}**

Export a form structure (questions, possible answers) to PDF or DOCX.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_id | integer | Required | ID of the form to export. |

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| format | string | Optional | Export format ('PDF' or 'DOCX'). Default: PDF. |
| page\_size | string | Optional | Page size ('A4', 'LETTER', 'LEGAL'). Default: LETTER. |
| margin\_top | float | Optional | Top margin in inches. Default: 1.0. |
| margin\_bottom | float | Optional | Bottom margin in inches. Default: 1.0. |
| margin\_left | float | Optional | Left margin in inches. Default: 1.0. |
| margin\_right | float | Optional | Right margin in inches. Default: 1.0. |
| line\_spacing | float | Optional | Line spacing multiplier. Default: 1.15. |
| font\_size | integer | Optional | Base font size in points. Default: 12. |
| logo\_path | string | Optional | Server path to a logo image (PNG/JPG). |
| signature\_count | integer | Optional | Number of signature lines (1-5). Default: 2. |
| signature&lt;N&gt;\_title | string | Optional | Title for signature line N (e.g., 'signature1\_title'). |
| signature&lt;N&gt;\_name | string | Optional | Pre-filled name for signature line N. |
| signature&lt;N&gt;\_date | boolean | Optional | Include date field for signature line N. Default: true. |
| signature\_space\_before | float | Optional | Space before signature section (points). Default: 20. |
| signature\_space\_between | float | Optional | Space between signatures (points). Default: 10. |
| signature\_space\_date | float | Optional | Space after date field (points). Default: 5. |
| signature\_space\_after | float | Optional | Space after signature section (points). Default: 20. |

**Request Body:**

N/A

**Responses:**

**Success (200):** File content (application/pdf or application/vnd.openxmlformats-officedocument.wordprocessingml.document) **Error (400):**

JSON

{

"error": "Invalid format | Logo file not found | Error generating export: <details>"

}

**GET /api/export/formats**

Get the list of supported export formats.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"formats": ["PDF", "DOCX"],

"default": "PDF"

}

**Error:** N/A

**GET /api/export/parameters**

Get details about available export formatting parameters.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"parameters": {

"page\_size": { "description": "...", "values": ["A4", "LETTER", "LEGAL"], "default": "LETTER" },

"margin\_top": { "description": "...", "range": [0.1, 3.0], "default": 1.0 },

... (all parameters described)

}

}

**Error:** N/A

**GET /api/export/form/{form\_id}/preview-params**

Get formatting parameters and example URLs for exporting a specific form.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| form\_id | integer | Required | ID of the form. |

**Query Parameters:** Accepts same query parameters as /api/export/form/{form\_id} to show current values

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"form\_info": { "id": 1, "title": "...", "questions\_count": 5 },

"current\_parameters": { "format": "PDF", "page\_size": "LETTER", ... },

"available\_parameters": { ... (details of all params) },

"example\_urls": { "minimal": "...", "with\_basic\_formatting": "...", "complete": "..." },

"notes": [ ... ]

}

**Error (404):**

JSON

{

"error": "Form not found"

}

**GET /api/health/ping**

Simple check if the API is responsive.

**Headers:**

N/A

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"status": "pong",

"message": "Server is responsive"

}

**Error (500):**

JSON

{

"status": "error",

"message": "Server error"

}

**GET /api/health/status**

Get detailed health status including database connectivity.

**Headers:**

N/A

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):** # or 503 if unhealthy

JSON

{

"server\_status": "ok",

"database\_status": "ok",

"timestamp": "...",

"health\_status": "healthy"

}

**Error (500):**

JSON

{

"status": "error",

"message": "Server error"

}

**POST /api/cmms-configs**

Create a new JSON configuration file in the 'configs' subfolder (Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| filename | string | Required | Filename ending in .json. |
| content | object | Required | JSON content for the file. |

**Example:**

JSON

{

"filename": "settings.json",

"content": { "theme": "dark", "notifications": true }

}

**Responses:**

**Success (201):**

JSON

{

"message": "Configuration file created successfully",

"config": {

"filename": "settings.json",

"path": "configs/settings.json",

"content": { "theme": "dark", "notifications": true },

"created\_at": "...",

"updated\_at": "..."

}

}

**Error (400):**

JSON

{

"error": "Missing required fields | Only JSON files are supported | Filename already exists | ..."

}

**POST /api/cmms-configs/upload**

Upload a configuration file (places it in the 'configs' subfolder, Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | multipart/form-data |

**URL Parameters:**

N/A

**Request Body:**

Format: Form Data **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| file | file | Required | The configuration file to upload. |

**Example:** N/A (Form data)

**Responses:**

**Success (201):**

JSON

{

"message": "Configuration file uploaded successfully",

"config": { ... (details of the uploaded config file) }

}

**Error (400):**

JSON

{

"error": "No file provided | No selected file | File type not allowed | ..."

}

**GET /api/cmms-configs/files**

List all files within the main 'uploads' directory and its subdirectories (e.g., configs, submission attachments).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"message": "Files retrieved successfully",

"files": [

{

"filename": "settings.json",

"path": "configs/settings.json",

"full\_path": "/path/to/app/uploads/configs/settings.json",

"size": 123,

"size\_formatted": "123 B",

"mime\_type": "application/json",

"created\_at": "...",

"modified\_at": "...",

"directory": "configs"

},

{

"filename": "report.pdf",

"path": "submission\_101/report.pdf",

"full\_path": "/path/to/app/uploads/submission\_101/report.pdf",

"size": 102400,

"size\_formatted": "100.0 KB",

"mime\_type": "application/pdf",

"created\_at": "...",

"modified\_at": "...",

"directory": "submission\_101"

},

{ ... }

],

"total\_files": 2

}

**Error (500):**

JSON

{

"error": "Internal server error"

}

**GET /api/cmms-configs/file/{filepath}**

Download a specific file from the 'uploads' directory structure using its relative path.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| filepath | string | Required | Relative path to the file within the uploads directory (e.g., 'configs/settings.json' or 'submission\_101/report.pdf'). |

**Request Body:**

N/A

**Responses:**

**Success (200):** File content **Error (404):**

JSON

{

"error": "File not found | Invalid path"

}

**GET /api/cmms-configs/{filename}**

Load the content of a configuration file from the 'configs' subfolder.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| filename | string | Required | Name of the config file in the 'configs' directory. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"filename": "settings.json",

"path": "configs/settings.json",

"file\_type": "application/json",

"content": { "theme": "dark", "notifications": true },

"created\_at": "...",

"updated\_at": "..."

}

**Error (404):**

JSON

{

"error": "Configuration file not found"

}

**PUT /api/cmms-configs/configs/{filename}**

Update the content of a JSON configuration file in the 'configs' subfolder (Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| filename | string | Required | Name of the config file to update. |

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| content | object | Required | New JSON content for the file. |

**Example:**

JSON

{

"content": { "theme": "light", "notifications": false, "new\_setting": 123 }

}

**Responses:**

**Success (200):**

JSON

{

"message": "Configuration file updated successfully",

"config": { ... (updated config details) }

}

**Error (400):**

JSON

{

"error": "Content is required | File is not a JSON file | Error updating file | ..."

}

**PUT /api/cmms-configs/{filename}/rename**

Rename a configuration file within the 'configs' subfolder (Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| filename | string | Required | Current name of the config file. |

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| new\_filename | string | Required | New filename (must end in .json). |

**Example:**

JSON

{

"new\_filename": "app\_settings.json"

}

**Responses:**

**Success (200):**

JSON

{

"message": "Configuration file renamed successfully",

"config": { ... (details with new filename) }

}

**Error (400):**

JSON

{

"error": "New filename is required | New filename must end with .json | File not found | New filename already exists | ..."

}

**DELETE /api/cmms-configs/{filename}**

Delete a configuration file from the 'configs' subfolder (Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| filename | string | Required | Name of the config file to delete. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"message": "Configuration file deleted successfully"

}

**Error (404):**

JSON

{

"error": "Configuration file not found"

}

**GET /api/cmms-configs/check**

Check if the main configuration file exists in the 'configs' subfolder.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

N/A

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"exists": true,

"metadata": {

"filename": "cmms\_config.json",

"path": "configs/cmms\_config.json",

"size": 512,

"modified\_at": "..."

},

"message": "Configuration file found in configs folder"

}

**or exists: false**

**Error (500):**

JSON

{

"error": "Internal server error",

"details": "<error\_details>"

}

**GET /api/export\_submissions/{submission\_id}/pdf**

Export a specific form submission as a PDF document.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| submission\_id | integer | Required | ID of the form submission to export. |

**Request Body:**

N/A

**Responses:**

**Success (200):** File content (application/pdf) **Error (400):**

JSON

{

"error": "Submission not found | Error generating PDF | Unauthorized access"

}

**POST /api/export\_submissions/{submission\_id}/pdf/logo**

Export a form submission as PDF with custom header image and signature formatting.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | multipart/form-data |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| submission\_id | integer | Required | ID of the form submission. |

**Request Body:**

Format: Form Data **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| header\_image | file | Optional | Image file (PNG/JPG/JPEG) for the header. |
| header\_opacity | float | Optional | Opacity of the header image (0-100). Default: 100. |
| header\_size | float | Optional | Size percentage for header image (maintains aspect ratio). Default: original size (100%). |
| header\_width | float | Optional | Explicit width in pixels for header image (overrides header\_size). |
| header\_height | float | Optional | Explicit height in pixels for header image (overrides header\_size). |
| header\_alignment | string | Optional | Alignment of header image ('left', 'center', 'right'). Default: center. |
| signatures\_size | float | Optional | Size percentage for signature images. Default: 100. |
| signatures\_alignment | string | Optional | Layout for signatures ('vertical', 'horizontal'). Default: vertical. |

**Example:** N/A (Form data)

**Responses:**

**Success (200):** File content (application/pdf) **Error (400):**

JSON

{

"error": "Submission not found | Header image must be PNG or JPEG | Header opacity must be a number | ..."

}

**GET /api/counts**

Get counts for all entity types accessible to the user.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| include\_deleted | boolean | Optional | Include counts of soft-deleted items (Admin only). Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"users": 15,

"roles": 4,

"permissions": 50,

"environments": 3,

"question\_types": 8,

"questions": 120,

"answers": 35,

"forms": 25,

"form\_submissions": 500,

"answers\_submitted": 2500,

"attachments": 150,

"role\_permissions": 100,

"form\_questions": 150,

"form\_answers": 60

}

**Error (500):**

JSON

{

"error": "Internal server error"

}

**GET /api/counts/{entity}**

Get the count for a specific entity type (e.g., users, forms, roles).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| entity | string | Required | The entity type plural name (e.g., 'users', 'forms'). |

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| include\_deleted | boolean | Optional | Include count of soft-deleted items (Admin only). Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"entity": "users",

"count": 15,

"include\_deleted": false

}

**Error (500):**

JSON

{

"error": "Internal server error"

}

**GET /api/entity\_basic/basic**

Get basic representations (id, name/title) of all entity types (Admin only).

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| page | integer | Optional | Page number. Default: 1. |
| per\_page | integer | Optional | Items per page. Default: 20. |
| include\_deleted | boolean | Optional | Include soft-deleted items. Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"metadata": { ... },

"items": [

{ "id": 1, "name": "Admin", "type": "roles" },

{ "id": 1, "title": "Daily Safety Check", "type": "forms" },

{ ... }

]

}

**Error (403):**

JSON

{

"error": "Unauthorized. Admin access required."

}

**GET /api/entity\_basic/{entity\_type}/basic**

Get basic representations (id, name/title) for a specific entity type.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| entity\_type | string | Required | Entity type (e.g., 'users', 'forms'). |

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| page | integer | Optional | Page number. Default: 1. |
| per\_page | integer | Optional | Items per page. Default: 20 (max 100). |
| include\_deleted | boolean | Optional | Include soft-deleted items (Admin only). Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"metadata": { ... },

"items": [

{ "id": 1, "name": "Daily Safety Check" },

{ "id": 2, "name": "Monthly Maintenance Report" },

{ ... }

]

}

**Error (400):**

JSON

{

"error": "Invalid entity type | Invalid pagination parameters"

}

**GET /api/entity\_basic/{entity\_type}/{entity\_id}/basic**

Get basic representation (id, name/title) for a specific entity instance.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| entity\_type | string | Required | Entity type (e.g., 'users', 'forms'). |
| entity\_id | integer | Required | ID of the entity. |

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| include\_deleted | boolean | Optional | Include if soft-deleted (Admin only). Default: false. |

**Request Body:**

N/A

**Responses:**

**Success (200):**

JSON

{

"id": 1,

"name": "Daily Safety Check"

}

**Error (404):**

JSON

{

"error": "Entity not found"

}

**POST /api/entity\_basic/{entity\_type}/basic/batch**

Get basic representations for multiple entities of the same type by their IDs.

**Headers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Purpose** | **Status** | **Example** |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

**Path Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| entity\_type | string | Required | Entity type (e.g., 'users', 'forms'). |

**Query Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| include\_deleted | boolean | Optional | Include soft-deleted items (Admin only). Default: false. |

**Request Body:**

Format: JSON **Fields:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Status** | **Description** |
| ids | array | Required | List of entity IDs. |

**Example:**

JSON

{

"ids": [1, 5, 12]

}

**Responses:**

**Success (200):**

JSON

{

"items": [

{ "id": 1, "name": "Daily Safety Check" },

{ "id": 5, "name": "Incident Report" },

{ "id": 12, "name": "Equipment Calibration" }

]

}

**Error (400):**

JSON

{

"error": "Invalid request format. Expected 'ids' list in JSON body"

}

**Report API Request Examples**

This document provides comprehensive examples for the /api/reports/generate endpoint, showcasing different report types, formats, and parameter combinations. These examples will help developers understand how to effectively use this API for generating various kinds of reports from the system.

**Basic Report Requests**

**Example 1: Basic Users Report (XLSX Default)**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "users"

}

This simple request generates a users report with default columns, no filters, and default sorting. The output is XLSX format (default).

**Example 2: Form Submissions with Column Selection**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "form\_submissions",

"columns": [

"id",

"form.title",

"submitted\_by",

"submitted\_at",

"form.creator.username",

"created\_at"

]

}

This request generates a form submissions report with only the specified columns.

**Example 3: Filtered Environment Report**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "environments",

"filters": [

{

"field": "name",

"operator": "like",

"value": "Production"

},

{

"field": "created\_at",

"operator": "gte",

"value": "2025-01-01T00:00:00Z"

}

]

}

This request filters environments with names containing "Production" created on or after January 1, 2025.

**Example 4: Sorted Role Permissions Report**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "role\_permissions",

"sort\_by": [

{

"field": "role.name",

"direction": "asc"

},

{

"field": "permission.entity",

"direction": "asc"

}

]

}

This request sorts role permissions by role name and permission entity in ascending order.

**Alternative Output Formats**

**Example 5: CSV Format Report**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "forms",

"output\_format": "csv",

"columns": [

"id",

"title",

"description",

"creator.username",

"is\_public",

"created\_at"

]

}

This request generates a forms report in CSV format instead of the default XLSX.

**Example 6: PDF Format Report**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "form\_submissions",

"output\_format": "pdf",

"report\_title": "Form Submissions Summary Report",

"filters": [

{

"field": "submitted\_at",

"operator": "between",

"value": ["2025-04-01T00:00:00Z", "2025-04-30T23:59:59Z"]

}

]

}

This request generates a PDF report for form submissions in April 2025 with a custom title.

**Example 7: DOCX Format Report**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "users",

"output\_format": "docx",

"filename": "active\_users\_report",

"filters": [

{

"field": "is\_deleted",

"operator": "eq",

"value": false

}

]

}

This request generates a Word document report for active (not deleted) users with a custom filename.

**Example 8: PPTX Format Report (Form Submissions)**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "form\_submissions",

"output\_format": "pptx",

"report\_title": "Monthly Submission Trends",

"include\_data\_table\_in\_ppt": true,

"max\_ppt\_table\_rows": 10

}

This request generates a PowerPoint presentation with submission trends and includes a data table with up to 10 rows.

**Advanced Report Requests**

**Example 9: Complex Filtered and Sorted Report**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "form\_submissions",

"columns": [

"id",

"form.title",

"submitted\_by",

"submitted\_at",

"answers.Safety Check Completed",

"answers.Equipment Condition",

"answers.Notes"

],

"filters": [

{

"field": "form.id",

"operator": "eq",

"value": 12

},

{

"field": "submitted\_at",

"operator": "between",

"value": ["2025-03-01T00:00:00Z", "2025-03-31T23:59:59Z"]

},

{

"field": "answers\_submitted.question",

"operator": "eq",

"value": "Safety Check Completed"

},

{

"field": "answers\_submitted.answer",

"operator": "eq",

"value": "Yes"

}

],

"sort\_by": [

{

"field": "submitted\_at",

"direction": "desc"

}

],

"output\_format": "xlsx",

"filename": "march\_safety\_checks\_report"

}

This complex request generates a report of March 2025 form submissions for form ID 12 where "Safety Check Completed" answer is "Yes", showing specific question answers and sorted by submission date.

**Example 10: Multi-Entity Report (All Tables)**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": "all",

"output\_format": "xlsx",

"filename": "full\_system\_data\_report"

}

This request generates a multi-sheet XLSX workbook containing data from all available entities, using default columns and settings for each.

**Example 11: Multi-Entity Specific Selection**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": ["users", "forms", "form\_submissions"],

"output\_format": "csv",

"filename": "user\_form\_activity"

}

This request generates a ZIP archive containing CSV files for users, forms, and form submissions data.

**Example 12: Multi-Entity with Sheet Name Customization**

POST /api/reports/generate HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

{

"report\_type": ["users", "environments"],

"output\_format": "xlsx",

"users\_sheet\_name": "System Users",

"environments\_sheet\_name": "Available Environments",

"filename": "system\_configuration",

"table\_options": {

"style": "Table Style Medium 2",

"banded\_rows": true

}

}

This request generates an XLSX workbook with customized sheet names and table styling.

**Database Schema Request**

**Example 13: Database Schema Information**

GET /api/reports/schema HTTP/1.1

Host: example.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

This request retrieves the database schema information (only accessible to administrators).

**GET /api/reports/schema**

Retrieves the database schema information for generating reports. This endpoint provides detailed metadata about available tables, columns, and relationships to help with report creation.

**Headers:**

| **Name** | **Purpose** | **Status** | **Example** |
| --- | --- | --- | --- |
| Authorization | Authentication token | Required | Bearer <JWT\_TOKEN> |
| Content-Type | Specify content type | Required | application/json |

**URL Parameters:**

N/A

**Request Body:**

Format: JSON **Fields:**

| **Name** | **Type** | **Status** | **Description** |
| --- | --- | --- | --- |
| report\_type | string | Optional | Type of report to get schema for (e.g., 'users', 'forms', 'submissions', 'all'). Default: 'all' |

**Example:**

{

"report\_type": "all"

}

**Responses:**

**Success (200):**

{

"database\_info": {

"application\_models": 14,

"name": "database\_name",

"total\_tables": 15,

"version": "PostgreSQL version info"

},

"model\_mapping": {

// Maps model names to table names

},

"tables": {

// Detailed table schema information including:

// - columns (name, type, nullable, etc.)

// - primary keys

// - foreign keys and relationships

// - row counts (active, deleted, total)

}

}

**Error (400):**

{

"error": "Invalid report type"

}

**Error (401):**

{

"error": "Unauthorized access"

}

**Error (403):**

{

"error": "Insufficient permissions, Admin role required"

}

**Notes:**

* This endpoint is intended for Admin users to understand the database structure for creating custom reports
* The schema information is useful for constructing advanced filters and column selections when using the POST /api/reports/generate endpoint
* Response includes comprehensive metadata about each table's structure, relationships, and row counts

This includes documentation for the main endpoints identified in the provided files. Additional endpoints related to permissions, environments, question types, answers, form questions, form answers, and role permissions follow similar patterns and are largely administrative or relate to setting up the structure of forms. Their specific parameters and responses can be inferred from the corresponding view files (permission\_views.py, environment\_views.py, etc.).