Contents

Authentication API Reference

Version: 5.52.29 | **Last Updated**: January 25, 2026 | **Base URL**: `https://api.radiant.ai/v1`

Complete API reference for RADIANT authentication endpoints.

Table of Contents

Overview

Base URLs

Environment	URL
Production	`https://api.radiant.ai/v1`
Staging	`https://api.staging.radiant.ai/v1`

Authentication Methods

Method	Header	Use Case
Bearer Token	`Authorization: Bearer <token>`	User requests
API Key	`X-API-Key: <key>`	Server-to-server

Common Headers

```
Content-Type: application/json
Accept: application/json
Authorization: Bearer eyJhbGciOiJSUzI1NiIs...
X-Request-ID: uuid-for-tracing
```

Rate Limits

Endpoint Category	Limit	Window
Authentication	10 requests	1 minute
Password reset	3 requests	1 hour
MFA verification	5 requests	1 minute
General API	100 requests	1 minute

Authentication

Get Current User

Retrieve the authenticated user's profile.

```
GET /auth/me
Authorization: Bearer <access_token>
```

Response 200:

```json
{
  "id": "usr_abc123def456",
  "email": "user@example.com",
  "name": "Jane Doe",
  "avatar_url": "https://cdn.radiant.ai/avatars/abc123.png",
  "tenant_id": "ten_xyz789",
  "roles": ["member"],
  "mfa_enabled": true,
  "language": "en",
  "timezone": "America/New_York",
  "created_at": "2024-01-15T10:30:00Z",
  "last_sign_in_at": "2026-01-25T08:00:00Z"
}
```

Sign In / Sign Out

Sign In with Email/Password

```
POST /auth/signin
Content-Type: application/json
```

Request Body:

```json
{
  "email": "user@example.com",
  "password": "SecurePassword123!",
  "remember_me": true
}
```

Response 200 (Success):

```json
{
  "access_token": "eyJhbGciOiJSUzI1NiIs...",
  "refresh_token": "dGhpcyBpcyBhIHJlZnJlc2g...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "user": {
    "id": "usr_abc123def456",
    "email": "user@example.com",
    "name": "Jane Doe"
  }
}
```

Response 200 (MFA Required):

```json
{
  "challenge": "MFA_REQUIRED",
  "session": "session_token_for_mfa",
  "mfa_methods": ["totp", "backup_code"]
}
```

Response 401:

```json
{
  "error": "invalid_credentials",
  "message": "Invalid email or password"
}
```

Sign In with SSO

Initiate SSO sign-in flow.

```
POST /auth/sso/initiate
Content-Type: application/json
```

Request Body:

```json
{
  "email": "user@company.com"
}
```

Response 200:

```json
{
  "sso_url": "https://idp.company.com/saml/sso?SAMLRequest=...",
  "provider": "saml",
  "provider_name": "Company SSO"
}
```

Response 200 (No SSO):

```json
{
  "sso_enabled": false,
  "message": "No SSO configured for this domain"
}
```

SSO Callback

Handle SSO provider callback (internal use).

```
POST /auth/sso/callback
Content-Type: application/x-www-form-urlencoded
```

Sign Out

```
POST /auth/signout
Authorization: Bearer <access_token>
```

Request Body (optional):

```
{
  "all_devices": false
}
```

Response 200:

```
{
  "success": true,
  "message": "Signed out successfully"
}
```

Refresh Token

Exchange refresh token for new access token.

```
POST /auth/refresh
Content-Type: application/json
```

Request Body:

```
{
  "refresh_token": "dGhpcyBpcyBhIHJlZnJlc2g..."
}
```

Response 200:

```
{
  "access_token": "eyJhbGciOiJSUzI1NiIs...",
  "refresh_token": "bmV3IHJlZnJlc2ggdG9rZW4...",
  "token_type": "Bearer",
  "expires_in": 3600
}
```

Password Management

Request Password Reset

```
POST /auth/password/forgot
Content-Type: application/json
```

Request Body:

```json
{
  "email": "user@example.com"
}
```

Response 200:

```json
{
  "success": true,
  "message": "If an account exists, a reset link has been sent"
}
```

> **Note**: Always returns success to prevent email enumeration.

Reset Password

```
POST /auth/password/reset
Content-Type: application/json
```

Request Body:

```json
{
  "token": "reset_token_from_email",
  "password": "NewSecurePassword123!",
  "password_confirmation": "NewSecurePassword123!"
}
```

Response 200:

```json
{
  "success": true,
  "message": "Password reset successfully"
}
```

Response 400:

```json
{
  "error": "invalid_token",
  "message": "Reset token is invalid or expired"
}
```

Change Password

```
POST /auth/password/change
Authorization: Bearer <access_token>
Content-Type: application/json
```

Request Body:

```json
{
  "current_password": "OldPassword123!",
  "new_password": "NewSecurePassword456!",
  "new_password_confirmation": "NewSecurePassword456!"
}
```

Response 200:

```json
{
  "success": true,
  "message": "Password changed successfully",
  "sessions_revoked": true
}
```

Validate Password Strength

```
POST /auth/password/validate
Content-Type: application/json
```

Request Body:

```json
{
  "password": "TestPassword123!"
}
```

Response 200:

```json
{
  "valid": true,
  "score": 4,
  "requirements": {
    "min_length": { "required": 12, "met": true },
    "uppercase": { "required": true, "met": true },
    "lowercase": { "required": true, "met": true },
    "number": { "required": true, "met": true },
    "special": { "required": true, "met": true }
  },
  "suggestions": []
}
```

Multi-Factor Authentication

Get MFA Status

```
GET /auth/mfa/status
Authorization: Bearer <access_token>
```

Response 200:

```json
{
  "enabled": true,
  "methods": [
    {
      "type": "totp",
      "enabled": true,
      "configured_at": "2024-06-15T10:30:00Z"
    }
  ],
  "backup_codes_remaining": 8,
  "trusted_devices": 2
}
```

Setup TOTP

```
POST /auth/mfa/totp/setup
Authorization: Bearer <access_token>
```

Response 200:

```json
{
  "secret": "JBSWY3DPEHPK3PXP",
  "qr_code": "data:image/png;base64,iVBORw0KGgo...",
  "otpauth_uri": "otpauth://totp/RADIANT:user@example.com?secret=JBSWY3DPEHPK3PXP&issuer=RADIA
}
```

Verify and Enable TOTP

```
POST /auth/mfa/totp/verify
Authorization: Bearer <access_token>
Content-Type: application/json
```

Request Body:

```json
{
  "code": "123456"
}
```

Response 200:

```json
{
  "success": true,
  "backup_codes": [
    "ABCD1234",
    "EFGH5678",
    "IJKL9012",
    "MNOP3456",
    "QRST7890",
    "UVWX1234",
    "YZAB5678",
    "CDEF9012",
    "GHIJ3456",
    "KLMN7890"
  ],
  "message": "MFA enabled. Save your backup codes securely."
}
```

Verify MFA Code (During Sign-In)

```
POST /auth/mfa/verify
Content-Type: application/json
```

Request Body:

```
{
  "session": "session_token_from_signin",
  "code": "123456",
  "trust_device": true
}
```

Response 200:

```
{
  "access_token": "eyJhbGciOiJSUzI1NiIs...",
  "refresh_token": "dGhpcyBpcyBhIHJlZnJlc2g...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "device_trusted": true
}
```

Use Backup Code

```
POST /auth/mfa/backup/verify
Content-Type: application/json
```

Request Body:

```
{
  "session": "session_token_from_signin",
  "backup_code": "ABCD1234"
}
```

Response 200:

```
{
  "access_token": "eyJhbGciOiJSUzI1NiIs...",
  "refresh_token": "dGhpcyBpcyBhIHJlZnJlc2g...",
  "backup_codes_remaining": 7,
  "message": "Backup code used. 7 codes remaining."
}
```

Regenerate Backup Codes

```
POST /auth/mfa/backup/regenerate
Authorization: Bearer <access_token>
Content-Type: application/json
```

Request Body:

```json
{
  "mfa_code": "123456"
}
```

Response 200:

```json
{
  "backup_codes": [
    "NEWC1234",
    "ODES5678",
    "..."
  ],
  "message": "New backup codes generated. Previous codes are now invalid."
}
```

Disable MFA

```
POST /auth/mfa/disable
Authorization: Bearer <access_token>
Content-Type: application/json
```

Request Body:

```json
{
  "mfa_code": "123456",
  "password": "CurrentPassword123!"
}
```

Response 200:

```json
{
  "success": true,
  "message": "MFA disabled"
}
```

Response 403:

```json
{
  "error": "mfa_required",
  "message": "MFA cannot be disabled for your account type"
}
```

Get Trusted Devices

```
GET /auth/mfa/devices
Authorization: Bearer <access_token>
```

Response 200:

```
{
  "devices": [
    {
      "id": "dev_abc123",
      "name": "Chrome on MacOS",
      "last_used": "2026-01-25T08:00:00Z",
      "trusted_at": "2026-01-20T10:00:00Z",
      "expires_at": "2026-02-19T10:00:00Z",
      "current": true
    },
    {
      "id": "dev_def456",
      "name": "Safari on iPhone",
      "last_used": "2026-01-24T15:30:00Z",
      "trusted_at": "2026-01-15T09:00:00Z",
      "expires_at": "2026-02-14T09:00:00Z",
      "current": false
    }
  ]
}
```

Remove Trusted Device

```
DELETE /auth/mfa/devices/{device_id}
Authorization: Bearer <access_token>
```

Response 200:

```
{
  "success": true,
  "message": "Device removed from trusted list"
}
```

Session Management

List Active Sessions

```
GET /auth/sessions
Authorization: Bearer <access_token>
```

Response 200:

```
{
  "sessions": [
    {
      "id": "sess_abc123",
      "device": "Chrome on MacOS",
      "ip_address": "192.168.1.100",
      "location": "New York, US",
      "created_at": "2026-01-20T10:00:00Z",
      "last_activity": "2026-01-25T08:30:00Z",
      "current": true
    },
    {
      "id": "sess_def456",
      "device": "Safari on iPhone",
      "ip_address": "10.0.0.50",
      "location": "New York, US",
      "created_at": "2026-01-22T14:00:00Z",
      "last_activity": "2026-01-24T18:00:00Z",
      "current": false
    }
  ]
}
```

Revoke Session

```
DELETE /auth/sessions/{session_id}
Authorization: Bearer <access_token>
```

Response 200:

```
{
  "success": true,
  "message": "Session revoked"
}
```

Revoke All Other Sessions

```
POST /auth/sessions/revoke-others
Authorization: Bearer <access_token>
```

Response 200:

```
{
  "success": true,
  "revoked_count": 3,
  "message": "3 sessions revoked"
}
```

OAuth 2.0 Endpoints

Authorization Endpoint

```
GET /oauth/authorize
```

Query Parameters:

Parameter	Required	Description
`client_id`	Yes	Application client ID
`redirect_uri`	Yes	Callback URL (must match registration)
`response_type`	Yes	`code`
`scope`	Yes	Space-separated scopes
`state`	Yes	CSRF protection token
`code_challenge`	Yes*	PKCE challenge (S256)
`code_challenge_method`	Yes*	`S256`

Example:

```
GET /oauth/authorize?client_id=app_123&redirect_uri=https://myapp.com/callback&response_type=c
```

Token Endpoint

```
POST /oauth/token
Content-Type: application/x-www-form-urlencoded
```

Authorization Code Grant:

```
grant_type=authorization_code
&code=AUTH_CODE
&redirect_uri=https://myapp.com/callback
&client_id=app_123
&client_secret=secret_456
&code_verifier=PKCE_VERIFIER
```

Refresh Token Grant:

```
grant_type=refresh_token
&refresh_token=REFRESH_TOKEN
&client_id=app_123
&client_secret=secret_456
```

Client Credentials Grant:

```
grant_type=client_credentials
&client_id=app_123
&client_secret=secret_456
&scope=read:data
```

Response 200:

```json
{
  "access_token": "eyJhbGciOiJSUzI1NiIs...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_token": "dGhpcyBpcyBhIHJlZnJlc2g...",
  "scope": "openid profile"
}
```

Token Introspection

```
POST /oauth/introspect
Content-Type: application/x-www-form-urlencoded
Authorization: Basic base64(client_id:client_secret)
```

Request:

```
token=ACCESS_TOKEN
```

Response 200:

```json
{
  "active": true,
  "client_id": "app_123",
  "scope": "openid profile",
  "sub": "usr_abc123",
  "exp": 1706234567,
  "iat": 1706230967,
  "token_type": "Bearer"
}
```

Token Revocation

```
POST /oauth/revoke
Content-Type: application/x-www-form-urlencoded
Authorization: Basic base64(client_id:client_secret)
```

Request:

```
token=REFRESH_TOKEN
&token_type_hint=refresh_token
```

Response 200:

```json
{
  "success": true
}
```

UserInfo Endpoint (OIDC)

```
GET /oauth/userinfo
Authorization: Bearer <access_token>
```

Response 200:

```json
{
  "sub": "usr_abc123def456",
  "name": "Jane Doe",
  "given_name": "Jane",
  "family_name": "Doe",
  "email": "user@example.com",
  "email_verified": true,
  "picture": "https://cdn.radiant.ai/avatars/abc123.png"
}
```

User Profile

Update Profile

```
PATCH /auth/profile
Authorization: Bearer <access_token>
Content-Type: application/json
```

Request Body:

```json
{
  "name": "Jane Smith",
  "language": "es",
  "timezone": "Europe/Madrid"
}
```

Response 200:

```json
{
  "id": "usr_abc123def456",
  "email": "user@example.com",
  "name": "Jane Smith",
  "language": "es",
  "timezone": "Europe/Madrid",
  "updated_at": "2026-01-25T10:00:00Z"
}
```

Update Avatar

```
POST /auth/profile/avatar
Authorization: Bearer <access_token>
Content-Type: multipart/form-data
```

Request:

```
avatar: (binary file, max 5MB, PNG/JPG/GIF)
```

Response 200:

```json
{
  "avatar_url": "https://cdn.radiant.ai/avatars/new_abc123.png"
}
```

Error Responses

Error Format

All errors follow this format:

```json
{
  "error": "error_code",
  "message": "Human-readable message",
  "details": { },
  "request_id": "req_abc123"
}
```

Error Codes

Code	HTTP Status	Description
`invalid_credentials`	401	Email or password incorrect
`account_locked`	403	Too many failed attempts
`account_suspended`	403	Account has been suspended
`mfa_required`	401	MFA verification needed
`mfa_invalid`	401	MFA code incorrect
`session_expired`	401	Session has expired
`token_invalid`	401	Token is invalid or expired
`token_revoked`	401	Token has been revoked
`insufficient_scope`	403	Token lacks required scope
`rate_limited`	429	Too many requests
`invalid_request`	400	Request validation failed
`not_found`	404	Resource not found
`server_error`	500	Internal server error

Validation Errors

```json
{
  "error": "validation_error",
  "message": "Validation failed",
  "details": {
    "fields": {
      "email": ["Invalid email format"],
      "password": ["Must be at least 12 characters"]
    }
  }
}
```

Related Documentation

- Authentication Overview
- OAuth Developer Guide
- Security Architecture