**✅ Endpoint: Login**

**URL:**

POST /api/login/

**Description:**

Authenticates a user and returns a pair of JWT access and refresh tokens along with user metadata including roles and profile type (SuperAdmin, SchoolAdmin, Teacher, or Student).

**Authentication Required:**

❌ No (this is the login endpoint)

**Permissions:**

Open to all users with valid credentials

**Request Body:**

{

"username": "example\_user",

"password": "secure\_password123"

}

**Response Example:**

{

"refresh": "eyJ0eXAiOiJKV1QiLCJhbGciOi...",

"access": "eyJ0eXAiOiJKV1QiLCJhbGciOi...",

"id": 3,

"username": "example\_user",

"email": "user@example.com",

"user\_roles": [

{

"id": "e7d2d24e-...",

"role": {

"id": "7e8a8a4e-...",

"name": "Student",

"description": "Student role"

}

}

],

"student": {

"student\_id": "cd1e7650-...",

"school\_id": "a3fc23f1-...",

"school": "Bright Future College",

"first\_name": "John",

"last\_name": "Doe",

"admission\_number": 1001,

"profile\_picture\_path": null

}

}

**Error Responses:**

* **401 Unauthorized**

{

"detail": "No active account found with the given credentials"

}

**✅ Endpoint: Logout**

**URL:**

POST /api/logout/

**Description:**

Logs out the currently authenticated user by blacklisting the provided JWT refresh token.

**Authentication Required:**

✅ Yes (JWT access token required in the header)

**Permissions:**

Any authenticated user

**Request Headers:**

Authorization: Bearer <your-access-token>

**Request Body:**

{

"refresh\_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."

}

**Response Example (Success):**

{

"message": "Logged out successfully"

}

**Error Responses:**

* **400 Bad Request – Missing Token**

{

"error": "Refresh token is required"

}

* **400 Bad Request – Invalid Token**

{

"error": "Invalid token"

}

**Notes:**

* This endpoint blacklists the refresh token only. Access tokens will naturally expire.
* Logout is handled in LogoutView and uses rest\_framework\_simplejwt.tokens.RefreshToken.

Awesome — here’s the documentation for the next endpoint.

**✅ Endpoint: List All Roles**

**URL:**

GET /api/roles/

**Description:**

Retrieves a list of all user roles available in the system. Useful for role selection during registration or user management.

**Authentication Required:**

✅ Yes (JWT access token required)

**Permissions:**

Authenticated users only

**Request Headers:**

Authorization: Bearer <your-access-token>

**Response Example:**

[

{

"id": "8f5d7c6e-94e6-4dd5-b879-32b7a4f91b0d",

"name": "Super Admin",

"description": "Has global access and can manage the platform"

},

{

"id": "1f97b0cb-1f7a-4b4f-bd1d-023234d8b2e1",

"name": "School Admin",

"description": "Manages individual school operations"

},

{

"id": "b31a0d9b-40ef-4f53-9ff5-d84aa9a72672",

"name": "Teacher",

"description": "Teaches assigned subjects and manages student results"

},

{

"id": "ce11db8b-96b3-49d5-9853-87a3bfa84926",

"name": "Student",

"description": "Accesses learning resources and results"

}

]

**Error Responses:**

* **401 Unauthorized – Missing or invalid token**

{

"detail": "Authentication credentials were not provided."

}

This endpoint is powered by RoleListView and uses RoleSerializer.

Would you like to proceed to POST /superadmins/create/?

Great — let’s continue with the next endpoint.

**✅ Endpoint: Create SuperAdmin**

**URL:**

POST /api/superadmins/create/

**Description:**

Registers a new SuperAdmin user by creating a linked User, UserRole, and SuperAdmin profile. Only existing SuperAdmins can perform this operation.

**Authentication Required:**

✅ Yes (JWT access token required)

**Permissions:**

🔒 IsSuperAdmin – only users with the “Super Admin” role

**Request Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**Request Body Example:**

{

"user": {

"username": "newadmin",

"email": "newadmin@example.com",

"password": "StrongPass123"

},

"user\_role": "08de96d7-2b13-4c17-a5d9-f477ad0a67f2", // UUID of Super Admin Role

"surname": "Doe",

"first\_name": "Jane",

"middle\_name": "A.",

"phone\_number": "08012345678",

"address": "12 Admin Lane, Abuja"

}

**Response Example:**

{

"id": "f2a537b1-6e8a-4f44-bf77-c2a5b897b3e9",

"user": {

"id": 23,

"username": "newadmin",

"email": "newadmin@example.com"

},

"role": {

"id": "08de96d7-2b13-4c17-a5d9-f477ad0a67f2",

"name": "Super Admin",

"description": "Has global access and can manage the platform"

},

"surname": "Doe",

"first\_name": "Jane",

"middle\_name": "A.",

"phone\_number": "08012345678",

"address": "12 Admin Lane, Abuja",

"created\_at": "2025-05-08T11:24:17.318Z",

"updated\_at": "2025-05-08T11:24:17.318Z"

}

**Validation Rules:**

* The user\_role must reference a valid Super Admin role ID.
* The username must be unique.
* All required fields must be present.

**Error Responses:**

* **400 Bad Request – Missing or invalid fields**

{

"user\_role": "Invalid role UUID provided."

}

* **403 Forbidden – User is not a SuperAdmin**

{

"detail": "You do not have permission to perform this action."

}

This endpoint uses the SuperAdminCreateSerializer and SuperAdminRegistrationAPIView.

Shall we move to the next one: GET /superadmins/?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List All SuperAdmins**

**URL:**

GET /api/superadmins/

**Description:**

Retrieves a paginated list of all SuperAdmins in the system. Supports filtering by user, surname, and first\_name.

**Authentication Required:**

✅ Yes (JWT access token required)

**Permissions:**

🔒 IsSuperAdmin

**Request Headers:**

Authorization: Bearer <your-access-token>

**Query Parameters (Optional):**

* surname – filter by surname
* first\_name – filter by first name
* user – filter by user ID
* page – pagination page number

**Response Example:**

{

"count": 2,

"next": null,

"previous": null,

"results": [

{

"id": "a1b2c3d4-5678-90ab-cdef-1234567890ab",

"user": 7,

"surname": "Doe",

"first\_name": "Jane",

"middle\_name": "A.",

"phone\_number": "08012345678",

"address": "12 Admin Lane, Abuja",

"created\_at": "2025-05-01T09:00:00Z",

"updated\_at": "2025-05-01T09:00:00Z"

},

...

]

}

**Error Responses:**

* **403 Forbidden – Insufficient permissions**

{

"detail": "You do not have permission to perform this action."

}

This endpoint uses:

* View: SuperAdminListAPIView
* Serializer: SuperAdminSerializer
* Pagination: StandardResultsSetPagination
* Filtering: DjangoFilterBackend

Would you like to continue with GET /superadmins/<uuid:id>/ next?

Here’s the next endpoint documentation.

**✅ Endpoint: SuperAdmin Detail, Update, or Delete**

**URL:**

GET /api/superadmins/<uuid:id>/  
PUT /api/superadmins/<uuid:id>/  
PATCH /api/superadmins/<uuid:id>/  
DELETE /api/superadmins/<uuid:id>/

**Description:**

Retrieve, update, or delete a specific SuperAdmin profile using their UUID.

**Authentication Required:**

✅ Yes (JWT access token required)

**Permissions:**

🔒 IsSuperAdmin

**Request Headers:**

Authorization: Bearer <your-access-token>

**🔹 GET – Retrieve SuperAdmin Profile**

**✅ Response Example:**

{

"id": "f2a537b1-6e8a-4f44-bf77-c2a5b897b3e9",

"user": 23,

"surname": "Doe",

"first\_name": "Jane",

"middle\_name": "A.",

"phone\_number": "08012345678",

"address": "12 Admin Lane, Abuja",

"created\_at": "2025-05-08T11:24:17.318Z",

"updated\_at": "2025-05-08T11:24:17.318Z"

}

**🔹 PUT/PATCH – Update SuperAdmin Profile**

**📤 Request Body (partial or full):**

{

"surname": "Updated",

"first\_name": "Jane",

"phone\_number": "09012345678"

}

**✅ Response Example:**

{

"id": "f2a537b1-6e8a-4f44-bf77-c2a5b897b3e9",

"user": 23,

"surname": "Updated",

"first\_name": "Jane",

"middle\_name": "A.",

"phone\_number": "09012345678",

"address": "12 Admin Lane, Abuja",

"created\_at": "2025-05-08T11:24:17.318Z",

"updated\_at": "2025-05-09T09:15:12.001Z"

}

**❗ Validation Rule:**

* You **cannot** update the id field.

{ "error": "Cannot update the SuperAdmin ID." }

**🔹 DELETE – Remove SuperAdmin Profile**

**✅ Success Response:**

{

"message": "SuperAdmin 'Jane Doe' has been deleted."

}

**❗ Restricted Deletion:**

You **cannot delete your own** SuperAdmin account.

{

"error": "You cannot delete your own profile."

}

**Internals:**

* View: SuperAdminDetailAPIView
* Serializer: SuperAdminSerializer
* Lookup field: id (UUID)

Ready to continue with the next one: POST /schools/create/?

**SCHOOL**

**✅ Endpoint: Create School**

**URL:**

POST /api/schools/create/

**Description:**

Creates a new school and automatically links it to the registered\_by SuperAdmin account that initiated the request.

**Authentication Required:**

✅ Yes (JWT access token required)

**Permissions:**

🔒 IsSuperAdmin

**Request Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**Request Body Example:**

{

"school\_name": "Greenlight International School",

"school\_address": "24 Knowledge Way, Ikeja",

"city": "Ikeja",

"state": "Lagos",

"region": "South-West",

"country": "Nigeria",

"email": "admin@greenlight.edu.ng",

"phone\_number": "08056784321",

"short\_name": "GLIS",

"logo": null,

"school\_type": "Private",

"education\_level": "Primary and Secondary"

}

**Response Example:**

{

"id": "fa3d9f92-6b7e-42c1-aed3-14cb84a3b2d5",

"school\_name": "Greenlight International School",

"school\_address": "24 Knowledge Way, Ikeja",

"city": "Ikeja",

"state": "Lagos",

"region": "South-West",

"country": "Nigeria",

"email": "admin@greenlight.edu.ng",

"phone\_number": "08056784321",

"short\_name": "GLIS",

"logo": null,

"school\_type": "Private",

"education\_level": "Primary and Secondary"

}

**❗ Validation Rules:**

* The combination of school\_name and school\_address must be unique.
* The email must also be unique (if provided).

**🔁 Internal Behavior:**

* registered\_by is auto-assigned from the authenticated user (request.user).
* A Subscription is automatically created via signals.py upon school creation.

**View:** SchoolCreateAPIView  
**Serializer:** SchoolCreateSerializer

Would you like to proceed with GET /schools/ next?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List All Schools**

**URL:**

GET /api/schools/

**Description:**

Retrieves a paginated list of all registered schools. Supports filtering and search.

**Authentication Required:**

✅ Yes (JWT access token required)

**Permissions:**

🔒 IsSuperAdmin

**Request Headers:**

Authorization: Bearer <your-access-token>

**Query Parameters (Optional):**

* city – filter by city
* state – filter by state
* school\_type – filter by school type
* education\_level – filter by education level
* search – search by school\_name or short\_name
* page – page number for pagination

**Response Example:**

{

"count": 2,

"next": null,

"previous": null,

"results": [

{

"id": "fa3d9f92-6b7e-42c1-aed3-14cb84a3b2d5",

"school\_name": "Greenlight International School",

"school\_address": "24 Knowledge Way, Ikeja",

"city": "Ikeja",

"state": "Lagos",

"region": "South-West",

"country": "Nigeria",

"email": "admin@greenlight.edu.ng",

"phone\_number": "08056784321",

"short\_name": "GLIS",

"logo": null,

"school\_type": "Private",

"education\_level": "Primary and Secondary",

"registered\_by": {

"id": "09b9eea0-28e7-4f12-a9ef-254a6ef6e0f4",

"surname": "Adebayo",

"first\_name": "Samuel"

},

"created\_at": "2025-05-08T13:09:00.123Z",

"status": true

}

]

}

**Internals:**

* View: SchoolListAPIView
* Serializer: SchoolListSerializer
* Pagination: StandardResultsSetPagination
* Filters:
  + DjangoFilterBackend
  + SearchFilter (on school\_name, short\_name)

Ready to continue with GET, PUT, DELETE /schools/<uuid:id>/?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Retrieve, Update, or Delete a School**

**URL:**

GET /api/schools/<uuid:id>/  
PUT /api/schools/<uuid:id>/  
PATCH /api/schools/<uuid:id>/  
DELETE /api/schools/<uuid:id>/

**Description:**

Retrieve detailed information about a specific school, update school details, or delete the school record.

**Authentication Required:**

✅ Yes

**Permissions:**

🔒 IsSuperAdmin

**Headers:**

Authorization: Bearer <your-access-token>

**🔹 GET – Retrieve a School**

**✅ Response:**

{

"id": "fa3d9f92-6b7e-42c1-aed3-14cb84a3b2d5",

"school\_name": "Greenlight International School",

"school\_address": "24 Knowledge Way, Ikeja",

"city": "Ikeja",

"state": "Lagos",

"region": "South-West",

"country": "Nigeria",

"email": "admin@greenlight.edu.ng",

"phone\_number": "08056784321",

"short\_name": "GLIS",

"logo": null,

"school\_type": "Private",

"education\_level": "Primary and Secondary"

}

**🔹 PUT/PATCH – Update a School**

**📤 Request Body:**

{

"school\_name": "Updated School Name",

"city": "Updated City",

"school\_type": "Public"

}

**❗ Cannot Update:**

* id – attempting to update this will return an error.

{ "error": "Cannot update the school ID." }

**🔹 DELETE – Delete a School**

**✅ Response:**

{

"message": "School 'Greenlight International School' and all associated records have been deleted."

}

**❗ Restricted To:**

Only authenticated users with a SuperAdmin role.

**Internals:**

* View: SchoolDetailAPIView
* Serializer: SchoolCreateSerializer
* Lookup Field: id

Ready to continue with the next endpoint: GET /schools/subscriptions/?

**SUBSCRIPTIONS**

Here’s the documentation for the next endpoint.

**✅ Endpoint: List All School Subscriptions**

**URL:**

GET /api/schools/subscriptions/

**Description:**

Retrieves a list of all school subscriptions, including live student count, expected fees, and active status.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin

**Headers:**

Authorization: Bearer <your-access-token>

**✅ Response Example:**

[

{

"subscription\_id": "f93dc642-06d5-44db-b589-e4e65d7fcd23",

"school": "a99dcac9-e3b3-4b8e-b4e2-d74979ce25f2",

"school\_name": "Greenlight International School",

"amount\_per\_student": 1000.0,

"amount\_paid": 95000.0,

"expired\_date": "2026-05-07",

"active\_date": "2025-05-07",

"live\_number\_students": 100,

"live\_expected\_fee": 100000.0,

"live\_is\_active": false

}

]

**🧠 Internals:**

* **Live values** like live\_number\_students, live\_expected\_fee, and live\_is\_active are computed dynamically from the database.
* View: SubscriptionListView
* Serializer: SubscriptionSerializer

Would you like to proceed with the next endpoint:  
GET/PATCH /schools/subscriptions/<uuid:subscription\_id>/?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Retrieve or Update a Specific School Subscription**

**URL:**

GET /api/schools/subscriptions/<uuid:subscription\_id>/  
PATCH /api/schools/subscriptions/<uuid:subscription\_id>/

**Description:**

Retrieves or updates details of a school’s subscription. Only amount\_per\_student and amount\_paid can be updated.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 GET – Retrieve Subscription**

**✅ Response:**

{

"subscription\_id": "b9aa8383-cc73-4695-b9c5-9db9295ddadc",

"school": "2a1f8c70-7ad7-4bd7-87f4-91737d59bacc",

"school\_name": "Greenlight International School",

"amount\_per\_student": 1500.0,

"amount\_paid": 120000.0,

"expired\_date": "2026-06-01",

"active\_date": "2025-06-01",

"live\_number\_students": 80,

"live\_expected\_fee": 120000.0,

"live\_is\_active": true

}

**🔹 PATCH – Update Subscription**

**📤 Request Body:**

{

"amount\_per\_student": 1600.0,

"amount\_paid": 128000.0

}

**✅ Response:**

{

"subscription\_id": "b9aa8383-cc73-4695-b9c5-9db9295ddadc",

"amount\_per\_student": 1600.0,

"amount\_paid": 128000.0,

"live\_expected\_fee": 128000.0,

"live\_is\_active": true,

...

}

**🧠 Internals:**

* View: SubscriptionDetailUpdateView
* Serializer: SubscriptionSerializer
* PATCH supports partial updates only.
* live\_is\_active returns true if:  
  amount\_paid >= live\_expected\_fee AND today <= expired\_date

Ready to proceed with POST /schools/compliance-verification/create/?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Create Compliance Verification for a School**

**URL:**

POST /api/schools/compliance-verification/create/

**Description:**

Uploads and submits a school's compliance documentation for verification.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin or SchoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: multipart/form-data

**🔹 Request Body (Multipart/Form-Data):**

| **Field** | **Type** | **Required** | **Description** |
| --- | --- | --- | --- |
| school | UUID | ✅ | ID of the school |
| accreditation\_certificates | file | ✅ | Upload of accreditation document (image/pdf) |
| proof\_of\_registration | file | ✅ | Upload of proof of registration |
| tax\_identification\_number | string | ✅ | School's TIN (Tax ID Number) |
| compliance | boolean | ❌ | Defaults to true |
| approved | boolean | ❌ | Defaults to false |

**✅ Response Example:**

{

"school": "adf91d29-801a-4ec5-b7f2-9476f5e49125",

"school\_name": "Greenlight International School",

"accreditation\_certificates": "/media/compliance/accreditation/accreditation\_cert.pdf",

"proof\_of\_registration": "/media/compliance/registration/registration\_cert.pdf",

"tax\_identification\_number": "TIN-9328483829",

"compliance": true,

"approved": false

}

**🧠 Internals:**

* View: ComplianceVerificationCreateView
* Serializer: ComplianceVerificationSerializer
* Files are saved using the specified upload paths in the model.

Ready to continue with the next endpoint:  
GET /schools/compliance-verification/?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List All Compliance Verification Records**

**URL:**

GET /api/schools/compliance-verification/

**Description:**

Fetches all compliance verification records. Supports filtering by school name, compliance status, and approval status.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin or SchoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

**🔹 Optional Query Parameters:**

| **Parameter** | **Type** | **Description** |
| --- | --- | --- |
| school\_name | string | Filters by partial/full school name match |
| compliance | bool | Filters by compliance status (true or false) |
| approved | bool | Filters by approval status (true or false) |

**✅ Response Example:**

[

{

"school": "adf91d29-801a-4ec5-b7f2-9476f5e49125",

"school\_name": "Greenlight International School",

"accreditation\_certificates": "/media/compliance/accreditation/accreditation\_cert.pdf",

"proof\_of\_registration": "/media/compliance/registration/registration\_cert.pdf",

"tax\_identification\_number": "TIN-9328483829",

"compliance": true,

"approved": false

}

]

**🧠 Internals:**

* View: ComplianceVerificationListView
* Serializer: ComplianceVerificationSerializer
* Filters handled via query parameters in get\_queryset()

Ready for the next endpoint?  
GET /schools/compliance-verification/<uuid:school\_id>/ (retrieve or update a specific compliance record)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Retrieve or Update a School's Compliance Verification**

**URL:**

GET /api/schools/compliance-verification/<uuid:school\_id>/  
PATCH /api/schools/compliance-verification/<uuid:school\_id>/

**Description:**

Fetches or updates the compliance verification record for a specific school.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin or SchoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 GET – Retrieve Compliance Record**

**✅ Response:**

{

"school": "adf91d29-801a-4ec5-b7f2-9476f5e49125",

"school\_name": "Greenlight International School",

"accreditation\_certificates": "/media/compliance/accreditation/accreditation\_cert.pdf",

"proof\_of\_registration": "/media/compliance/registration/registration\_cert.pdf",

"tax\_identification\_number": "TIN-9328483829",

"compliance": true,

"approved": false

}

**🔹 PATCH – Update Compliance Record**

**📤 Request Body Example:**

{

"compliance": true,

"approved": true

}

**✅ Response:**

{

"school": "adf91d29-801a-4ec5-b7f2-9476f5e49125",

"school\_name": "Greenlight International School",

"accreditation\_certificates": "/media/compliance/accreditation/accreditation\_cert.pdf",

"proof\_of\_registration": "/media/compliance/registration/registration\_cert.pdf",

"tax\_identification\_number": "TIN-9328483829",

"compliance": true,

"approved": true

}

**🧠 Internals:**

* View: ComplianceVerificationDetailView
* Serializer: ComplianceVerificationSerializer
* Lookup Field: school\_id

Ready for the next endpoint:  
POST /messages/send/ (send message to users)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Send Message to Users**

**URL:**

POST /api/messages/send/

**Description:**

Allows sending messages to individual users, all teachers, or students in a specific class arm. Only users in the same school or SuperAdmins can send messages.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsAuthenticated (with role-based restrictions)

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Options:**

**✅ Option 1: Send to Specific Users**

{

"recipients": ["<user\_id\_1>", "<user\_id\_2>"],

"subject": "Reminder",

"content": "All students must submit assignments by Friday."

}

**✅ Option 2: Send to All Teachers in Sender’s School**

{

"all\_teachers": true,

"subject": "Meeting Notice",

"content": "There will be a staff meeting on Thursday at 10 AM."

}

**✅ Option 3: Send to Students in a Specific Class**

{

"class\_level": "<class\_id>",

"arm\_name": "A",

"subject": "Notice",

"content": "Please bring your textbooks tomorrow."

}

**✅ Response Example:**

{

"message": "Messages sent successfully."

}

**🔒 Access Rules:**

* SuperAdmins → can only message SchoolAdmins.
* SchoolAdmins/Teachers → can only message users in their school.
* Messages are saved to the Message model.

**🧠 Internals:**

* View: MessageCreateView
* Serializer: MessageSerializer
* Logic supports atomic creation and validation of message delivery based on sender role.

Ready to continue with the next endpoint:  
GET /messages/inbox/ (list received messages)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List Messages for the Current User**

**URL:**

GET /api/messages/inbox/

**Description:**

Fetches all messages received by the authenticated user.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsAuthenticated

**Headers:**

Authorization: Bearer <your-access-token>

**✅ Response Example:**

[

{

"message\_id": "e7ad3c94-b172-45f9-86e8-5b670fca5bb8",

"sender": 4,

"recipient": 11,

"content": "Please ensure to submit your report by Friday.",

"sent\_at": "2025-05-08T10:23:45.178Z",

"is\_read": false

},

{

"message\_id": "8e47fa15-3bc6-4c1a-83a9-3ccf776fa08d",

"sender": 7,

"recipient": 11,

"content": "Your class has been rescheduled to 1 PM.",

"sent\_at": "2025-05-06T13:02:10.349Z",

"is\_read": true

}

]

**🧠 Internals:**

* View: MessageListView
* Serializer: MessageSerializer
* Query filters messages by: recipient=request.user

Ready for the next endpoint:  
POST /school\_admin/create/ (create new SchoolAdmin)?

**SCHOOL ADMIN**

Here’s the documentation for the next endpoint.

**✅ Endpoint: Create a School Admin**

**URL:**

POST /api/school\_admin/create/

**Description:**

Registers a new school admin with an associated user account and role.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body:**

{

"user": {

"username": "admin\_greenlight",

"email": "admin@greenlight.edu.ng",

"password": "securePassword123"

},

"user\_role": "d0bfcad2-bbf1-4a4b-9ed3-b6c6e5d62100", // UUID of "School Admin" role

"school": "eabf791c-23fc-43d9-b2fa-1c8a17e441c3",

"surname": "Adebayo",

"first\_name": "Grace",

"middle\_name": "O.",

"email": "grace.adebayo@greenlight.edu.ng",

"phone\_number": "08012345678",

"address": "12 Adewale Crescent",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"designation": "Principal"

}

**✅ Response Example:**

{

"schooladmin\_id": "d4a88dc1-3c88-4a93-91b9-89e85770b311",

"user": {

"id": 16,

"username": "admin\_greenlight",

"email": "admin@greenlight.edu.ng"

},

"role": "School Admin",

"school": "eabf791c-23fc-43d9-b2fa-1c8a17e441c3",

"surname": "Adebayo",

"first\_name": "Grace",

"middle\_name": "O.",

"email": "grace.adebayo@greenlight.edu.ng",

"phone\_number": "08012345678",

"address": "12 Adewale Crescent",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"designation": "Principal",

"created\_at": "2025-05-08T10:55:03.382Z",

"updated\_at": "2025-05-08T10:55:03.382Z"

}

**🧠 Internals:**

* View: SchoolAdminCreateView
* Serializer: SchoolAdminCreateSerializer
* User, UserRole, and SchoolAdmin models are all created within a single atomic transaction.

Ready for the next endpoint:  
GET /school\_admin/ (list all SchoolAdmins)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List All School Admins**

**URL:**

GET /api/school\_admin/

**Description:**

Retrieves a paginated list of all registered School Admins across schools.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin

**Headers:**

Authorization: Bearer <your-access-token>

**✅ Response Example:**

[

{

"schooladmin\_id": "ad221183-2b1f-4642-8b64-1e2c17940c3f",

"first\_name": "Grace",

"surname": "Adebayo",

"email": "grace.adebayo@greenlight.edu.ng",

"phone\_number": "08012345678",

"designation": "Principal",

"school\_name": "Greenlight International School"

},

{

"schooladmin\_id": "b18945f3-22f2-4e19-b2e9-bb2d066f1af4",

"first\_name": "John",

"surname": "Obinna",

"email": "john.obinna@sunsethigh.edu.ng",

"phone\_number": "08123456789",

"designation": "Vice Principal",

"school\_name": "Sunset High School"

}

]

**🔍 Optional Query Filters:**

* Supports filtering using query parameters like first\_name, surname, email, etc. if extended with filters.

**🧠 Internals:**

* View: SchoolAdminListView
* Serializer: SchoolAdminListSerializer
* Returns concise details per admin with associated school name.

Ready for the next endpoint:  
GET /school\_admin/<uuid:schooladmin\_id>/ (retrieve specific School Admin detail)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Retrieve a Specific School Admin**

**URL:**

GET /api/school\_admin/<uuid:schooladmin\_id>/

**Description:**

Fetches detailed information about a specific School Admin using their UUID.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin or the School Admin themselves

**Headers:**

Authorization: Bearer <your-access-token>

**✅ Response Example:**

{

"schooladmin\_id": "ad221183-2b1f-4642-8b64-1e2c17940c3f",

"user": 21,

"school": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"school\_name": "Greenlight International School",

"surname": "Adebayo",

"first\_name": "Grace",

"email": "grace.adebayo@greenlight.edu.ng",

"phone\_number": "08012345678",

"address": "12 Adewale Crescent",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"designation": "Principal",

"created\_at": "2025-05-08T10:55:03.382Z",

"updated\_at": "2025-05-08T10:55:03.382Z"

}

**🧠 Internals:**

* View: SchoolAdminDetailView
* Serializer: SchoolAdminDetailSerializer
* Lookup by: schooladmin\_id

Would you like to continue with the next endpoint:  
PUT/DELETE /school\_admin/<uuid:schooladmin\_id>/update/ (update or delete School Admin)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Update or Delete a School Admin**

**URL:**

PUT /api/school\_admin/<uuid:schooladmin\_id>/update/  
DELETE /api/school\_admin/<uuid:schooladmin\_id>/update/

**Description:**

Allows updating or deleting a School Admin’s profile.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdminOrSchoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 PUT Request Body Example:**

{

"user": {

"username": "admin\_greenlight",

"email": "admin\_new@greenlight.edu.ng",

"password": "newSecurePassword"

},

"user\_role": "d0bfcad2-bbf1-4a4b-9ed3-b6c6e5d62100",

"school": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"surname": "Adebayo",

"first\_name": "Grace",

"email": "grace.adebayo@greenlight.edu.ng",

"phone\_number": "08098765432",

"address": "14 New Crescent",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"designation": "Vice Principal"

}

**✅ PUT Response Example:**

{

"schooladmin\_id": "ad221183-2b1f-4642-8b64-1e2c17940c3f",

"user": {

"id": 21,

"username": "admin\_greenlight",

"email": "admin\_new@greenlight.edu.ng"

},

"role": "School Admin",

"school": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"surname": "Adebayo",

"first\_name": "Grace",

"email": "grace.adebayo@greenlight.edu.ng",

"phone\_number": "08098765432",

"address": "14 New Crescent",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"designation": "Vice Principal",

"created\_at": "2025-05-08T10:55:03.382Z",

"updated\_at": "2025-05-08T12:42:19.814Z"

}

**✅ DELETE Response Example:**

{

"message": "SchoolAdmin 'Grace Adebayo' has been deleted."

}

**🧠 Internals:**

* View: SchoolAdminUpdateDeleteView
* Serializer: SchoolAdminUpdateSerializer
* Only SuperAdmins can delete; School Admins can update their own profile.

Ready for the next endpoint:  
POST /registration-pins/generate/ (generate OTP pins for student registration)?

**Self-Registration**

**✅ Endpoint: Generate Registration PINs for Student Self-Registration**

**URL:**

POST /api/registration-pins/generate/

**Description:**

Generates one-time registration PINs (OTPs) that students can use to self-register into a specific school.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body:**

{

"num\_pins": 5

}

**✅ Response Example:**

{

"message": "Pins generated successfully.",

"pins": [

{

"pin\_id": "d7e388c8-519a-4053-9dc3-743e44d29dd5",

"school\_id": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"otp": "371829"

},

{

"pin\_id": "ccf122d0-292c-4568-950e-d6e30f3ec69d",

"school\_id": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"otp": "419028"

}

// ...

]

}

**🧠 Internals:**

* View: GenerateRegistrationPinsView
* Model: StudentRegistrationPin
* Default OTP length is 6 digits.
* Each PIN is marked as is\_used=False on creation and linked to the admin’s school.

Ready for the next endpoint:  
POST /registration/verify/ (verify a PIN before self-registration)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Verify Student Registration PIN**

**URL:**

POST /api/registration/verify/

**Description:**

Verifies a student's one-time PIN (otp) and school\_id before allowing self-registration. Returns a temporary token and class options if successful.

**Authentication Required:**

❌ No

**Request Body:**

{

"otp": "371829",

"school\_id": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0"

}

**✅ Successful Response:**

{

"message": "Verification successful.",

"temp\_token": "d7e388c8-519a-4053-9dc3-743e44d29dd5:QWNiRXJ...hX0E", // Used for registration

"class\_years": [

{

"id": "e4b08394-71d2-4d17-9f49-6db1ac7ccaff",

"name": "SS 1",

"class\_arms": [

{

"class\_id": "12137f85-4f3b-49c5-964c-f70992a3d6d0",

"arm\_name": "A"

},

{

"class\_id": "8cdc8b1f-b4e0-48f0-8513-9170ba5ff513",

"arm\_name": "B"

}

]

},

{

"id": "c8f9fa3a-0d9a-4206-98de-126d39cf1c91",

"name": "SS 2"

"class\_arms": [

{

"class\_id": "12137f85-4f3b-49c5-964c-f70992a3d6d0",

"arm\_name": "A"

},

{

"class\_id": "8cdc8b1f-b4e0-48f0-8513-9170ba5ff513",

"arm\_name": "B"

}

]

},

}

],

}

**❌ Error Response:**

{

"error": "Invalid or used OTP."

}

**🧠 Internals:**

* View: VerifyRegistrationPinView
* Utilizes: validate\_temp\_token() from utils.py
* On success: returns class/year options and a temporary token for StudentSelfRegistrationView.

Next up:  
GET /registration-pins/ (list all generated registration pins for the school admin)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List All Registration PINs for a School**

**URL:**

GET /api/registration-pins/

**Description:**

Returns a list of all one-time student registration PINs generated by the logged-in School Admin.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

**✅ Response Example:**

{

"message": "List of registration pins.",

"pins": [

{

"pin\_id": "e3f8ec57-3c3c-4456-8998-683b2fa1f8c1",

"school": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"school\_name": "Bright Future Academy",

"otp": "123456",

"is\_used": false,

"created\_at": "2025-05-08T09:41:12.345Z"

},

{

"pin\_id": "cd321c6f-8b0b-487a-8e60-b74f4f6d8944",

"school": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"school\_name": "Bright Future Academy",

"otp": "654321",

"is\_used": true,

"created\_at": "2025-05-07T13:15:45.789Z"

}

]

}

**🧠 Internals:**

* View: ListRegistrationPinsView
* Serializer: RegistrationPinSerializer
* Filters pins by the current School Admin’s school.

Next endpoint:  
POST /students/create/ (create a single student)?

**STUDENT**

Here’s the documentation for the next endpoint.

**✅ Endpoint: Create a Single Student**

**URL:**

POST /api/students/create/

**Description:**

Creates a new student profile, including their User, assigns the Student role, links them to the School Admin's school, and assigns them to a class.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Example:**

{

"user": {

"username": "studentjoy",

"email": "joy@example.com",

"password": "joypassword"

},

"admission\_number": 2025006,

"first\_name": "Joy",

"middle\_name": "T.",

"last\_name": "Okafor",

"date\_of\_birth": "2010-09-23",

"gender": "Female",

"address": "12 Peace Avenue",

"city": "Aba",

"state": "Abia",

"region": "South East",

"country": "Nigeria",

"admission\_date": "2025-09-01",

"status": "Active",

"parent\_first\_name": "Emeka",

"parent\_last\_name": "Okafor",

"parent\_occupation": "Trader",

"parent\_contact\_info": "08011223344",

"parent\_emergency\_contact": "08033445566",

"parent\_relationship": "Father",

"class\_year": "4e08c552-e9dc-4c9d-81fc-ef4716b8e13b",

"class\_arm": "f612e195-d970-47f2-99c8-7a89b59d6c1b"

}

**✅ Response Example:**

{

"student\_id": "ae984e90-6d67-4984-91ae-b0b6c15c8e1f",

"user": {

"id": 17,

"username": "studentjoy",

"email": "joy@example.com"

},

"role": "Student",

"school": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

"admission\_number": 2025006,

"first\_name": "Joy",

"middle\_name": "T.",

"last\_name": "Okafor",

"date\_of\_birth": "2010-09-23",

"gender": "Female",

"address": "12 Peace Avenue",

"city": "Aba",

"state": "Abia",

"region": "South East",

"country": "Nigeria",

"admission\_date": "2025-09-01",

"status": "Active",

"profile\_picture\_path": null,

"parent\_first\_name": "Emeka",

"parent\_middle\_name": null,

"parent\_last\_name": "Okafor",

"parent\_occupation": "Trader",

"parent\_contact\_info": "08011223344",

"parent\_emergency\_contact": "08033445566",

"parent\_relationship": "Father"

}

**🧠 Internals:**

* View: StudentCreateView
* Serializer: StudentCreateSerializer
* Automatically assigns school from the School Admin’s user session.
* Class assignment is handled via StudentClass creation.

Next endpoint:  
POST /students/bulk\_create/ (bulk student registration from CSV)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Bulk Create Students via CSV Upload**

**URL:**

POST /api/students/bulk\_create/

**Description:**

Allows a School Admin to bulk register multiple students by uploading a CSV file. Each row in the CSV corresponds to a student record and their assigned class.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: multipart/form-data

**🔹 CSV Fields Required:**

The file must include the following headers:

username,password,email,first\_name,last\_name,admission\_number,

date\_of\_birth,gender,address,city,state,region,country,admission\_date,

parent\_first\_name,parent\_last\_name,parent\_occupation,parent\_contact\_info,

parent\_emergency\_contact,parent\_relationship,class\_year\_name,class\_arm\_name

**✅ Successful Response:**

{

"message": "Students registered successfully.",

"errors": null

}

**❌ Example of Error Response:**

{

"errors": [

{

"row": 3,

"error": "Missing required fields."

},

{

"row": 7,

"error": "Invalid class year or arm: JSS 2, Z"

}

]

}

**🧠 Internals:**

* View: StudentBulkCreateView
* Uses pandas to read the file.
* Assigns each student to the school of the current School Admin.
* Also assigns students to class via StudentClass using class\_year\_name and class\_arm\_name.

Next endpoint:  
POST /students/self-register/ (student self-registration using temporary token)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Student Self-Registration**

**URL:**

POST /api/students/self-register/

**Description:**

Allows a student to self-register using a verified temporary token (temp\_token) obtained from the OTP verification. Automatically assigns the student to the correct school and class.

**Authentication Required:**

❌ No

**Headers:**

Content-Type: application/json

**🔹 Request Body Example:**

{

"temp\_token": "e3f8ec57-3c3c-4456-8998-683b2fa1f8c1:9rTPYn8kjISY5F3...",

"user": {

"username": "jstudent",

"email": "js@example.com",

"password": "securePass123"

},

"admission\_number": 2025009,

"first\_name": "John",

"last\_name": "Student",

"gender": "Male",

"date\_of\_birth": "2009-06-15",

"address": "45 Unity Lane",

"city": "Kaduna",

"state": "Kaduna",

"region": "North West",

"country": "Nigeria",

"admission\_date": "2025-09-01",

"parent\_first\_name": "Mark",

"parent\_last\_name": "Student",

"parent\_occupation": "Civil Servant",

"parent\_contact\_info": "07033445566",

"parent\_emergency\_contact": "08099887766",

"parent\_relationship": "Father",

"class\_year\_name": "JSS 1",

"class\_arm\_name": "A"

}

**✅ Success Response:**

{

"student\_id": "123e4567-e89b-12d3-a456-426614174000",

"user": {

"id": 12,

"username": "jstudent",

"email": "js@example.com"

},

"role": "Student",

"school": "e8d84323-32f7-4662-912b-6fc5e9ff3056",

"admission\_number": 2025009,

"first\_name": "John",

"last\_name": "Student",

...

}

**❌ Error Responses:**

{

"error": "Invalid or expired temporary token."

}

{

"error": "Invalid class\_year\_name or class\_arm\_name."

}

**🧠 Internals:**

* View: StudentSelfRegistrationView
* Uses validate\_temp\_token() from utils.py
* Resolves class\_year\_name and class\_arm\_name to actual model instances for assignment.

Next endpoint:  
PUT /students/update/ (update student profile and class assignment)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Update Student Profile**

**URL:**

PUT /api/students/update/

**Description:**

Allows a student to update their own profile, or allows a School Admin to update any student’s profile. Supports re-assigning the student to a different class and arm.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin or ISstudent

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Example:**

{

"user": {

"username": "student\_updated",

"email": "student\_updated@example.com",

"password": "newpass123"

},

"admission\_number": 2025009,

"first\_name": "John",

"last\_name": "Updated",

"gender": "Male",

"date\_of\_birth": "2009-06-15",

"address": "123 Unity Way",

"city": "Lagos",

"state": "Lagos",

"region": "South West",

"country": "Nigeria",

"admission\_date": "2025-09-01",

"status": "Active",

"parent\_first\_name": "Mark",

"parent\_last\_name": "Updated",

"parent\_occupation": "Lecturer",

"parent\_contact\_info": "08034567890",

"parent\_emergency\_contact": "08011122334",

"parent\_relationship": "Father",

"class\_year": "4e08c552-e9dc-4c9d-81fc-ef4716b8e13b",

"class\_arm": "f612e195-d970-47f2-99c8-7a89b59d6c1b"

}

**✅ Success Response:**

{

"student\_id": "ae984e90-6d67-4984-91ae-b0b6c15c8e1f",

"user": {

"id": 17,

"username": "student\_updated",

"email": "student\_updated@example.com"

},

"role": "Student",

"school": "f35f909b-4d4f-49ee-81a1-bf255d8f84f0",

...

}

**🧠 Internals:**

* View: StudentUpdateView
* Serializer: StudentUpdateSerializer
* Replaces the user and student class assignment using the provided UUIDs for class\_year and class\_arm.

Next endpoint:  
GET /students/list/ (list all students in a school)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List All Students**

**URL:**

GET /api/students/list/

**Description:**

Retrieves a paginated list of all students in the school of the currently authenticated School Admin.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

**🔍 Optional Query Parameters for Filtering:**

* first\_name=John
* last\_name=Smith
* middle\_name=David
* gender=Male
* search=Abuja (matches city, region, or country)

**✅ Sample Response:**

{

"count": 3,

"next": null,

"previous": null,

"results": [

{

"student\_id": "8f7765b3-f48d-43a7-b3e7-38fdee062d91",

"admission\_number": 1001,

"first\_name": "John",

"last\_name": "Smith",

"status": "Active",

"parent\_contact\_info": "08012345678"

},

{

"student\_id": "a8739b18-6b1b-4b23-8ee5-68fc2d9dce73",

"admission\_number": 1002,

"first\_name": "Jane",

"last\_name": "Doe",

"status": "Active",

"parent\_contact\_info": "08098765432"

}

]

}

**🧠 Internals:**

* View: StudentListView
* Serializer: StudentListSerializer
* Filters students using DjangoFilterBackend and SearchFilter.
* Pagination: StandardResultsSetPagination

Next endpoint:  
GET /students/detail/<uuid:pk>/ (retrieve single student’s full profile)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Retrieve Student Details**

**URL:**

GET /api/students/detail/<uuid:pk>/

**Description:**

Fetches the full profile of a specific student, including personal information and their current class assignment (year and arm).

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin or ISstudent

**Headers:**

Authorization: Bearer <your-access-token>

**✅ Sample Response:**

{

"student\_id": "1b44d62e-e9e0-4f6d-8a94-09c4709d3b93",

"user": 18,

"school": "2cd79180-bfd2-4e1e-9c36-3b65b99e4be1",

"admission\_number": 3005,

"first\_name": "David",

"middle\_name": "Ayomide",

"last\_name": "Ojo",

"date\_of\_birth": "2008-12-15",

"gender": "Male",

"address": "34 Example Street",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"admission\_date": "2023-09-01",

"status": "Active",

"profile\_picture\_path": null,

"parent\_first\_name": "Samuel",

"parent\_middle\_name": null,

"parent\_last\_name": "Ojo",

"parent\_occupation": "Trader",

"parent\_contact\_info": "08090987654",

"parent\_emergency\_contact": "08123456789",

"parent\_relationship": "Father",

"class\_year": "JSS 2",

"class\_arm": "A"

}

**🧠 Internals:**

* View: StudentDetailView
* Serializer: StudentDetailSerializer
* Uses subject\_class.first() to derive class\_year and class\_arm.

Next endpoint:  
DELETE /students/delete-multiple/ (bulk delete students by IDs)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Delete Multiple Students**

**URL:**

DELETE /api/students/delete-multiple/

**Description:**

Deletes multiple student records at once, using a list of student\_ids. Only accessible by School Admins.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Example:**

{

"student\_ids": [

"bfa3b1de-1f6a-4f32-b503-f0b1b67b1185",

"d48751c5-1d94-4c7f-8517-d640c298bf2a"

]

}

**✅ Success Response:**

{

"message": "2 student(s) deleted successfully."

}

**❌ Error Responses:**

{

"error": "A list of student IDs is required."

}

**🧠 Internals:**

* View: DeleteMultipleStudentsView
* Calls Student.objects.filter(student\_id\_\_in=student\_ids).delete()

Next endpoint:  
POST /teachers/create/ (create a single teacher)?

**TEACHER**

Here’s the documentation for the next endpoint.

**✅ Endpoint: Create Teacher**

**URL:**

POST /api/teachers/create/

**Description:**

Creates a new teacher profile along with a user account. Automatically assigns the Teacher role and associates the teacher with the current School Admin's school.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Example:**

{

"user": {

"username": "mrs\_clara",

"email": "clara@school.org",

"password": "teach12345"

},

"first\_name": "Clara",

"middle\_name": "Yemi",

"last\_name": "Adesina",

"date\_of\_birth": "1985-04-10",

"gender": "Female",

"address": "12 Wisdom Lane",

"city": "Abuja",

"state": "FCT",

"region": "North Central",

"country": "Nigeria",

"date\_hire": "2022-01-10",

"qualification": "M.Ed",

"specialization": "English Language",

"status": "Active"

}

**✅ Success Response:**

{

"teacher\_id": "9d7ef6ca-343a-4e3c-84fa-e9339aee128a",

"user": {

"id": 25,

"username": "mrs\_clara",

"email": "clara@school.org"

},

"role": "Teacher",

"school": "f2f3cd76-56d9-46a7-b9fa-7b7e41f8d7a3",

...

}

**🧠 Internals:**

* View: TeacherCreateView
* Serializer: TeacherCreateSerializer
* Automatically:
  + creates User
  + assigns Teacher role via UserRole
  + links teacher to the requesting admin's school

Next endpoint:  
POST /teachers/bulk\_create/ (bulk upload teachers via CSV)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Bulk Register Teachers**

**URL:**

POST /api/teachers/bulk\_create/

**Description:**

Allows School Admins to upload a CSV file to register multiple teachers at once. Each row in the CSV should represent a teacher's details.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: multipart/form-data

**📄 CSV File Requirements:**

**Required Fields in CSV:**

* username
* password
* email
* first\_name
* last\_name
* date\_of\_birth
* gender
* address
* city
* state
* region
* country
* date\_hire
* qualification
* specialization

**✅ Example CSV Row:**

username,password,email,first\_name,last\_name,date\_of\_birth,gender,address,city,state,region,country,date\_hire,qualification,specialization

adebayo123,securepass,adebayo@example.com,Adebayo,Olu,1980-03-21,Male,45 Example Rd,Lagos,Lagos,South West,Nigeria,2020-09-01,B.Ed,Mathematics

**✅ Success Response:**

{

"message": "Teachers registered successfully."

}

**❌ Error Response:**

{

"errors": [

{

"row": 3,

"error": "Missing required fields."

}

]

}

**🧠 Internals:**

* View: TeacherBulkCreateView
* CSV is parsed using pandas
* Data is validated row-by-row
* User, UserRole, and Teacher records are created using bulk\_create within a transaction

Next endpoint:  
POST /teachers/self-register/ (teacher registration using OTP)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Teacher Self-Registration**

**URL:**

POST /api/teachers/self-register/

**Description:**

Allows a teacher to register themselves after verifying an OTP (registration pin). The system links the teacher to the school that generated the pin and assigns them the Teacher role.

**Authentication Required:**

❌ No — Public access, but requires a valid temp\_token.

**Permissions:**

🔐 Must provide a valid temp\_token generated via OTP verification.

**Headers:**

Content-Type: application/json

**🔹 Request Body Example:**

{

"temp\_token": "8d78dd44-3b2d-47a9-a99f-45acacbaba4d:UzFNaE93V3RkdnpMcXBzREpZb3lmZz09",

"user": {

"username": "teach\_ife",

"email": "ife@school.com",

"password": "password123"

},

"first\_name": "Ife",

"middle\_name": "Grace",

"last\_name": "Ajayi",

"date\_of\_birth": "1990-05-05",

"gender": "Female",

"address": "24 Scholars Street",

"city": "Ilorin",

"state": "Kwara",

"region": "North Central",

"country": "Nigeria",

"date\_hire": "2023-01-05",

"qualification": "M.Ed",

"specialization": "Biology"

}

**✅ Success Response:**

{

"teacher\_id": "abe271f5-25a4-4bbf-8f39-9984f2432d4e",

"user": {

"id": 35,

"username": "teach\_ife",

"email": "ife@school.com"

},

"role": "Teacher",

"school": "f821b038-0a0b-4e28-918e-d4c9fd7bcb63",

...

}

**❌ Error Response (Invalid token):**

{

"error": "Invalid or expired temporary token."

}

**🧠 Internals:**

* View: TeacherSelfRegistrationView
* Validates the temp\_token using HMAC via validate\_temp\_token()
* Assigns school automatically based on the OTP’s school
* Saves Teacher and linked User with the Teacher role

Next endpoint:  
PATCH /teachers/update/ (update teacher's own profile)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Update Teacher Profile**

**URL:**

PATCH /api/teachers/update/

**Description:**

Allows a teacher to update their own profile information. Accessible only by the authenticated teacher.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 ISteacher

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Example:**

{

"user": {

"username": "teach\_john\_updated",

"email": "john\_updated@example.com",

"password": "newsecurepassword123"

},

"first\_name": "John",

"middle\_name": "A.",

"last\_name": "Doe",

"address": "42 Updated Road",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"status": "Active",

"qualification": "Ph.D.",

"specialization": "Physics"

}

**✅ Success Response:**

{

"teacher\_id": "3a9c8a1b-8e5d-4f4f-9cd9-ff207d392789",

"user": {

"id": 52,

"username": "teach\_john\_updated",

"email": "john\_updated@example.com"

},

"role": "Teacher",

"school": "6d10eafc-fbf3-4ab9-9eea-30981e1dcf0f",

...

}

**❌ Error Response:**

{

"error": "Teacher creation failed: ..."

}

**🧠 Internals:**

* View: TeacherUpdateView
* Serializer: TeacherUpdateSerializer
* Retrieves the authenticated teacher’s record via get\_queryset()
* Updates both User and Teacher data in a transaction

Next endpoint:  
GET /teachers/list/ (list all teachers in the admin’s school)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: List Teachers in a School**

**URL:**

GET /api/teachers/list/

**Description:**

Retrieves a list of all teachers registered in the school of the requesting School Admin. Teachers can also use this to view their colleagues.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin | ISteacher

**Headers:**

Authorization: Bearer <your-access-token>

**🔍 Query Parameters (Optional):**

| **Parameter** | **Description** |
| --- | --- |
| first\_name | Filter by first name |
| last\_name | Filter by last name |
| gender | Filter by gender |
| search | Full-text search on city, region, country |

**Example:**

/api/teachers/list/?gender=Female&search=Ogun

**✅ Success Response:**

[

{

"teacher\_id": "d7b2e230-f9ad-41a4-bef2-8df8e9d5a123",

"first\_name": "Jane",

"last\_name": "Smith",

"qualification": "M.Sc.",

"specialization": "Mathematics",

"status": "Active",

"school\_name": "AlgorithmX High School"

},

...

]

**🧠 Internals:**

* View: TeacherListView
* Serializer: TeacherListSerializer
* Uses DjangoFilterBackend and SearchFilter
* Filters based on the school tied to the authenticated user

Next endpoint:  
DELETE /teachers/delete-multiple/ (delete multiple teachers)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Delete Multiple Teachers**

**URL:**

DELETE /api/teachers/delete-multiple/

**Description:**

Allows School Admins to delete multiple teachers at once by submitting a list of their UUIDs.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Example:**

{

"teacher\_ids": [

"3f9b7d9e-7c0e-453b-aac6-77cfb94b2df2",

"d1b4a0e6-0ae7-49b1-8719-8d4e64907d6c"

]

}

**✅ Success Response:**

{

"message": "2 teacher(s) deleted successfully."

}

**❌ Error Response:**

{

"error": "A list of teacher IDs is required."

}

**🧠 Internals:**

* View: DeleteMultipleTeachersView
* Validates that teacher\_ids is a non-empty list
* Deletes all matching teachers via filter(...).delete()

Next endpoint:  
GET /teachers/detail/<uuid:pk>/ (retrieve a teacher’s profile)?

Here’s the documentation for the next endpoint.

**✅ Endpoint: Retrieve Teacher Profile**

**URL:**

GET /api/teachers/detail/<uuid:pk>/

**Description:**

Returns detailed information about a specific teacher, including personal, professional, and school-related fields.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsschoolAdmin | ISteacher

**Headers:**

Authorization: Bearer <your-access-token>

**🔹 Path Parameter:**

| **Parameter** | **Description** |
| --- | --- |
| pk | UUID of the teacher to retrieve |

**Example:**

/api/teachers/detail/3f9b7d9e-7c0e-453b-aac6-77cfb94b2df2/

**✅ Success Response:**

{

"teacher\_id": "3f9b7d9e-7c0e-453b-aac6-77cfb94b2df2",

"user": {

"id": 21,

"username": "teach\_mary",

"email": "mary@school.com"

},

"school": "f389a370-4570-4655-93b1-7cbeb7bbd65c",

"first\_name": "Mary",

"middle\_name": "A.",

"last\_name": "Oladele",

"date\_of\_birth": "1987-03-11",

"gender": "Female",

"address": "2 Teacher Lane",

"city": "Ado Ekiti",

"state": "Ekiti",

"region": "South West",

"country": "Nigeria",

"date\_hire": "2023-01-10",

"status": "Active",

"qualification": "Ph.D.",

"specialization": "Chemistry",

"profile\_picture\_path": null,

"cv": null

}

**🧠 Internals:**

* View: TeacherDetailView
* Serializer: TeacherDetailSerializer
* Retrieves the record by teacher\_id via lookup\_field='pk'

Ready to proceed to the next endpoint:  
POST /school\_admin/create/ (create a new School Admin)?

**school\_admin**

Here’s the documentation for the next endpoint.

**✅ Endpoint: Create School Admin**

**URL:**

POST /api/school\_admin/create/

**Description:**

Creates a new School Admin account, including a linked User and assigned UserRole. Only SuperAdmins can perform this operation.

**Authentication Required:**

✅ Yes (JWT access token)

**Permissions:**

🔒 IsSuperAdmin

**Headers:**

Authorization: Bearer <your-access-token>

Content-Type: application/json

**🔹 Request Body Example:**

{

"user": {

"username": "admin\_alex",

"email": "alex@school.com",

"password": "securepass123"

},

"user\_role": "248e10d5-f926-44bc-b01b-4b6c91f3fa47",

"school": "ab56d939-5407-44c5-8d32-b3f5370e11fc",

"surname": "Alex",

"first\_name": "Johnson",

"middle\_name": "Lee",

"email": "alex@school.com",

"phone\_number": "08012345678",

"address": "42 Main Street",

"city": "Ibadan",

"state": "Oyo",

"region": "South West",

"country": "Nigeria",

"designation": "Principal"

}

**✅ Success Response:**

{

"schooladmin\_id": "41cc2b9c-991d-4c7e-a845-dbe83c50c56c",

"user": {

"id": 12,

"username": "admin\_alex",

"email": "alex@school.com"

},

"role": "School Admin",

"school": "ab56d939-5407-44c5-8d32-b3f5370e11fc",

"surname": "Alex",

"first\_name": "Johnson",

"middle\_name": "Lee",

"phone\_number": "08012345678",

"address": "42 Main Street",

"created\_at": "2025-05-08T13:45:00Z",

"updated\_at": "2025-05-08T13:45:00Z"

}

**❌ Error Response:**

{

"user\_role": ["The provided role must be 'School Admin'."]

}

**🧠 Internals:**

* View: SchoolAdminCreateView
* Serializer: SchoolAdminCreateSerializer
* Creates User, assigns UserRole (must be "School Admin"), and links to a school

Next endpoint:  
GET /school\_admin/ (list all School Admins)?