**Test reCAPTCHA Endpoint**

* **URL:** /api/authuser/test-captcha/
* **Method:** POST
* **Description:** Validates a reCAPTCHA token to test its integration.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + recaptcha\_token (string, required): The reCAPTCHA token obtained from the frontend.
* **Notes:**
  + Useful for verifying that reCAPTCHA is working correctly before testing other endpoints.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/test-captcha/

Content-Type: application/json

{

"recaptcha\_token": "your-recaptcha-token"

}

**Student Endpoints**

**1. Register a New Student**

* **URL:** /api/authuser/student/register/
* **Method:** POST
* **Description:** Registers a new student account.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + mobile (string, required)
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
  + Other optional fields as defined in the StudentSerializer.
* **Notes:**
  + Sends an activation email upon successful registration.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/student/register/

Content-Type: application/json

{

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

"mobile": "1234567890",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**2. Activate Student Account**

* **URL:** /api/authuser/student/activate/<token>/
* **Method:** GET
* **Description:** Activates a student account using the activation token.
* **Headers:** None
* **Parameters:**
  + <token> (string, required): The activation token sent via email.
* **Notes:**
  + No authentication required.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/student/activate/your-activation-token/

**3. Student Login**

* **URL:** /api/authuser/student/login/
* **Method:** POST
* **Description:** Authenticates a student and returns JWT tokens.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + identifier (string, required): Student's email or mobile.
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Returns access and refresh tokens upon successful login.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/student/login/

Content-Type: application/json

{

"identifier": "student@example.com",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**4. Student Logout**

* **URL:** /api/authuser/student/logout/
* **Method:** POST
* **Description:** Logs out the student by blacklisting the JWT token.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/student/logout/

Authorization: Bearer your-access-token

**5. Change Student Password**

* **URL:** /api/authuser/student/change-password/
* **Method:** POST
* **Description:** Allows the student to change their password.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + old\_password (string, required)
  + new\_password (string, required)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/student/change-password/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"old\_password": "OldP@ssw0rd!",

"new\_password": "NewSecureP@ssw0rd!"

}

**6. Forgot Student Password**

* **URL:** /api/authuser/student/forgot-password/
* **Method:** POST
* **Description:** Initiates a password reset by sending a reset link to the student's email.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Sends a password reset email with a token.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/student/forgot-password/

Content-Type: application/json

{

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

"recaptcha\_token": "your-recaptcha-token"

}

**7. Reset Student Password**

* **URL:** /api/authuser/student/reset-password/<token>/
* **Method:** POST
* **Description:** Resets the student's password using the reset token.
* **Headers:**
  + Content-Type: application/json
* **Parameters:**
  + <token> (string, required): The password reset token.
* **Body Parameters:**
  + new\_password (string, required)
* **Notes:**
  + No authentication required.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/student/reset-password/your-reset-token/

Content-Type: application/json

{

"new\_password": "NewSecureP@ssw0rd!"

}

**8. Student Profile**

**a. Get Student Profile**

* **URL:** /api/authuser/student/profile/
* **Method:** GET
* **Description:** Retrieves the authenticated student's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/student/profile/

Authorization: Bearer your-access-token

**b. Update Student Profile**

* **URL:** /api/authuser/student/profile/
* **Method:** POST
* **Description:** Updates the authenticated student's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + Any fields defined in StudentProfileSerializer except read-only fields (e.g., first\_name, last\_name, etc.)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/student/profile/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"first\_name": "John",

"last\_name": "Doe",

"birthdate": "1990-01-01",

"is\_kyc\_approved": true

}

**Employee Endpoints**

**1. Register a New Employee**

* **URL:** /api/authuser/employee/register/
* **Method:** POST
* **Description:** Registers a new employee account.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + mobile (string, required)
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
  + Other optional fields as defined in the EmployeeSerializer.
* **Notes:**
  + Sends an activation email upon successful registration.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/employee/register/

Content-Type: application/json

{

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

"mobile": "0987654321",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**2. Activate Employee Account**

* **URL:** /api/authuser/employee/activate/<token>/
* **Method:** GET
* **Description:** Activates an employee account using the activation token.
* **Headers:** None
* **Parameters:**
  + <token> (string, required): The activation token sent via email.
* **Notes:**
  + No authentication required.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/employee/activate/your-activation-token/

**3. Employee Login**

* **URL:** /api/authuser/employee/login/
* **Method:** POST
* **Description:** Authenticates an employee and returns JWT tokens.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + identifier (string, required): Employee's email or mobile.
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Returns access and refresh tokens upon successful login.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/employee/login/

Content-Type: application/json

{

"identifier": "employee@example.com",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**4. Employee Logout**

* **URL:** /api/authuser/employee/logout/
* **Method:** POST
* **Description:** Logs out the employee by blacklisting the JWT token.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/employee/logout/

Authorization: Bearer your-access-token

**5. Change Employee Password**

* **URL:** /api/authuser/employee/change-password/
* **Method:** POST
* **Description:** Allows the employee to change their password.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + old\_password (string, required)
  + new\_password (string, required)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/employee/change-password/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"old\_password": "OldP@ssw0rd!",

"new\_password": "NewSecureP@ssw0rd!"

}

**6. Forgot Employee Password**

* **URL:** /api/authuser/employee/forgot-password/
* **Method:** POST
* **Description:** Initiates a password reset by sending a reset link to the employee's email.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Sends a password reset email with a token.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/employee/forgot-password/

Content-Type: application/json

{

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

"recaptcha\_token": "your-recaptcha-token"

}

**7. Reset Employee Password**

* **URL:** /api/authuser/employee/reset-password/<token>/
* **Method:** POST
* **Description:** Resets the employee's password using the reset token.
* **Headers:**
  + Content-Type: application/json
* **Parameters:**
  + <token> (string, required): The password reset token.
* **Body Parameters:**
  + new\_password (string, required)
* **Notes:**
  + No authentication required.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/employee/reset-password/your-reset-token/

Content-Type: application/json

{

"new\_password": "NewSecureP@ssw0rd!"

}

**8. Employee Profile**

**a. Get Employee Profile**

* **URL:** /api/authuser/employee/profile/
* **Method:** GET
* **Description:** Retrieves the authenticated employee's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/employee/profile/

Authorization: Bearer your-access-token

**b. Update Employee Profile**

* **URL:** /api/authuser/employee/profile/
* **Method:** POST
* **Description:** Updates the authenticated employee's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + Any fields defined in EmployeeProfileSerializer except read-only fields (e.g., first\_name, last\_name, etc.)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/employee/profile/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"first\_name": "Jane",

"last\_name": "Smith",

"birthdate": "1985-05-15",

"is\_kyc\_approved": true

}

**Instructor Endpoints**

**1. Register a New Instructor**

* **URL:** /api/authuser/instructor/register/
* **Method:** POST
* **Description:** Registers a new instructor account.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + mobile (string, required)
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
  + Other optional fields as defined in the InstructorSerializer.
* **Notes:**
  + Sends an activation email upon successful registration.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/instructor/register/

Content-Type: application/json

{

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

"mobile": "1122334455",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**2. Activate Instructor Account**

* **URL:** /api/authuser/instructor/activate/<token>/
* **Method:** GET
* **Description:** Activates an instructor account using the activation token.
* **Headers:** None
* **Parameters:**
  + <token> (string, required): The activation token sent via email.
* **Notes:**
  + No authentication required.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/instructor/activate/your-activation-token/

**3. Instructor Login**

* **URL:** /api/authuser/instructor/login/
* **Method:** POST
* **Description:** Authenticates an instructor and returns JWT tokens.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + identifier (string, required): Instructor's email or mobile.
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Returns access and refresh tokens upon successful login.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/instructor/login/

Content-Type: application/json

{

"identifier": "instructor@example.com",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**4. Instructor Logout**

* **URL:** /api/authuser/instructor/logout/
* **Method:** POST
* **Description:** Logs out the instructor by blacklisting the JWT token.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/instructor/logout/

Authorization: Bearer your-access-token

**5. Change Instructor Password**

* **URL:** /api/authuser/instructor/change-password/
* **Method:** POST
* **Description:** Allows the instructor to change their password.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + old\_password (string, required)
  + new\_password (string, required)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/instructor/change-password/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"old\_password": "OldP@ssw0rd!",

"new\_password": "NewSecureP@ssw0rd!"

}

**6. Forgot Instructor Password**

* **URL:** /api/authuser/instructor/forgot-password/
* **Method:** POST
* **Description:** Initiates a password reset by sending a reset link to the instructor's email.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Sends a password reset email with a token.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/instructor/forgot-password/

Content-Type: application/json

{

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

"recaptcha\_token": "your-recaptcha-token"

}

**7. Reset Instructor Password**

* **URL:** /api/authuser/instructor/reset-password/<token>/
* **Method:** POST
* **Description:** Resets the instructor's password using the reset token.
* **Headers:**
  + Content-Type: application/json
* **Parameters:**
  + <token> (string, required): The password reset token.
* **Body Parameters:**
  + new\_password (string, required)
* **Notes:**
  + No authentication required.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/instructor/reset-password/your-reset-token/

Content-Type: application/json

{

"new\_password": "NewSecureP@ssw0rd!"

}

**8. Instructor Profile**

**a. Get Instructor Profile**

* **URL:** /api/authuser/instructor/profile/
* **Method:** GET
* **Description:** Retrieves the authenticated instructor's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/instructor/profile/

Authorization: Bearer your-access-token

**b. Update Instructor Profile**

* **URL:** /api/authuser/instructor/profile/
* **Method:** POST
* **Description:** Updates the authenticated instructor's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + Any fields defined in InstructorProfileSerializer except read-only fields (e.g., first\_name, last\_name, etc.)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/instructor/profile/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"first\_name": "Alice",

"last\_name": "Johnson",

"birthdate": "1975-07-20",

"is\_kyc\_approved": true

}

**Institute Endpoints**

**1. Register a New Institute**

* **URL:** /api/authuser/institute/register/
* **Method:** POST
* **Description:** Registers a new institute account.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + mobile (string, required)
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
  + Other optional fields as defined in the InstituteSerializer.
* **Notes:**
  + Sends an activation email upon successful registration.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/institute/register/

Content-Type: application/json

{

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

"mobile": "5566778899",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**2. Activate Institute Account**

* **URL:** /api/authuser/institute/activate/<token>/
* **Method:** GET
* **Description:** Activates an institute account using the activation token.
* **Headers:** None
* **Parameters:**
  + <token> (string, required): The activation token sent via email.
* **Notes:**
  + No authentication required.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/institute/activate/your-activation-token/

**3. Institute Login**

* **URL:** /api/authuser/institute/login/
* **Method:** POST
* **Description:** Authenticates an institute and returns JWT tokens.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + identifier (string, required): Institute's email or mobile.
  + password (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Returns access and refresh tokens upon successful login.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/institute/login/

Content-Type: application/json

{

"identifier": "institute@example.com",

"password": "SecureP@ssw0rd!",

"recaptcha\_token": "your-recaptcha-token"

}

**4. Institute Logout**

* **URL:** /api/authuser/institute/logout/
* **Method:** POST
* **Description:** Logs out the institute by blacklisting the JWT token.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/institute/logout/

Authorization: Bearer your-access-token

**5. Change Institute Password**

* **URL:** /api/authuser/institute/change-password/
* **Method:** POST
* **Description:** Allows the institute to change their password.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + old\_password (string, required)
  + new\_password (string, required)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/institute/change-password/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"old\_password": "OldP@ssw0rd!",

"new\_password": "NewSecureP@ssw0rd!"

}

**6. Forgot Institute Password**

* **URL:** /api/authuser/institute/forgot-password/
* **Method:** POST
* **Description:** Initiates a password reset by sending a reset link to the institute's email.
* **Headers:**
  + Content-Type: application/json
* **Body Parameters:**
  + email (string, required)
  + recaptcha\_token (string, required): reCAPTCHA token for verification.
* **Notes:**
  + Sends a password reset email with a token.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/institute/forgot-password/

Content-Type: application/json

{

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

"recaptcha\_token": "your-recaptcha-token"

}

**7. Reset Institute Password**

* **URL:** /api/authuser/institute/reset-password/<token>/
* **Method:** POST
* **Description:** Resets the institute's password using the reset token.
* **Headers:**
  + Content-Type: application/json
* **Parameters:**
  + <token> (string, required): The password reset token.
* **Body Parameters:**
  + new\_password (string, required)
* **Notes:**
  + No authentication required.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/institute/reset-password/your-reset-token/

Content-Type: application/json

{

"new\_password": "NewSecureP@ssw0rd!"

}

**8. Institute Profile**

**a. Get Institute Profile**

* **URL:** /api/authuser/institute/profile/
* **Method:** GET
* **Description:** Retrieves the authenticated institute's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
* **Body Parameters:** None
* **Notes:**
  + Requires authentication.

**Example Request:**

GET http://127.0.0.1:8000/api/authuser/institute/profile/

Authorization: Bearer your-access-token

**b. Update Institute Profile**

* **URL:** /api/authuser/institute/profile/
* **Method:** POST
* **Description:** Updates the authenticated institute's profile information.
* **Headers:**
  + Authorization: Bearer <access\_token>
  + Content-Type: application/json
* **Body Parameters:**
  + Any fields defined in InstituteProfileSerializer except read-only fields (e.g., name, email, etc.)
* **Notes:**
  + Requires authentication.

**Example Request:**

POST http://127.0.0.1:8000/api/authuser/institute/profile/

Authorization: Bearer your-access-token

Content-Type: application/json

{

"name": "GrowUpMore Institute",

"is\_kyc\_approved": true

}

**Additional Notes for Testing with Postman**

1. **Authentication Tokens:**
   * After successful login (/login/), you'll receive access and refresh tokens.
   * Use the access token in the Authorization header for protected routes:

**Authorization: Bearer <access\_token>**

1. **reCAPTCHA v3 Integration:**
   * Obtain a recaptcha\_token from your frontend integration of Google reCAPTCHA v3.
   * Include the recaptcha\_token in the body of requests that require it (register/, login/, forgot-password/, etc.).
2. **Activation and Reset Links:**
   * Activation and password reset links are sent via email.
   * Use the token from these links to perform activation (/activate/<token>/) and password reset (/reset-password/<token>/).
3. **Handling Responses:**
   * **Success Responses:** Typically include messages indicating successful operations or return the requested data.
   * **Error Responses:** Contain error fields with descriptive messages. Ensure to handle these appropriately in your frontend or testing procedures.
4. **Rate Limiting:**
   * Login endpoints are rate-limited (e.g., 10 requests per hour) to prevent brute-force attacks. Exceeding this limit will result in blocked requests.
5. **Profile Endpoints:**
   * **GET /profile/:** Retrieves user information.
   * **POST /profile/:** Updates user information. Only include fields you intend to update.

**Summary of All Endpoints**

Below is a summarized table of all endpoints categorized by user type for quick reference.

**Test Endpoint**

| **Method** | **URL** | **Description** |
| --- | --- | --- |
| POST | /api/authuser/test-captcha/ | Test Google reCAPTCHA v3 integration |

**Student Endpoints**

| **Method** | **URL** | **Description** |
| --- | --- | --- |
| POST | /api/authuser/student/register/ | Register a new student |
| GET | /api/authuser/student/activate/<token>/ | Activate student account |
| POST | /api/authuser/student/login/ | Student login |
| POST | /api/authuser/student/logout/ | Student logout |
| POST | /api/authuser/student/change-password/ | Change student password |
| POST | /api/authuser/student/forgot-password/ | Initiate student password reset |
| POST | /api/authuser/student/reset-password/<token>/ | Reset student password |
| GET | /api/authuser/student/profile/ | Get student profile |
| POST | /api/authuser/student/profile/ | Update student profile |

**Employee Endpoints**

| **Method** | **URL** | **Description** |
| --- | --- | --- |
| POST | /api/authuser/employee/register/ | Register a new employee |
| GET | /api/authuser/employee/activate/<token>/ | Activate employee account |
| POST | /api/authuser/employee/login/ | Employee login |
| POST | /api/authuser/employee/logout/ | Employee logout |
| POST | /api/authuser/employee/change-password/ | Change employee password |
| POST | /api/authuser/employee/forgot-password/ | Initiate employee password reset |
| POST | /api/authuser/employee/reset-password/<token>/ | Reset employee password |
| GET | /api/authuser/employee/profile/ | Get employee profile |
| POST | /api/authuser/employee/profile/ | Update employee profile |

**Instructor Endpoints**

| **Method** | **URL** | **Description** |
| --- | --- | --- |
| POST | /api/authuser/instructor/register/ | Register a new instructor |
| GET | /api/authuser/instructor/activate/<token>/ | Activate instructor account |
| POST | /api/authuser/instructor/login/ | Instructor login |
| POST | /api/authuser/instructor/logout/ | Instructor logout |
| POST | /api/authuser/instructor/change-password/ | Change instructor password |
| POST | /api/authuser/instructor/forgot-password/ | Initiate instructor password reset |
| POST | /api/authuser/instructor/reset-password/<token>/ | Reset instructor password |
| GET | /api/authuser/instructor/profile/ | Get instructor profile |
| POST | /api/authuser/instructor/profile/ | Update instructor profile |

**Institute Endpoints**

| **Method** | **URL** | **Description** |
| --- | --- | --- |
| POST | /api/authuser/institute/register/ | Register a new institute |
| GET | /api/authuser/institute/activate/<token>/ | Activate institute account |
| POST | /api/authuser/institute/login/ | Institute login |
| POST | /api/authuser/institute/logout/ | Institute logout |
| POST | /api/authuser/institute/change-password/ | Change institute password |
| POST | /api/authuser/institute/forgot-password/ | Initiate institute password reset |
| POST | /api/authuser/institute/reset-password/<token>/ | Reset institute password |
| GET | /api/authuser/institute/profile/ | Get institute profile |
| POST | /api/authuser/institute/profile/ | Update institute profile |

**Tips for Testing with Postman**

1. **Setting Up Authorization:**
   * After logging in, copy the access token from the response.
   * In Postman, go to the **Headers** tab and add:

Key: Authorization

Value: Bearer <access\_token>

1. **Handling Tokens in URLs:**
   * For activation and reset endpoints, replace <token> with the actual token received via email.
   * Example:

http://127.0.0.1:8000/api/authuser/student/activate/eyJ0eXAiOiJKV1QiLCJhbGciOiJI...

1. **Managing reCAPTCHA Tokens:**
   * Ensure you have a frontend setup that integrates Google reCAPTCHA v3 to obtain valid recaptcha\_token values.
   * Alternatively, use the /test-captcha/ endpoint to validate tokens manually.
2. **Environment Variables in Postman:**
   * Use Postman's environment variables to store tokens and reuse them across requests.
   * Example:
     + {{base\_url}} = http://127.0.0.1:8000/api/authuser/
     + {{student\_access\_token}} = <access\_token>
3. **Testing Profile Endpoints:**
   * **GET Profile:** Ensure the Authorization header is set.
   * **POST Profile:** Send only the fields you wish to update. Read-only fields will be ignored or cause errors.
4. **Error Handling:**
   * Review error messages returned by the API to understand what went wrong (e.g., invalid tokens, missing parameters).
   * Common HTTP status codes:
     + 200 OK: Successful operation.
     + 201 Created: Resource created successfully.
     + 400 Bad Request: Missing or invalid parameters.
     + 401 Unauthorized: Invalid credentials.
     + 403 Forbidden: Access denied (e.g., account locked).
     + 404 Not Found: Resource does not exist.
5. **Rate Limiting:**
   * Be cautious with login attempts to avoid hitting rate limits. Implement delays or use unique identifiers during testing.
6. **Logging:**
   * Check your Django application's logs for detailed information on each request, which can help in debugging issues during testing.