**API Specifications for E-Commerce Website**

**Authentication Details**

* **JWT Authentication**: All endpoints requiring authentication expect a JWT token in the request header:
* Authorization: Bearer <jwt\_token>
* **Token Usage**: The token is generated upon successful login (POST /auth/login for customers, POST /admin/auth/login for admins) and must be included in every subsequent authenticated request. The backend verifies the token to extract user details (e.g., customer ID, admin role) without requiring additional user IDs in the request.
* **Token Expiry**: Tokens should have a reasonable expiry (e.g., 1 hour), with a refresh token mechanism (not detailed here but can be added if needed).
* **Error Handling**: Invalid or expired tokens return:
* {
* "error": "Invalid or expired token",
* "status": 401
* }

**Customer-Centric API Endpoints**

1. **GET /products**
   * **Functionality**: Retrieve a list of all available products with filtering and sorting.
   * **Authentication**: None (public endpoint).
   * **Query Parameters**:
     + category: String (e.g., "textiles") - Filter by category.
     + price\_min: Number - Minimum price filter.
     + price\_max: Number - Maximum price filter.
     + location: String (e.g., "village\_name") - Filter by producer location.
     + sort: String (e.g., "price\_asc", "price\_desc", "newest") - Sort order.
     + page: Number - Pagination page (default: 1).
     + limit: Number - Items per page (default: 20).
   * **Request Example**:
   * GET /products?category=textiles&price\_min=100&sort=price\_asc&page=1&limit=10
   * **Response Schema** (200 OK):
   * {
   * "products": [
   * {
   * "id": "string",
   * "name": "string",
   * "description": "string",
   * "price": "number",
   * "category": "string",
   * "images": ["string"],
   * "stock": "number",
   * "producer": {
   * "id": "string",
   * "name": "string",
   * "village": "string"
   * },
   * "rating": "number"
   * }
   * ],
   * "total": "number",
   * "page": "number",
   * "limit": "number"
   * }
   * **Error Response** (400 Bad Request):
   * {
   * "error": "Invalid query parameters",
   * "status": 400
   * }
2. **GET /products/:id**
   * **Functionality**: Fetch details of a specific product.
   * **Authentication**: None (public endpoint).
   * **Path Parameters**:
     + id: Product ID (string).
   * **Request Example**:
   * GET /products/12345
   * **Response Schema** (200 OK):
   * {
   * "id": "string",
   * "name": "string",
   * "description": "string",
   * "price": "number",
   * "category": "string",
   * "images": ["string"],
   * "stock": "number",
   * "producer": {
   * "id": "string",
   * "name": "string",
   * "village": "string",
   * "story": "string"
   * },
   * "rating": "number",
   * "reviews": [
   * {
   * "id": "string",
   * "customer\_name": "string",
   * "rating": "number",
   * "comment": "string",
   * "created\_at": "string"
   * }
   * ]
   * }
   * **Error Response** (404 Not Found):
   * {
   * "error": "Product not found",
   * "status": 404
   * }
3. **GET /products/search**
   * **Functionality**: Search products by keyword with filters.
   * **Authentication**: None (public endpoint).
   * **Query Parameters**:
     + query: String - Search keyword.
     + category: String - Optional category filter.
     + page: Number - Pagination page (default: 1).
     + limit: Number - Items per page (default: 10).
   * **Request Example**:
   * GET /products/search?query=handwoven&page=1&limit=5
   * **Response Schema** (200 OK):
   * {
   * "products": [
   * {
   * "id": "string",
   * "name": "string",
   * "price": "number",
   * "category": "string",
   * "images": ["string"]
   * }
   * ],
   * "total": "number",
   * "page": "number",
   * "limit": "number"
   * }
   * **Error Response** (400 Bad Request):
   * {
   * "error": "Invalid search query",
   * "status": 400
   * }
4. **POST /auth/register**
   * **Functionality**: Register a new customer account.
   * **Authentication**: None (public endpoint).
   * **Request Schema**:
   * {
   * "email": "string",
   * "password": "string",
   * "phone": "string",
   * "name": "string"
   * }
   * **Request Example**:
   * POST /auth/register
   * Content-Type: application/json
   * {
   * "email": "user@example.com",
   * "password": "Password123!",
   * "phone": "+919876543210",
   * "name": "John Doe"
   * }
   * **Response Schema** (201 Created):
   * {
   * "message": "Registration successful",
   * "customer\_id": "string"
   * }
   * **Error Response** (409 Conflict):
   * {
   * "error": "Email or phone already exists",
   * "status": 409
   * }
5. **POST /auth/login**
   * **Functionality**: Authenticate a customer and return a JWT token.
   * **Authentication**: None (public endpoint).
   * **Request Schema**:
   * {
   * "email": "string",
   * "password": "string"
   * }
   * **Request Example**:
   * POST /auth/login
   * Content-Type: application/json
   * {
   * "email": "user@example.com",
   * "password": "Password123!"
   * }
   * **Response Schema** (200 OK):
   * {
   * "token": "string",
   * "customer\_id": "string",
   * "name": "string"
   * }
   * **Error Response** (401 Unauthorized):
   * {
   * "error": "Invalid credentials",
   * "status": 401
   * }
6. **GET /auth/profile**
   * **Functionality**: Retrieve customer profile details.
   * **Authentication**: JWT required.
   * **Request Example**:
   * GET /auth/profile
   * Authorization: Bearer <jwt\_token>
   * **Response Schema** (200 OK):
   * {
   * "id": "string",
   * "name": "string",
   * "email": "string",
   * "phone": "string",
   * "addresses": [
   * {
   * "id": "string",
   * "street": "string",
   * "city": "string",
   * "state": "string",
   * "country": "string",
   * "pincode": "string"
   * }
   * ]
   * }
   * **Error Response** (401 Unauthorized):
   * {
   * "error": "Invalid or expired token",
   * "status": 401
   * }
7. **PUT /auth/profile**
   * **Functionality**: Update customer profile information.
   * **Authentication**: JWT required.
   * **Request Schema**:
   * {
   * "name": "string",
   * "phone": "string",
   * "addresses": [
   * {
   * "id": "string",
   * "street": "string",
   * "city": "string",
   * "state": "string",
   * "country": "string",
   * "pincode": "string"
   * }
   * ]
   * }
   * **Request Example**:
   * PUT /auth/profile
   * Authorization: Bearer <jwt\_token>
   * Content-Type: application/json
   * {
   * "name": "Jane Doe",
   * "phone": "+919876543211",
   * "addresses": [
   * {
   * "id": "addr1",
   * "street": "123 Main St",
   * "city": "Mumbai",
   * "state": "Maharashtra",
   * "country": "India",
   * "pincode": "400001"
   * }
   * ]
   * }
   * **Response Schema** (200 OK):
   * {
   * "message": "Profile updated successfully"
   * }
   * **Error Response** (400 Bad Request):
   * {
   * "error": "Invalid input data",
   * "status": 400
   * }
8. **POST /cart**
   * **Functionality**: Add a product to the customer’s cart.
   * **Authentication**: JWT required.
   * **Request Schema**:
   * {
   * "product\_id": "string",
   * "quantity": "number"
   * }
   * **Request Example**:
   * POST /cart
   * Authorization: Bearer <jwt\_token>
   * Content-Type: application/json
   * {
   * "product\_id": "12345",
   * "quantity": 2
   * }
   * **Response Schema** (201 Created):
   * {
   * "message": "Product added to cart"
   * }
   * **Error Response** (400 Bad Request):
   * {
   * "error": "Invalid product or quantity",
   * "status": 400
   * }
9. **GET /cart**
   * **Functionality**: Retrieve the customer’s cart contents.
   * **Authentication**: JWT required.
   * **Request Example**:
   * GET /cart
   * Authorization: Bearer <jwt\_token>
   * **Response Schema** (200 OK):
   * {
   * "items": [
   * {
   * "product\_id": "string",
   * "name": "string",
   * "price": "number",
   * "quantity": "number",
   * "image": "string"
   * }
   * ],
   * "total\_price": "number"
   * }
   * **Error Response** (401 Unauthorized):
   * {
   * "error": "Invalid or expired token",
   * "status": 401
   * }
10. **PUT /cart**
    * **Functionality**: Update cart (change quantity or remove items).
    * **Authentication**: JWT required.
    * **Request Schema**:
    * {
    * "items": [
    * {
    * "product\_id": "string",
    * "quantity": "number"
    * }
    * ]
    * }
    * **Request Example**:
    * PUT /cart
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "items": [
    * {
    * "product\_id": "12345",
    * "quantity": 3
    * }
    * ]
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Cart updated successfully"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid product or quantity",
    * "status": 400
    * }
11. **DELETE /cart/:productId**
    * **Functionality**: Remove a specific product from the cart.
    * **Authentication**: JWT required.
    * **Path Parameters**:
      + productId: Product ID (string).
    * **Request Example**:
    * DELETE /cart/12345
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "message": "Product removed from cart"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Product not in cart",
    * "status": 404
    * }
12. **POST /orders**
    * **Functionality**: Create a new order from cart contents.
    * **Authentication**: JWT required.
    * **Request Schema**:
    * {
    * "address\_id": "string",
    * "payment\_method": "string",
    * "coupon\_code": "string"
    * }
    * **Request Example**:
    * POST /orders
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "address\_id": "addr1",
    * "payment\_method": "card",
    * "coupon\_code": "SAVE10"
    * }
    * **Response Schema** (201 Created):
    * {
    * "order\_id": "string",
    * "total\_price": "number",
    * "status": "string"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid address or payment method",
    * "status": 400
    * }
13. **GET /orders**
    * **Functionality**: Retrieve a list of customer’s orders.
    * **Authentication**: JWT required.
    * **Query Parameters**:
      + status: String (e.g., "shipped") - Filter by order status.
      + page: Number - Pagination page (default: 1).
      + limit: Number - Items per page (default: 10).
    * **Request Example**:
    * GET /orders?status=shipped&page=1&limit=5
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "orders": [
    * {
    * "id": "string",
    * "total\_price": "number",
    * "status": "string",
    * "created\_at": "string"
    * }
    * ],
    * "total": "number",
    * "page": "number",
    * "limit": "number"
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }
14. **GET /orders/:id**
    * **Functionality**: Fetch details of a specific order.
    * **Authentication**: JWT required.
    * **Path Parameters**:
      + id: Order ID (string).
    * **Request Example**:
    * GET /orders/67890
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "id": "string",
    * "items": [
    * {
    * "product\_id": "string",
    * "name": "string",
    * "price": "number",
    * "quantity": "number"
    * }
    * ],
    * "total\_price": "number",
    * "status": "string",
    * "tracking\_info": {
    * "carrier": "string",
    * "tracking\_number": "string",
    * "estimated\_delivery": "string"
    * },
    * "address": {
    * "street": "string",
    * "city": "string",
    * "state": "string",
    * "country": "string",
    * "pincode": "string"
    * },
    * "created\_at": "string"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Order not found",
    * "status": 404
    * }
15. **POST /orders/:id/cancel**
    * **Functionality**: Cancel an order if allowed.
    * **Authentication**: JWT required.
    * **Path Parameters**:
      + id: Order ID (string).
    * **Request Example**:
    * POST /orders/67890/cancel
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "message": "Order cancelled successfully"
    * }
    * **Error Response** (403 Forbidden):
    * {
    * "error": "Order cannot be cancelled",
    * "status": 403
    * }
16. **POST /payments**
    * **Functionality**: Initiate a payment for an order.
    * **Authentication**: JWT required.
    * **Request Schema**:
    * {
    * "order\_id": "string",
    * "payment\_method": "string",
    * "amount": "number"
    * }
    * **Request Example**:
    * POST /payments
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "order\_id": "67890",
    * "payment\_method": "card",
    * "amount": 1500.00
    * }
    * **Response Schema** (201 Created):
    * {
    * "payment\_id": "string",
    * "payment\_url": "string",
    * "status": "string"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid payment details",
    * "status": 400
    * }
17. **GET /payments/:orderId/status**
    * **Functionality**: Check the status of a payment.
    * **Authentication**: JWT required.
    * **Path Parameters**:
      + orderId: Order ID (string).
    * **Request Example**:
    * GET /payments/67890/status
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "order\_id": "string",
    * "payment\_id": "string",
    * "status": "string",
    * "amount": "number"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Payment not found",
    * "status": 404
    * }
18. **POST /reviews**
    * **Functionality**: Submit a review for a purchased product.
    * **Authentication**: JWT required.
    * **Request Schema**:
    * {
    * "product\_id": "string",
    * "rating": "number",
    * "comment": "string"
    * }
    * **Request Example**:
    * POST /reviews
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "product\_id": "12345",
    * "rating": 4,
    * "comment": "Great quality!"
    * }
    * **Response Schema** (201 Created):
    * {
    * "message": "Review submitted successfully"
    * }
    * **Error Response** (403 Forbidden):
    * {
    * "error": "Review not allowed for this product",
    * "status": 403
    * }
19. **GET /reviews/:productId**
    * **Functionality**: Retrieve all reviews for a product.
    * **Authentication**: None (public endpoint).
    * **Path Parameters**:
      + productId: Product ID (string).
    * **Request Example**:
    * GET /reviews/12345
    * **Response Schema** (200 OK):
    * {
    * "reviews": [
    * {
    * "id": "string",
    * "customer\_name": "string",
    * "rating": "number",
    * "comment": "string",
    * "created\_at": "string"
    * }
    * ],
    * "average\_rating": "number"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "No reviews found",
    * "status": 404
    * }
20. **POST /wishlist**
    * **Functionality**: Add a product to the customer’s wishlist.
    * **Authentication**: JWT required.
    * **Request Schema**:
    * {
    * "product\_id": "string"
    * }
    * **Request Example**:
    * POST /wishlist
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "product\_id": "12345"
    * }
    * **Response Schema** (201 Created):
    * {
    * "message": "Product added to wishlist"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid product ID",
    * "status": 400
    * }
21. **GET /wishlist**
    * **Functionality**: Retrieve the customer’s wishlist.
    * **Authentication**: JWT required.
    * **Request Example**:
    * GET /wishlist
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "wishlist": [
    * {
    * "product\_id": "string",
    * "name": "string",
    * "price": "number",
    * "image": "string"
    * }
    * ]
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }
22. **DELETE /wishlist/:productId**
    * **Functionality**: Remove a product from the wishlist.
    * **Authentication**: JWT required.
    * **Path Parameters**:
      + productId: Product ID (string).
    * **Request Example**:
    * DELETE /wishlist/12345
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "message": "Product removed from wishlist"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Product not in wishlist",
    * "status": 404
    * }
23. **GET /promotions**
    * **Functionality**: Fetch active promotions or discounts.
    * **Authentication**: None (public endpoint).
    * **Request Example**:
    * GET /promotions
    * **Response Schema** (200 OK):
    * {
    * "promotions": [
    * {
    * "id": "string",
    * "code": "string",
    * "discount": "number",
    * "valid\_until": "string"
    * }
    * ]
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "No active promotions",
    * "status": 404
    * }
24. **POST /promotions/apply**
    * **Functionality**: Apply a coupon code to the cart or order.
    * **Authentication**: JWT required.
    * **Request Schema**:
    * {
    * "code": "string",
    * "order\_id": "string"
    * }
    * **Request Example**:
    * POST /promotions/apply
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "code": "SAVE10",
    * "order\_id": "67890"
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Coupon applied successfully",
    * "discount": "number"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid or expired coupon",
    * "status": 400
    * }
25. **GET /shops/nearby**
    * **Functionality**: Retrieve nearby local shops stocking products.
    * **Authentication**: None (public endpoint).
    * **Query Parameters**:
      + latitude: Number - Customer’s latitude.
      + longitude: Number - Customer’s longitude.
      + radius: Number - Search radius in kilometers (default: 10).
    * **Request Example**:
    * GET /shops/nearby?latitude=19.0760&longitude=72.8777&radius=5
    * **Response Schema** (200 OK):
    * {
    * "shops": [
    * {
    * "id": "string",
    * "name": "string",
    * "address": "string",
    * "latitude": "number",
    * "longitude": "number",
    * "products": [
    * {
    * "product\_id": "string",
    * "name": "string"
    * }
    * ]
    * }
    * ]
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid coordinates",
    * "status": 400
    * }
26. **POST /support/tickets**
    * **Functionality**: Create a customer support ticket.
    * **Authentication**: JWT required.
    * **Request Schema**:
    * {
    * "subject": "string",
    * "description": "string",
    * "order\_id": "string"
    * }
    * **Request Example**:
    * POST /support/tickets
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "subject": "Delayed delivery",
    * "description": "Order not delivered yet",
    * "order\_id": "67890"
    * }
    * **Response Schema** (201 Created):
    * {
    * "ticket\_id": "string",
    * "message": "Ticket created successfully"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid input data",
    * "status": 400
    * }
27. **GET /support/tickets**
    * **Functionality**: Retrieve customer’s support tickets.
    * **Authentication**: JWT required.
    * **Query Parameters**:
      + status: String (e.g., "open") - Filter by ticket status.
      + page: Number - Pagination page (default: 1).
      + limit: Number - Items per page (default: 10).
    * **Request Example**:
    * GET /support/tickets?status=open&page=1&limit=5
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "tickets": [
    * {
    * "id": "string",
    * "subject": "string",
    * "status": "string",
    * "created\_at": "string"
    * }
    * ],
    * "total": "number",
    * "page": "number",
    * "limit": "number"
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }
28. **GET /producers/:id**
    * **Functionality**: Fetch details of a rural producer.
    * **Authentication**: None (public endpoint).
    * **Path Parameters**:
      + id: Producer ID (string).
    * **Request Example**:
    * GET /producers/producer1
    * **Response Schema** (200 OK):
    * {
    * "id": "string",
    * "name": "string",
    * "village": "string",
    * "story": "string",
    * "products": [
    * {
    * "id": "string",
    * "name": "string"
    * }
    * ]
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Producer not found",
    * "status": 404
    * }
29. **POST /share/product/:id**
    * **Functionality**: Generate a shareable link for a product.
    * **Authentication**: JWT required.
    * **Path Parameters**:
      + id: Product ID (string).
    * **Request Example**:
    * POST /share/product/12345
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "share\_url": "string"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Product not found",
    * "status": 404
    * }

**Admin-Centric API Endpoints**

1. **POST /admin/auth/login**
   * **Functionality**: Authenticate an admin/volunteer and return a JWT token.
   * **Authentication**: None (public endpoint).
   * **Request Schema**:
   * {
   * "email": "string",
   * "password": "string"
   * }
   * **Request Example**:
   * POST /admin/auth/login
   * Content-Type: application/json
   * {
   * "email": "admin@example.com",
   * "password": "Admin123!"
   * }
   * **Response Schema** (200 OK):
   * {
   * "token": "string",
   * "admin\_id": "string",
   * "role": "string"
   * }
   * **Error Response** (401 Unauthorized):
   * {
   * "error": "Invalid credentials",
   * "status": 401
   * }
2. **POST /admin/products**
   * **Functionality**: Add a new product.
   * **Authentication**: JWT required (admin role).
   * **Request Schema**:
   * {
   * "name": "string",
   * "description": "string",
   * "price": "number",
   * "category": "string",
   * "images": ["string"],
   * "stock": "number",
   * "producer\_id": "string"
   * }
   * **Request Example**:
   * POST /admin/products
   * Authorization: Bearer <jwt\_token>
   * Content-Type: application/json
   * {
   * "name": "Handwoven Scarf",
   * "description": "Made by artisans",
   * "price": 500.00,
   * "category": "textiles",
   * "images": ["url1", "url2"],
   * "stock": 10,
   * "producer\_id": "producer1"
   * }
   * **Response Schema** (201 Created):
   * {
   * "product\_id": "string",
   * "message": "Product added successfully"
   * }
   * **Error Response** (400 Bad Request):
   * {
   * "error": "Invalid product data",
   * "status": 400
   * }
3. **PUT /admin/products/:id**
   * **Functionality**: Update an existing product.
   * **Authentication**: JWT required (admin role).
   * **Path Parameters**:
     + id: Product ID (string).
   * **Request Schema**:
   * {
   * "name": "string",
   * "description": "string",
   * "price": "number",
   * "category": "string",
   * "images": ["string"],
   * "stock": "number",
   * "producer\_id": "string"
   * }
   * **Request Example**:
   * PUT /admin/products/12345
   * Authorization: Bearer <jwt\_token>
   * Content-Type: application/json
   * {
   * "name": "Updated Scarf",
   * "description": "New design",
   * "price": 550.00,
   * "category": "textiles",
   * "images": ["url1"],
   * "stock": 15,
   * "producer\_id": "producer1"
   * }
   * **Response Schema** (200 OK):
   * {
   * "message": "Product updated successfully"
   * }
   * **Error Response** (404 Not Found):
   * {
   * "error": "Product not found",
   * "status": 404
   * }
4. **DELETE /admin/products/:id**
   * **Functionality**: Remove a product from the catalog.
   * **Authentication**: JWT required (admin role).
   * **Path Parameters**:
     + id: Product ID (string).
   * **Request Example**:
   * DELETE /admin/products/12345
   * Authorization: Bearer <jwt\_token>
   * **Response Schema** (200 OK):
   * {
   * "message": "Product deleted successfully"
   * }
   * **Error Response** (404 Not Found):
   * {
   * "error": "Product not found",
   * "status": 404
   * }
5. **POST /admin/products/bulk**
   * **Functionality**: Upload multiple products via JSON.
   * **Authentication**: JWT required (admin role).
   * **Request Schema**:
   * {
   * "products": [
   * {
   * "name": "string",
   * "description": "string",
   * "price": "number",
   * "category": "string",
   * "images": ["string"],
   * "stock": "number",
   * "producer\_id": "string"
   * }
   * ]
   * }
   * **Request Example**:
   * POST /admin/products/bulk
   * Authorization: Bearer <jwt\_token>
   * Content-Type: application/json
   * {
   * "products": [
   * {
   * "name": "Scarf",
   * "description": "Handwoven",
   * "price": 500.00,
   * "category": "textiles",
   * "images": ["url1"],
   * "stock": 10,
   * "producer\_id": "producer1"
   * }
   * ]
   * }
   * **Response Schema** (201 Created):
   * {
   * "message": "Bulk upload successful",
   * "product\_ids": ["string"]
   * }
   * **Error Response** (400 Bad Request):
   * {
   * "error": "Invalid product data",
   * "status": 400
   * }
6. **GET /admin/products**
   * **Functionality**: Retrieve a list of all products with admin details.
   * **Authentication**: JWT required (admin role).
   * **Query Parameters**:
     + status: String (e.g., "pending") - Filter by approval status.
     + category: String - Filter by category.
     + page: Number - Pagination page (default: 1).
     + limit: Number - Items per page (default: 20).
   * **Request Example**:
   * GET /admin/products?status=pending&page=1&limit=10
   * Authorization: Bearer <jwt\_token>
   * **Response Schema** (200 OK):
   * {
   * "products": [
   * {
   * "id": "string",
   * "name": "string",
   * "price": "number",
   * "category": "string",
   * "stock": "number",
   * "status": "string"
   * }
   * ],
   * "total": "number",
   * "page": "number",
   * "limit": "number"
   * }
   * **Error Response** (401 Unauthorized):
   * {
   * "error": "Invalid or expired token",
   * "status": 401
   * }
7. **POST /admin/products/:id/approve**
   * **Functionality**: Approve a product for listing.
   * **Authentication**: JWT required (super admin role).
   * **Path Parameters**:
     + id: Product ID (string).
   * **Request Example**:
   * POST /admin/products/12345/approve
   * Authorization: Bearer <jwt\_token>
   * **Response Schema** (200 OK):
   * {
   * "message": "Product approved successfully"
   * }
   * **Error Response** (404 Not Found):
   * {
   * "error": "Product not found",
   * "status": 404
   * }
8. **GET /admin/inventory**
   * **Functionality**: Retrieve inventory status for all products.
   * **Authentication**: JWT required (admin role).
   * **Query Parameters**:
     + low\_stock: Boolean - Filter for low-stock products.
     + page: Number - Pagination page (default: 1).
     + limit: Number - Items per page (default: 20).
   * **Request Example**:
   * GET /admin/inventory?low\_stock=true&page=1&limit=10
   * Authorization: Bearer <jwt\_token>
   * **Response Schema** (200 OK):
   * {
   * "inventory": [
   * {
   * "product\_id": "string",
   * "name": "string",
   * "stock": "number",
   * "low\_stock\_alert": "boolean"
   * }
   * ],
   * "total": "number",
   * "page": "number",
   * "limit": "number"
   * }
   * **Error Response** (401 Unauthorized):
   * {
   * "error": "Invalid or expired token",
   * "status": 401
   * }
9. **PUT /admin/inventory/:productId**
   * **Functionality**: Update stock levels for a product.
   * **Authentication**: JWT required (admin role).
   * **Path Parameters**:
     + productId: Product ID (string).
   * **Request Schema**:
   * {
   * "stock": "number"
   * }
   * **Request Example**:
   * PUT /admin/inventory/12345
   * Authorization: Bearer <jwt\_token>
   * Content-Type: application/json
   * {
   * "stock": 20
   * }
   * **Response Schema** (200 OK):
   * {
   * "message": "Inventory updated successfully"
   * }
   * **Error Response** (404 Not Found):
   * {
   * "error": "Product not found",
   * "status": 404
   * }
10. **GET /admin/orders**
    * **Functionality**: Retrieve a list of all orders.
    * **Authentication**: JWT required (admin role).
    * **Query Parameters**:
      + status: String (e.g., "shipped") - Filter by order status.
      + date\_from: String - Filter by start date (ISO format).
      + date\_to: String - Filter by end date (ISO format).
      + page: Number - Pagination page (default: 1).
      + limit: Number - Items per page (default: 20).
    * **Request Example**:
    * GET /admin/orders?status=shipped&page=1&limit=10
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "orders": [
    * {
    * "id": "string",
    * "customer\_name": "string",
    * "total\_price": "number",
    * "status": "string",
    * "created\_at": "string"
    * }
    * ],
    * "total": "number",
    * "page": "number",
    * "limit": "number"
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }
11. **GET /admin/orders/:id**
    * **Functionality**: Fetch details of a specific order.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Order ID (string).
    * **Request Example**:
    * GET /admin/orders/67890
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "id": "string",
    * "customer": {
    * "id": "string",
    * "name": "string",
    * "email": "string"
    * },
    * "items": [
    * {
    * "product\_id": "string",
    * "name": "string",
    * "price": "number",
    * "quantity": "number"
    * }
    * ],
    * "total\_price": "number",
    * "status": "string",
    * "address": {
    * "street": "string",
    * "city": "string",
    * "state": "string",
    * "country": "string",
    * "pincode": "string"
    * },
    * "created\_at": "string"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Order not found",
    * "status": 404
    * }
12. **PUT /admin/orders/:id/status**
    * **Functionality**: Update the status of an order.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Order ID (string).
    * **Request Schema**:
    * {
    * "status": "string",
    * "tracking\_info": {
    * "carrier": "string",
    * "tracking\_number": "string",
    * "estimated\_delivery": "string"
    * }
    * }
    * **Request Example**:
    * PUT /admin/orders/67890/status
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "status": "shipped",
    * "tracking\_info": {
    * "carrier": "DHL",
    * "tracking\_number": "XYZ123",
    * "estimated\_delivery": "2025-06-20"
    * }
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Order status updated successfully"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid status",
    * "status": 400
    * }
13. **POST /admin/orders/:id/invoice**
    * **Functionality**: Generate an invoice for an order.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Order ID (string).
    * **Request Example**:
    * POST /admin/orders/67890/invoice
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "invoice\_url": "string"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Order not found",
    * "status": 404
    * }
14. **POST /admin/orders/:id/shipping-label**
    * **Functionality**: Generate a shipping label for an order.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Order ID (string).
    * **Request Example**:
    * POST /admin/orders/67890/shipping-label
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "label\_url": "string"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Order not found",
    * "status": 404
    * }
15. **GET /admin/analytics/sales**
    * **Functionality**: Retrieve sales reports.
    * **Authentication**: JWT required (admin role).
    * **Query Parameters**:
      + date\_from: String - Start date (ISO format).
      + date\_to: String - End date (ISO format).
      + category: String - Filter by product category.
    * **Request Example**:
    * GET /admin/analytics/sales?date\_from=2025-06-01&date\_to=2025-06-15
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "total\_revenue": "number",
    * "total\_orders": "number",
    * "top\_products": [
    * {
    * "product\_id": "string",
    * "name": "string",
    * "sales": "number"
    * }
    * ]
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid date range",
    * "status": 400
    * }
16. **GET /admin/analytics/customers**
    * **Functionality**: Fetch customer insights.
    * **Authentication**: JWT required (admin role).
    * **Query Parameters**:
      + date\_from: String - Start date (ISO format).
      + date\_to: String - End date (ISO format).
    * **Request Example**:
    * GET /admin/analytics/customers?date\_from=2025-06-01&date\_to=2025-06-15
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "total\_customers": "number",
    * "repeat\_customers": "number",
    * "top\_regions": [
    * {
    * "city": "string",
    * "orders": "number"
    * }
    * ]
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid date range",
    * "status": 400
    * }
17. **GET /admin/analytics/producers**
    * **Functionality**: Retrieve impact metrics for rural producers.
    * **Authentication**: JWT required (admin role).
    * **Query Parameters**:
      + date\_from: String - Start date (ISO format).
      + date\_to: String - End date (ISO format).
    * **Request Example**:
    * GET /admin/analytics/producers?date\_from=2025-06-01&date\_to=2025-06-15
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "total\_producers": "number",
    * "total\_sales": "number",
    * "top\_villages": [
    * {
    * "village": "string",
    * "sales": "number"
    * }
    * ]
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid date range",
    * "status": 400
    * }
18. **POST /admin/promotions**
    * **Functionality**: Create a new promotion.
    * **Authentication**: JWT required (admin role).
    * **Request Schema**:
    * {
    * "code": "string",
    * "discount": "number",
    * "valid\_until": "string"
    * }
    * **Request Example**:
    * POST /admin/promotions
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "code": "SAVE10",
    * "discount": 10.00,
    * "valid\_until": "2025-07-01"
    * }
    * **Response Schema** (201 Created):
    * {
    * "promotion\_id": "string",
    * "message": "Promotion created successfully"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid promotion data",
    * "status": 400
    * }
19. **PUT /admin/promotions/:id**
    * **Functionality**: Update an existing promotion.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Promotion ID (string).
    * **Request Schema**:
    * {
    * "code": "string",
    * "discount": "number",
    * "valid\_until": "string"
    * }
    * **Request Example**:
    * PUT /admin/promotions/promo1
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "code": "SAVE15",
    * "discount": 15.00,
    * "valid\_until": "2025-07-15"
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Promotion updated successfully"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Promotion not found",
    * "status": 404
    * }
20. **DELETE /admin/promotions/:id**
    * **Functionality**: Deactivate a promotion.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Promotion ID (string).
    * **Request Example**:
    * DELETE /admin/promotions/promo1
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "message": "Promotion deactivated successfully"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Promotion not found",
    * "status": 404
    * }
21. **GET /admin/support/tickets**
    * **Functionality**: Retrieve all customer support tickets.
    * **Authentication**: JWT required (admin role).
    * **Query Parameters**:
      + status: String (e.g., "open") - Filter by ticket status.
      + priority: String (e.g., "high") - Filter by priority.
      + page: Number - Pagination page (default: 1).
      + limit: Number - Items per page (default: 20).
    * **Request Example**:
    * GET /admin/support/tickets?status=open&page=1&limit=10
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "tickets": [
    * {
    * "id": "string",
    * "subject": "string",
    * "status": "string",
    * "priority": "string",
    * "customer\_name": "string",
    * "created\_at": "string"
    * }
    * ],
    * "total": "number",
    * "page": "number",
    * "limit": "number"
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }
22. **PUT /admin/support/tickets/:id**
    * **Functionality**: Update the status or response for a support ticket.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Ticket ID (string).
    * **Request Schema**:
    * {
    * "status": "string",
    * "response": "string"
    * }
    * **Request Example**:
    * PUT /admin/support/tickets/ticket1
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "status": "resolved",
    * "response": "Issue resolved"
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Ticket updated successfully"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Ticket not found",
    * "status": 404
    * }
23. **POST /admin/shops**
    * **Functionality**: Add a new local shop as a partner.
    * **Authentication**: JWT required (admin role).
    * **Request Schema**:
    * {
    * "name": "string",
    * "address": "string",
    * "latitude": "number",
    * "longitude": "number",
    * "products": ["string"]
    * }
    * **Request Example**:
    * POST /admin/shops
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "name": "Local Store",
    * "address": "123 Market St, Mumbai",
    * "latitude": 19.0760,
    * "longitude": 72.8777,
    * "products": ["12345"]
    * }
    * **Response Schema** (201 Created):
    * {
    * "shop\_id": "string",
    * "message": "Shop added successfully"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid shop data",
    * "status": 400
    * }
24. **PUT /admin/shops/:id**
    * **Functionality**: Update details of a local shop.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Shop ID (string).
    * **Request Schema**:
    * {
    * "name": "string",
    * "address": "string",
    * "latitude": "number",
    * "longitude": "number",
    * "products": ["string"]
    * }
    * **Request Example**:
    * PUT /admin/shops/shop1
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "name": "Updated Store",
    * "address": "456 Market St, Mumbai",
    * "latitude": 19.0760,
    * "longitude": 72.8777,
    * "products": ["12345", "67890"]
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Shop updated successfully"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Shop not found",
    * "status": 404
    * }
25. **GET /admin/shops**
    * **Functionality**: Retrieve a list of partnered shops.
    * **Authentication**: JWT required (admin role).
    * **Query Parameters**:
      + page: Number - Pagination page (default: 1).
      + limit: Number - Items per page (default: 20).
    * **Request Example**:
    * GET /admin/shops?page=1&limit=10
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "shops": [
    * {
    * "id": "string",
    * "name": "string",
    * "address": "string",
    * "products": [
    * {
    * "product\_id": "string",
    * "name": "string"
    * }
    * ]
    * }
    * ],
    * "total": "number",
    * "page": "number",
    * "limit": "number"
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }
26. **POST /admin/producers**
    * **Functionality**: Add a new rural producer profile.
    * **Authentication**: JWT required (admin role).
    * **Request Schema**:
    * {
    * "name": "string",
    * "village": "string",
    * "story": "string"
    * }
    * **Request Example**:
    * POST /admin/producers
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "name": "Arjun Patel",
    * "village": "Kutch",
    * "story": "Handweaving expert"
    * }
    * **Response Schema** (201 Created):
    * {
    * "producer\_id": "string",
    * "message": "Producer added successfully"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid producer data",
    * "status": 400
    * }
27. **PUT /admin/producers/:id**
    * **Functionality**: Update a rural producer’s profile.
    * **Authentication**: JWT required (admin role).
    * **Path Parameters**:
      + id: Producer ID (string).
    * **Request Schema**:
    * {
    * "name": "string",
    * "village": "string",
    * "story": "string"
    * }
    * **Request Example**:
    * PUT /admin/producers/producer1
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "name": "Arjun Patel",
    * "village": "Kutch",
    * "story": "Updated story"
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Producer updated successfully"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Producer not found",
    * "status": 404
    * }
28. **GET /admin/volunteers**
    * **Functionality**: Retrieve a list of volunteers.
    * **Authentication**: JWT required (super admin role).
    * **Query Parameters**:
      + role: String (e.g., "product\_manager") - Filter by role.
      + page: Number - Pagination page (default: 1).
      + limit: Number - Items per page (default: 20).
    * **Request Example**:
    * GET /admin/volunteers?role=product\_manager&page=1&limit=10
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "volunteers": [
    * {
    * "id": "string",
    * "name": "string",
    * "email": "string",
    * "role": "string"
    * }
    * ],
    * "total": "number",
    * "page": "number",
    * "limit": "number"
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }
29. **POST /admin/volunteers**
    * **Functionality**: Add a new volunteer.
    * **Authentication**: JWT required (super admin role).
    * **Request Schema**:
    * {
    * "name": "string",
    * "email": "string",
    * "password": "string",
    * "role": "string"
    * }
    * **Request Example**:
    * POST /admin/volunteers
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "name": "Volunteer One",
    * "email": "volunteer@example.com",
    * "password": "Volunteer123!",
    * "role": "product\_manager"
    * }
    * **Response Schema** (201 Created):
    * {
    * "volunteer\_id": "string",
    * "message": "Volunteer added successfully"
    * }
    * **Error Response** (400 Bad Request):
    * {
    * "error": "Invalid volunteer data",
    * "status": 400
    * }
30. **PUT /admin/volunteers/:id**
    * **Functionality**: Update a volunteer’s role or status.
    * **Authentication**: JWT required (super admin role).
    * **Path Parameters**:
      + id: Volunteer ID (string).
    * **Request Schema**:
    * {
    * "name": "string",
    * "email": "string",
    * "role": "string"
    * }
    * **Request Example**:
    * PUT /admin/volunteers/volunteer1
    * Authorization: Bearer <jwt\_token>
    * Content-Type: application/json
    * {
    * "name": "Updated Volunteer",
    * "email": "volunteer@example.com",
    * "role": "order\_processor"
    * }
    * **Response Schema** (200 OK):
    * {
    * "message": "Volunteer updated successfully"
    * }
    * **Error Response** (404 Not Found):
    * {
    * "error": "Volunteer not found",
    * "status": 404
    * }
31. **GET /admin/logs**
    * **Functionality**: Retrieve activity logs for admin actions.
    * **Authentication**: JWT required (super admin role).
    * **Query Parameters**:
      + action: String (e.g., "product\_add") - Filter by action type.
      + date\_from: String - Start date (ISO format).
      + date\_to: String - End date (ISO format).
      + page: Number - Pagination page (default: 1).
      + limit: Number - Items per page (default: 20).
    * **Request Example**:
    * GET /admin/logs?action=product\_add&page=1&limit=10
    * Authorization: Bearer <jwt\_token>
    * **Response Schema** (200 OK):
    * {
    * "logs": [
    * {
    * "id": "string",
    * "admin\_id": "string",
    * "action": "string",
    * "details": "string",
    * "created\_at": "string"
    * }
    * ],
    * "total": "number",
    * "page": "number",
    * "limit": "number"
    * }
    * **Error Response** (401 Unauthorized):
    * {
    * "error": "Invalid or expired token",
    * "status": 401
    * }