**API Endpoints Documentation**

A document covering all API endpoints (with what is sent and received) is mandatory.

**Table of Contents**

1. [Authentication Endpoints](#authentication-endpoints)
2. [Email Management Endpoints](#email-management-endpoints)
3. [Role Management Endpoints](#role-management-endpoints)
4. [User Management Endpoints](#user-management-endpoints)

**Authentication Endpoints**

**Base URL: /api/auth**

**1. Register User**

* **Endpoint**: POST /api/auth/register
* **Description**: Create a new user account
* **Authentication**: None required
* **Request Body**:

{

"firstname": "string",

"lastname": "string",

"email": "string",

"password": "string",

"roleId": "integer"

}

* **Response**:
  + **Success (200)**:
* {
* "message": "User registered successfully!"
* }
  + **Error (400)**:
* {
* "message": "Error: Email is already in use!"
* }

**2. Login User**

* **Endpoint**: POST /api/auth/login
* **Description**: Authenticate user and return JWT token
* **Authentication**: None required
* **Request Body**:

{

"email": "string",

"password": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "token": "string",
* "type": "Bearer",
* "id": "integer",
* "email": "string",
* "firstname": "string",
* "lastname": "string",
* "role": "string"
* }
  + **Error (401)**:
* {
* "message": "Error: Invalid email or password!"
* }

**3. Refresh Token**

* **Endpoint**: POST /api/auth/refresh
* **Description**: Refresh JWT token
* **Authentication**: None required
* **Request Body**:

{

"token": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "token": "string",
* "type": "Bearer"
* }
  + **Error (401)**:
* {
* "message": "Invalid refresh token!"
* }

**4. Logout User**

* **Endpoint**: POST /api/auth/logout
* **Description**: Logout user (frontend clears token)
* **Authentication**: None required
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "message": "User logged out successfully!"
* }

**5. Validate Token**

* **Endpoint**: POST /api/auth/validate
* **Description**: Validate JWT token and return user info
* **Authentication**: None required
* **Request Body**:

{

"token": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "id": "integer",
* "email": "string",
* "firstname": "string",
* "lastname": "string",
* "role": "string",
* "active": "boolean"
* }
  + **Error (401)**:
* {
* "message": "Invalid token!"
* }

**6. Change Password**

* **Endpoint**: POST /api/auth/change-password
* **Description**: Change the current user's password
* **Authentication**: Bearer Token required
* **Request Body**:

{

"oldPassword": "string",

"newPassword": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "message": "Password changed successfully!"
* }
  + **Error (400/401)**:
* {
* "message": "Error changing password: [error details]"
* }

**7. Forgot Password**

* **Endpoint**: POST /api/auth/forgot-password
* **Description**: Request password reset
* **Authentication**: None required
* **Request Body**:

{

"email": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "message": "Password reset instructions sent to your email!"
* }

**Email Management Endpoints**

**Base URL: /api/emails**

**1. Send Test Email**

* **Endpoint**: POST /api/emails/test
* **Description**: Send a test email to verify email configuration
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**:

{

"email": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "message": "Test email sent successfully to [email]",
* "status": "success"
* }

**2. Send Simple Email**

* **Endpoint**: POST /api/emails/simple
* **Description**: Send a simple text email
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**:

{

"to": "string",

"subject": "string",

"text": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "message": "Simple email sent successfully",
* "status": "success"
* }

**3. Send Custom Email**

* **Endpoint**: POST /api/emails/custom
* **Description**: Send a custom email using template
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**:

{

"to": "string",

"subject": "string",

"templateName": "string",

"variables": {

"key1": "value1",

"key2": "value2"

}

}

* **Response**:
  + **Success (200)**:
* {
* "message": "Custom email sent successfully",
* "status": "success"
* }

**4. Send Bulk Email**

* **Endpoint**: POST /api/emails/bulk
* **Description**: Send bulk emails to multiple recipients
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**:

{

"recipients": ["string", "string"],

"subject": "string",

"templateName": "string",

"variables": {

"key1": "value1",

"key2": "value2"

}

}

* **Response**:
  + **Success (200)**:
* {
* "message": "Bulk emails sent successfully to [count] recipients",
* "status": "success"
* }

**5. Send Admin Notification**

* **Endpoint**: POST /api/emails/admin-notification
* **Description**: Send notification to all administrators
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**:

{

"subject": "string",

"message": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "message": "Admin notification sent successfully",
* "status": "success"
* }

**6. Get Email Statistics**

* **Endpoint**: GET /api/emails/statistics
* **Description**: Get email sending statistics
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "totalEmailsSent": "long",
* "totalEmailsFailed": "long",
* "totalEmailsToday": "long"
* }

**7. Resend Welcome Email**

* **Endpoint**: POST /api/emails/resend-welcome/{userId}
* **Description**: Resend welcome email to specific user
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: userId (integer)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "message": "Welcome email resent successfully",
* "status": "success"
* }

**Role Management Endpoints**

**Base URL: /api/roles**

**1. Get All Roles**

* **Endpoint**: GET /api/roles
* **Description**: Get all roles
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* [
* {
* "idRole": "integer",
* "role": "string"
* }
* ]

**2. Get Role by ID**

* **Endpoint**: GET /api/roles/{id}
* **Description**: Get role by ID
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: id (integer)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "idRole": "integer",
* "role": "string"
* }
  + **Not Found (404)**: Empty response

**3. Get Role by Name**

* **Endpoint**: GET /api/roles/name/{roleName}
* **Description**: Get role by name
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: roleName (string)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "idRole": "integer",
* "role": "string"
* }

**4. Create Role**

* **Endpoint**: POST /api/roles
* **Description**: Create new role
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**:

{

"role": "string"

}

* **Response**:
  + **Success (201)**:
* {
* "idRole": "integer",
* "role": "string"
* }
  + **Error (400)**:
* {
* "message": "Error: Role name already exists!",
* "status": "error"
* }

**5. Update Role**

* **Endpoint**: PUT /api/roles/{id}
* **Description**: Update role
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: id (integer)
* **Request Body**:

{

"role": "string"

}

* **Response**:
  + **Success (200)**:
* {
* "idRole": "integer",
* "role": "string"
* }

**6. Delete Role**

* **Endpoint**: DELETE /api/roles/{id}
* **Description**: Delete role
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: id (integer)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "message": "Role deleted successfully",
* "status": "success"
* }
  + **Error (400)**:
* {
* "message": "Cannot delete role: It has associated users",
* "status": "error"
* }

**7. Get Users Count by Role**

* **Endpoint**: GET /api/roles/{id}/users/count
* **Description**: Get users count for a specific role
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: id (integer)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* 5

**8. Get Role Statistics**

* **Endpoint**: GET /api/roles/statistics
* **Description**: Get role statistics
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "totalRoles": "long",
* "rolesWithUsers": "long",
* "emptyRoles": "long"
* }

**9. Get All Role Names**

* **Endpoint**: GET /api/roles/names
* **Description**: Get all role names (for dropdowns)
* **Authentication**: Bearer Token required (ADMIN or GERANT role)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* ["ADMIN", "CLIENT", "GERANT"]

**10. Create Default Roles**

* **Endpoint**: POST /api/roles/init-defaults
* **Description**: Create default roles
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "message": "Default roles created successfully",
* "status": "success"
* }

**11. Validate Role Name**

* **Endpoint**: GET /api/roles/validate/{roleName}
* **Description**: Check if role name is valid
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: roleName (string)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* true

**User Management Endpoints**

**Base URL: /api/users**

**1. Get All Users**

* **Endpoint**: GET /api/users
* **Description**: Get all users
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* [
* {
* "idUser": "integer",
* "firstname": "string",
* "lastname": "string",
* "email": "string",
* "active": "boolean",
* "role": {
* "idRole": "integer",
* "role": "string"
* }
* }
* ]

**2. Get User by ID**

* **Endpoint**: GET /api/users/{id}
* **Description**: Get user by ID
* **Authentication**: Bearer Token required (ADMIN role or current user)
* **Path Parameters**: id (integer)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* {
* "idUser": "integer",
* "firstname": "string",
* "lastname": "string",
* "email": "string",
* "active": "boolean",
* "role": {
* "idRole": "integer",
* "role": "string"
* }
* }

**3. Get User by Email**

* **Endpoint**: GET /api/users/email/{email}
* **Description**: Get user by email
* **Authentication**: Bearer Token required (ADMIN role or authenticated user's own email)
* **Path Parameters**: email (string)
* **Request Body**: None
* **Response**: Same as Get User by ID

**4. Create User**

* **Endpoint**: POST /api/users
* **Description**: Create new user
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**:

{

"firstname": "string",

"lastname": "string",

"email": "string",

"password": "string",

"role": {

"idRole": "integer"

}

}

* **Response**:
  + **Success (201)**:
* {
* "idUser": "integer",
* "firstname": "string",
* "lastname": "string",
* "email": "string",
* "active": "boolean",
* "role": {
* "idRole": "integer",
* "role": "string"
* }
* }

**5. Update User**

* **Endpoint**: PUT /api/users/{id}
* **Description**: Update user
* **Authentication**: Bearer Token required (ADMIN role or current user)
* **Path Parameters**: id (integer)
* **Request Body**: Same as Create User
* **Response**: Same as Create User

**6. Deactivate User**

* **Endpoint**: PUT /api/users/{id}/deactivate
* **Description**: Deactivate user (soft delete)
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: id (integer)
* **Request Body**: None
* **Response**: Same as Get User by ID

**7. Activate User**

* **Endpoint**: PUT /api/users/{id}/activate
* **Description**: Activate user
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: id (integer)
* **Request Body**: None
* **Response**: Same as Get User by ID

**8. Delete User**

* **Endpoint**: DELETE /api/users/{id}
* **Description**: Delete user permanently
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: id (integer)
* **Request Body**: None
* **Response**:
  + **Success (200)**:
* "User deleted successfully"

**9. Get Users by Role**

* **Endpoint**: GET /api/users/role/{roleId}
* **Description**: Get users by role
* **Authentication**: Bearer Token required (ADMIN role)
* **Path Parameters**: roleId (integer)
* **Request Body**: None
* **Response**: Same as Get All Users

**10. Get Active Users**

* **Endpoint**: GET /api/users/active
* **Description**: Get active users only
* **Authentication**: Bearer Token required (ADMIN role)
* **Request Body**: None
* **Response**: Same as Get All Users

**11. Change User Password**

* **Endpoint**: PUT /api/users/{id}/password
* **Description**: Change user password
* **Authentication**: Bearer Token required (ADMIN role or current user)
* **Path Parameters**: id (integer)
* **Request Body**:

{

"oldPassword": "string",

"newPassword": "string"

}

* **Response**:
  + **Success (200)**:
* "Password changed successfully"

**Authentication Notes**

* **Bearer Token**: Include in the Authorization header as Authorization: Bearer <token>
* **Role-based Access**:
  + ADMIN: Full access to all endpoints
  + GERANT: Limited access (specified in individual endpoints)
  + CLIENT: Basic access to own data only
* **Current User Access**: Users can access their own data even without ADMIN role

**Error Response Format**

All error responses follow this general format:

{

"message": "Error description",

"status": "error"

}

Common HTTP status codes:

* 200: Success
* 201: Created
* 400: Bad Request
* 401: Unauthorized
* 403: Forbidden
* 404: Not Found
* 500: Internal Server Error