# **User Authentication API Documentation**

## **Base URL**

https://api.yoraa.in/api/auth

## **1. User Signup**

**Endpoint:** /signup  
 **Method:** POST  
 **Description:** Registers a new user and sends an OTP for verification.

### **Request Body (JSON)**

{

"phNo": "9876543210",

"password": "securePassword123"

}

### **Response**

✅ **201 Created**

{

"success": true,

"message": "Signup successful. OTP sent successfully.",

"data": null

}

## **2. Verify OTP**

**Endpoint:** /verify-otp  
 **Method:** POST  
 **Description:** Verifies the OTP and activates the user account.

### **Request Body (JSON)**

{

"phNo": "9876543210",

"otp": "1234"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "OTP verified successfully",

"data": {

"token": "jwt-token-here",

"user": { "phNo": "9876543210", "isVerified": true }

}

}

## **3. Resend OTP**

**Endpoint:** /generate-otp  
 **Method:** POST  
 **Description:** Generates and resends an OTP for user verification.

### **Request Body (JSON)**

{

"phNo": "9876543210"

}

### **Response**

✅ **201 Created**

{

"success": true,

"message": "OTP sent successfully",

"data": null

}

## **4. Login**

**Endpoint:** /login  
 **Method:** POST  
 **Description:** Authenticates a user using phone number and password.

### **Request Body (JSON)**

{

"phNo": "9876543210",

"password": "securePassword123"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "Login successful",

"data": {

"token": "jwt-token-here",

"user": { "phNo": "9876543210", "isVerified": true }

}

}

## **5. Firebase Signup**

**Endpoint:** /signup-firebase  
 **Method:** POST  
 **Description:** Registers a user using Firebase authentication.

### **Request Body (JSON)**

{

"idToken": "firebase-id-token"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "OTP verified successfully",

"data": {

"token": "jwt-token-here",

"user": { "phNo": "+919876543210", "isVerified": true }

}

}

## **6. Firebase Login**

**Endpoint:** /login-firebase  
 **Method:** POST  
 **Description:** Logs in a user via Firebase authentication.

### **Request Body (JSON)**

{

"idToken": "firebase-id-token"

}

### **Response**

✅ **200 OK**

{

"token": "jwt-token-here"

}

## **7. Verify Firebase OTP**

**Endpoint:** /verify-firebase-otp  
 **Method:** POST  
 **Description:** Verifies the Firebase OTP and registers/logs in the user.

### **Request Body (JSON)**

{

"idToken": "firebase-id-token",

"phNo": "+919876543210"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "OTP verified successfully",

"data": {

"token": "jwt-token-here",

"user": { "phNo": "+919876543210", "isVerified": true }

}

}

## **8. Send Email Verification OTP**

**Endpoint:** /send-verification-email  
 **Method:** POST  
 **Description:** Sends an OTP to the user's email for verification.

### **Request Body (JSON)**

{

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

"phone": "9876543210"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "OTP sent to email",

"data": null

}

## **9. Verify Email OTP**

**Endpoint:** /verify-email  
 **Method:** POST  
 **Description:** Verifies the OTP sent to the user's email.

### **Request Body (JSON)**

{

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

"otp": "123456",

"phone": "9876543210"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "Email verified successfully",

"data": null

}

## **10. Reset Password**

**Endpoint:** /reset-password  
 **Method:** POST  
 **Description:** Allows a user to reset their password.

### **Request Body (JSON)**

{

"phNo": "9876543210",

"newPassword": "newSecurePassword123"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "Password reset successfully",

"data": null

}

## **11. Logout**

**Endpoint:** /logout  
 **Method:** POST  
 **Description:** Logs out the user by clearing the authentication token.

### **Response**

✅ **200 OK**

{

"message": "Logout successful"

}

## **12. Delete User Account**

**Endpoint:** /delete-user  
 **Method:** DELETE  
 **Description:** Deletes the authenticated user's account.

### **Headers**

{

"Authorization": "Bearer jwt-token-here"

}

### **Response**

✅ **200 OK**

{

"success": true,

"message": "User deleted successfully",

"data": null

}

# **Category API Documentation**

## **Base URL**

https://api.yoraa.in/api/categories

## **Create a Category**

### **Endpoint:**

POST /

### **Description:**

Creates a new category.

### **Request Body: Multipart/formData**

{

"name": "Electronics",

"description": "All electronic gadgets",

"imageUrl": "https://s3-bucket-url.com/category-image.jpg"

}

It will generate the aws s3 image url

### **Response:**

#### **Success (201 Created)**

{

"success": true,

"status": 201,

"message": "Category created successfully",

"data": {

"\_id": "65f8d5e25b3d2a001c7f8b22",

"name": "Electronics",

"description": "All electronic gadgets",

"imageUrl": "https://s3-bucket-url.com/category-image.jpg",

"createdAt": "2025-02-25T12:34:56.789Z",

"updatedAt": "2025-02-25T12:34:56.789Z"

}

}

#### **Error (400 Bad Request)**

{

"success": false,

"status": 400,

"message": "Image is required",

"data": null

}

## **Get All Categories**

### **Endpoint:**

GET /

### **Description:**

Fetches all categories.

### **Response:**

#### **Success (200 OK)**

{

"success": true,

"status": 200,

"message": "Categories fetched successfully",

"data": [

{

"\_id": "65f8d5e25b3d2a001c7f8b22",

"name": "Electronics",

"description": "All electronic gadgets",

"imageUrl": "https://s3-bucket-url.com/category-image.jpg"

}

]

}

## **Get Category by ID**

### **Endpoint:**

GET /:id

### **Description:**

Fetches a single category by ID.

### **Response:**

#### **Success (200 OK)**

{

"success": true,

"status": 200,

"message": "Category fetched successfully",

"data": {

"\_id": "65f8d5e25b3d2a001c7f8b22",

"name": "Electronics",

"description": "All electronic gadgets",

"imageUrl": "https://s3-bucket-url.com/category-image.jpg"

}

}

#### **Error (404 Not Found)**

{

"success": false,

"status": 404,

"message": "Category not found",

"data": null

}

## **Update a Category**

### **Endpoint:**

PUT /:id

### **Description:**

Updates an existing category.

### **Request Body: MultiPart/formdata**

{

"name": "Updated Electronics",

"description": "Updated description",

"imageUrl": "https://s3-bucket-url.com/updated-category-image.jpg"

}

It will generate the aws s3 image url

### **Response:**

#### **Success (200 OK)**

{

"success": true,

"status": 200,

"message": "Category updated successfully",

"data": {

"\_id": "65f8d5e25b3d2a001c7f8b22",

"name": "Updated Electronics",

"description": "Updated description",

"imageUrl": "https://s3-bucket-url.com/updated-category-image.jpg"

}

}

## **Delete a Category**

### **Endpoint:**

DELETE /:id

### **Description:**

Deletes a category by ID, along with related subcategories and items.

### **Response:**

#### **Success (200 OK)**

{

"success": true,

"status": 200,

"message": "Category deleted successfully",

"data": null

}

#### **Error (404 Not Found)**

{

"success": false,

"status": 404,

"message": "Category not found",

"data": null

}

## **Get Total Categories Count**

### **Endpoint:**

GET /total

### **Description:**

Fetches the total number of categories.

### **Response:**

#### **Success (200 OK)**

{

"success": true,

"status": 200,

"message": "Total categories count fetched successfully",

"data": {

"totalCategories": 5

}

}

# **SubCategory API Documentation**

## **Base URL**

https://api.yoraa.in/api/subcategories

## **Create a SubCategory**

**Endpoint:**

POST /

**Description:** Create a new subcategory.

**Request Headers:**

Content-Type: application/json

**Request Body:MultiPart/FormData**

{

"name": "Electronics",

"description": "Electronic devices and accessories",

"categoryId": "65f2a0d48b12345f67891011",

"imageUrl": "https://your-s3-bucket.com/subcategory-image.jpg"

}

It will generate the aws s3 image url

**Response:**

{

"success": true,

"message": "Subcategory created successfully",

"data": {

"\_id": "65f2a0d48b12345f67891011",

"name": "Electronics",

"description": "Electronic devices and accessories",

"categoryId": "65f2a0d48b12345f67891011",

"imageUrl": "https://your-s3-bucket.com/subcategory-image.jpg"

}

}

## **Get All SubCategories**

**Endpoint:**

GET /

**Description:** Retrieve a list of all subcategories.

**Response:**

{

"success": true,

"message": "Subcategories retrieved successfully",

"data": [

{

"\_id": "65f2a0d48b12345f67891011",

"name": "Electronics",

"description": "Electronic devices and accessories",

"categoryId": "65f2a0d48b12345f67891011",

"imageUrl": "https://your-s3-bucket.com/subcategory-image.jpg"

}

]

}

## **Get SubCategory by ID**

**Endpoint:**

GET /:id

**Description:** Retrieve a single subcategory by its ID.

**Response:**

{

"success": true,

"message": "Subcategory fetched successfully",

"data": {

"\_id": "65f2a0d48b12345f67891011",

"name": "Electronics",

"description": "Electronic devices and accessories",

"categoryId": "65f2a0d48b12345f67891011",

"imageUrl": "https://your-s3-bucket.com/subcategory-image.jpg"

}

}

## **Get SubCategories by Category ID**

**Endpoint:**

GET /category/:categoryId

**Description:** Retrieve all subcategories under a specific category.

**Response:**

{

"success": true,

"message": "Subcategories retrieved successfully",

"data": [

{

"\_id": "65f2a0d48b12345f67891011",

"name": "Electronics",

"description": "Electronic devices and accessories",

"categoryId": "65f2a0d48b12345f67891011",

"imageUrl": "https://your-s3-bucket.com/subcategory-image.jpg"

}

]

}

## **Update a SubCategory**

**Endpoint:**

PUT /:id

**Description:** Update a subcategory's details.

**Request Body:**

{

"name": "Updated Electronics",

"description": "Updated description",

"imageUrl": "https://your-s3-bucket.com/updated-image.jpg"

}

**Response:**

{

"success": true,

"message": "Subcategory updated successfully",

"data": {

"\_id": "65f2a0d48b12345f67891011",

"name": "Updated Electronics",

"description": "Updated description",

"categoryId": "65f2a0d48b12345f67891011",

"imageUrl": "https://your-s3-bucket.com/updated-image.jpg"

}

}

## **Delete a SubCategory**

**Endpoint:**

DELETE /:id

**Description:** Delete a subcategory by ID.

**Response:**

{

"success": true,

"message": "Subcategory deleted successfully"

}

## **Get Total SubCategory Count**

**Endpoint:**

GET /total

**Description:** Retrieve the total number of subcategories.

**Response:**

{

"success": true,

"message": "Total subcategories count fetched successfully",

"data": {

"totalSubCategories": 10

}

}

# **Item API Documentation**

## **Base URL**

https://api.yoraa.in/api/items

## **Endpoints**

### **1. Create a New Item**

**Endpoint:**

POST /api/items

**Description:** Creates a new item in the database.

**Request Body: MultiPart/FormData**

{

"data": "

{

"name": "Item Name",

"description": "Item Description",

"price": 100, "stock": 10, "brand": "Brand Name", "style": ["Casual"], "occasion": ["Party"], "fit": ["Regular"], "material": ["Cotton"], "discountPrice": 90 }

",

"subCategoryId": "subCategoryObjectId",

"categoryId": "categoryObjectId",

"imageUrl": "https://s3-bucket-url/item-image.jpg"

}

**Response:**

{

"success": true,

"message": "Item created successfully",

"statusCode": 201,

"data": { ... }

}

### **2. Get Items by SubCategory**

**Endpoint:**

POST /api/items/by-subcategory/:subCategoryId

**Description:** Fetches items belonging to a specific subcategory with filtering and pagination.

**Request Body:**

{

"page": 1,

"limit": 10,

"filters": {

"brand": ["Brand Name"],

"style": ["Casual"],

"minPrice": 50,

"maxPrice": 200,

"minRating": 3

}

}

**Response:**

{

"success": true,

"message": "Items fetched successfully",

"statusCode": 200,

"data": [...],

"pagination": { "totalPages": 3, "currentPage": 1 }

}

### **3. Get Items by Filter**

**Endpoint:**

POST /api/items/filter

**Description:** Fetches items based on multiple filters including categories, subcategories, brand, price, etc.

**Request Body:**

{

"page": 1,

"limit": 10,

"filters": {

"categoryId": "categoryObjectId",

"subCategoryId": "subCategoryObjectId",

"brand": ["Brand Name"],

"minPrice": 50,

"maxPrice": 200,

"minRating": 3

},

"searchText": "Shirt"

}

**Response:**

{

"success": true,

"message": "Items fetched successfully",

"statusCode": 200,

"data": [...],

"pagination": { "totalPages": 5, "currentPage": 1 }

}

### **4. Get All Items**

**Endpoint:**

GET /api/items

**Description:** Fetches all items with pagination.

**Query Parameters:**

?page=1&limit=10

**Response:**

{

"items": [...],

"totalPages": 3,

"currentPage": 1

}

### **5. Get Single Item by ID**

**Endpoint:**

GET /api/items/:id

**Description:** Fetches details of a single item by its ID.

**Response:**

{

"\_id": "itemId",

"name": "Item Name",

"description": "Item Description",

"price": 100,

"subCategoryId": { ... }

}

### **6. Update an Item**

**Endpoint:**

PUT /api/items/:id

**Description:** Updates an existing item by its ID.

**Request Body:**

{

"name": "Updated Name",

"description": "Updated Description",

"price": 120,

"stock": 15,

"subCategoryId": "newSubCategoryId",

"imageUrl": "https://new-image-url.jpg"

}

**Response:**

{

"\_id": "itemId",

"name": "Updated Name",

"description": "Updated Description",

"price": 120,

"stock": 15,

"subCategoryId": "newSubCategoryId"

}

### **7. Delete an Item**

**Endpoint:**

DELETE /api/items/:id

**Description:** Deletes an item and its image from S3.

**Response:**

{

"message": "Item deleted successfully"

}

### **8. Get Latest Items by SubCategory**

**Endpoint:**

GET /api/items/latest/:subCategoryId

**Description:** Fetches the latest items in a given subcategory, sorted by creation date.

**Query Parameters:**

?page=1&limit=10

**Response:**

{

"success": true,

"message": "Latest items fetched successfully",

"statusCode": 200,

"data": [...],

"pagination": { "totalPages": 2, "currentPage": 1 }

}

### **9. Get Total Item Count**

**Endpoint:**

GET /api/items/count

**Description:** Fetches the total number of items in the database.

**Response:**

{

"success": true,

"message": "Total item count fetched successfully",

"statusCode": 200,

"data": { "totalItems": 120 }

}

# **ItemDetails API Documentation**

## **Overview**

The ItemDetails API provides CRUD operations for managing detailed information about items, including descriptions, fit details, care instructions, manufacturer details, images, and dimensions. It integrates AWS S3 for image storage.

## **Endpoints**

### **1. Create ItemDetails**

**Endpoint:**

POST /api/item-details/:itemId

**Description:** Creates a new ItemDetails entry for the given itemId, uploads images to AWS S3, and updates the Item to reflect the presence of item details.

**Request Parameters:**

* itemId (Path Parameter) - ID of the item to associate details with.

**Request Body: MultiPart/FormData**

data:{

"descriptionAndReturns": "An urban essential designed for the modern gentleman. Crafted for comfort and versatility, these trousers redefine your everyday wardrobe. From desk to dinner, embrace a timeless aesthetic that exudes confidence and refinement. Elevate your urban sophistication with every stride.",

"fitDetails": [

"Regular Fit",

"Mid Rise",

"8-way Stretch Fabric",

"4 Pocket Detailing",

"Button and Zipper Closure"

],

"careInstructions": "7 Days No Wash & No Iron",

"size": {

"modelHeight": "Model height 188cm.",

"modelMeasurements": "Chest-39, Waist-32, Hips-38",

"modelWearingSize": "Size M"

},

"manufacturerDetails": {

"name": "Radhamani Textiles Private Limited",

"address": "Address 27, VTMS Arcade, Bomannahalli, Bengaluru, 560068",

"countryOfOrigin": "India",

"contactDetails": {

"phone": "+91 80 66085236",

"email": "support@thehouseofrare.com"

}

},

"shippingAndReturns": {

"shippingDetails": [

"50 Rs shipping charges will be charged on Orders Below 2500Rs.",

"100Rs COD(Cash on Delivery) Charges will be charged on COD Orders Below 3000Rs.",

"For international orders, customs duties may be levied at the time of delivery in certain countries.",

"Product are shipped from our warehouse within 4 working days.",

"The order will be delivered in 10 working days.",

"You will receive order tracking number as soon as we ship your order."

],

"returnPolicy": [

"We have 7 days return policy (only for returnable products).",

"Please ensure that the products you return are unused, unworn and the original tags are intact.",

"A non-refundable charge of Rs. 100 will be applied on all Cash on Delivery (COD) orders below Rs. 3,000.",

"Shipping charge and COD Charge are Non Refundable.",

"International orders are not eligible for return.",

"Once the product is picked, refund will be initiated in 3 working days."

]

}

}

* data (String) - JSON string containing item details.
* images(Form-data) - Array of image files to upload(max 4).

**Response:**

{

"\_id": "12345",

"descriptionAndReturns": "Product details...",

"fitDetails": ["Regular Fit"],

"images": ["https://s3.amazonaws.com/..."],

"items": "67890"

}

**Error Responses:**

* 400 - Invalid JSON format or item name already exists.
* 404 - Item not found.

### **2. Get All ItemDetails**

**Endpoint:**

GET /api/item-details

**Description:** Retrieves all ItemDetails entries, populating the associated items field.

**Response:**

[

{

"\_id": "12345",

"descriptionAndReturns": "Product details...",

"items": { "\_id": "67890", "name": "Item Name" }

}

]

**Error Responses:**

* 500 - Server error.

### **3. Get ItemDetails by Item ID**

**Endpoint:**

GET /api/item-details/:itemId

**Description:** Retrieves ItemDetails for a specific itemId.

**Response:**

{

"\_id": "12345",

"descriptionAndReturns": "Product details...",

"items": { "\_id": "67890", "name": "Item Name" }

}

**Error Responses:**

* 404 - Item details not found.
* 500 - Server error.

### **4. Update ItemDetails**

**Endpoint:**

PUT /api/item-details/:itemId

**Description:** Updates ItemDetails for a given itemId. If new images are uploaded, it replaces existing images in S3.

**Request Body:**

{

"data": "{ \"descriptionAndReturns\": \"Updated details...\", \"fitDetails\": [\"Slim Fit\"] }"

}

* data (String) - JSON string containing updated item details.
* files (Form-data) - New images to replace existing ones.

**Response:**

{

"\_id": "12345",

"descriptionAndReturns": "Updated details...",

"fitDetails": ["Slim Fit"],

"images": ["https://s3.amazonaws.com/new\_image.jpg"],

"items": "67890"

}

**Error Responses:**

* 400 - Invalid JSON format.
* 404 - Item or ItemDetails not found.
* 500 - Server error.

### **5. Delete ItemDetails**

**Endpoint:**

DELETE /api/item-details/:id

**Description:** Deletes an ItemDetails entry and removes associated images from S3.

**Response:**

{

"message": "Item Details deleted successfully"

}

**Error Responses:**

* 404 - ItemDetails not found.
* 500 - Server error.

### **6. Delete Image from ItemDetails**

**Endpoint:**

DELETE /api/item-details/:itemId/image

**Description:** Deletes a specific image from ItemDetails in S3 and updates the database.

**Request Body:**

{

"imageUrl": "https://s3.amazonaws.com/image.jpg"

}

**Response:**

{

"message": "Image deleted successfully",

"updatedImages": ["https://s3.amazonaws.com/new\_image.jpg"]

}

**Error Responses:**

* 400 - Image URL is required.
* 404 - ItemDetails or image not found.
* 500 - Server error.

## **Notes**

* All image uploads and deletions are handled using AWS S3.
* ItemDetails is always associated with an Item.
* Error handling ensures proper JSON parsing and validation.