

tanojrahul:

> Generate complete API documentation for the **FoodShare** full-stack web application.

>

> Include all RESTful API endpoints used in the project. For each API endpoint, provide the following details:

> - **Endpoint URL**

> - **HTTP Method** ( `GET` , `POST` , `PUT` , `DELETE` , etc.)

> - **Request Headers** (if required)

> - **Request Body Schema** (with example JSON)

> - **Response Body Schema** (with example JSON)

> - **Authentication Requirements** (e.g., JWT required, public, admin only)

> - **Status Codes and Messages**

> - **Brief Description** of the functionality

>

> Categorize the APIs based on their functionality:

> 1. **Authentication APIs** (login, signup, logout)

> 2. **Donation APIs** (create, browse, request, update status, delete)

> 3. **User APIs** (profile, role management, rewards)

> 4. **Admin APIs** (approval, analytics, user management)

> 5. **Notification APIs** (alerts, expiration, push/email)

> 6. **Utility APIs** (file upload, maps integration, etc.)

>

> Follow OpenAPI/Swagger-like structure, but make it human-readable and implementation-ready.

first give me for

> 1. **Authentication APIs** (login, signup, logout)

and when i say next then give me for the next one

GitHub Copilot: I'll help you generate comprehensive API documentation for the authentication APIs in the FoodShare application. Let me first explore the codebase to understand the authentication implementation. Let me analyze the authentication-related code in the FoodShare application to document the Authentication APIs properly.

Based on my analysis of the codebase, I can see that the application has a well-structured authentication service that handles login, registration, and logout functionality. The authentication APIs are defined primarily in authService.js.

## # FoodShare API Documentation - Authentication APIs

### ## 1. Authentication APIs

#### ### 1.1 User Login

**\*\*Endpoint URL\*\*:** `/api/auth/login`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

``

Content-Type: application/json

``

**\*\*Request Body Schema\*\*:**

```json

{

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

"password": "securePassword123"

}

```

**\*\*Response Body Schema\*\*:**

```
`` `json
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "user": {
    "id": "u12345",
    "name": "John Doe",
    "email": "user@example.com",
    "role": "donor",
    "status": "Active",
    "createdAt": "2023-11-10T15:30:00Z"
  }
}
`` `
```

**\*\*Authentication Requirements\*\*:** None (Public API)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful login
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid credentials
- `403 Forbidden` : Account disabled or pending verification
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Authenticates a user with email and password credentials. Returns a JWT token and basic user information upon successful authentication.

### ### 1.2 User Registration

**\*\*Endpoint URL\*\*:** `/api/auth/register`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```json

```
{
  "name": "Jane Smith",
  "email": "jane.smith@example.com",
  "password": "securePassword123",
  "role": "beneficiary",
  "address": {
    "street": "123 Main St",
    "city": "Anytown",
    "state": "CA",
    "zipCode": "12345",
    "country": "USA"
  },
  "phone": "+1234567890"
}
```

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "user": {
    "id": "u67890",
    "name": "Jane Smith",
```

```
"email": "jane.smith@example.com",  
"role": "beneficiary",  
"status": "Active",  
"createdAt": "2023-11-15T10:20:30Z"  
}  
}  
` ``
```

**\*\*Authentication Requirements\*\*:** None (Public API)

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Successfully registered
- `400 Bad Request` : Invalid input data or validation errors
- `409 Conflict` : Email already registered
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Registers a new user with the provided information. Returns a JWT token and user data upon successful registration. For NGO accounts, the status will be set to "Pending Verification" until approved by an admin.

### ### 1.3 User Logout

**\*\*Endpoint URL\*\*:** `/api/auth/logout`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` ``

Authorization: Bearer {token}

` ``

**\*\*Request Body Schema\*\*:** Empty

**\*\*Response Body Schema\*\*:**

```
```json
{
  "message": "Logout successful"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successfully logged out
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Invalidates the user's session token on the server side. The client should also remove the token from local storage.

### ### 1.4 Token Refresh

**\*\*Endpoint URL\*\*:** `/api/auth/refresh-token`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```
```json
```

```
{
  "token": "current_jwt_token"
}
...
```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expiresAt": "2023-12-15T10:20:30Z"
}
...
```

**\*\*Authentication Requirements\*\*:** Valid but potentially expired JWT Token

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successfully refreshed token
- `401 Unauthorized` : Invalid token
- `403 Forbidden` : Refresh not allowed (token revoked)
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Refreshes an expired or soon-to-expire JWT token without requiring the user to log in again. The client should replace the old token with the new one for subsequent API calls.

### ### 1.5 Update User Profile

**\*\*Endpoint URL\*\*:** `/api/auth/profile`

**\*\*HTTP Method\*\*:** `PUT`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```json

```
{
  "name": "Jane Updated Smith",
  "address": {
    "street": "456 Oak Avenue",
    "city": "New City",
    "state": "NY",
    "zipCode": "67890",
    "country": "USA"
  },
  "phone": "+10987654321"
}
```

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "user": {
    "id": "u67890",
    "name": "Jane Updated Smith",
    "email": "jane.smith@example.com",
    "role": "beneficiary",
    "status": "Active",
    "address": {
      "street": "456 Oak Avenue",
```



```
"city": "New City",
"state": "NY",
"zipCode": "67890",
"country": "USA"
},
"phone": "+10987654321",
"updatedAt": "2023-11-18T14:25:30Z"
}
}
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successfully updated profile
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Updates the authenticated user's profile information. Only certain fields can be updated (cannot change email or role through this endpoint).

Let me continue with documenting the Donation APIs for the FoodShare application. I'll first explore the codebase to understand the donation-related functionality. Based on my analysis of the codebase, here's the documentation for the Donation APIs in the FoodShare application:

# FoodShare API Documentation - Donation APIs

## 2. Donation APIs

### 2.1 Create Donation

**\*\*Endpoint URL\*\*:** `/api/donations/create`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` `` `

Content-Type: application/json

Authorization: Bearer {token}

` `` `

**\*\*Request Body Schema\*\*:**

` `` `json

{

"title": "Fresh Vegetables Bundle",

"foodType": "Vegetables",

"quantity": 5,

"unit": "kg",

"category": "Produce",

"expiresAt": "2025-04-18T15:30:00Z",

"location": "123 Main St, Anytown",

"description": "Assorted fresh vegetables including carrots, broccoli, and bell peppers"

}

` `` `

**\*\*Response Body Schema\*\*:**

` `` `json

{

"id": "don123",

"title": "Fresh Vegetables Bundle",

"foodType": "Vegetables",

"quantity": 5,

```
"unit": "kg",
"category": "Produce",
"expiresAt": "2025-04-18T15:30:00Z",
"location": "123 Main St, Anytown",
"description": "Assorted fresh vegetables including carrots, broccoli, and bell peppers",
"donorId": "u12345",
"donorName": "Green Grocers",
"status": "Pending Approval",
"createdAt": "2025-04-14T10:20:30Z"
}
` ` `
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Donor role)

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Successfully created donation
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have appropriate role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Creates a new food donation listing with details provided by a donor. The donation initially has a status of "Pending Approval" until reviewed by an admin.

### ### 2.2 Get All Donations

**\*\*Endpoint URL\*\*:** `/api/donations/list`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `category` (optional): Filter by food category
- `status` (optional): Filter by donation status
- `fromDate` (optional): Filter donations created after date
- `toDate` (optional): Filter donations created before date
- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\*:**

```json

```
{
  "donations": [
    {
      "id": "don123",
      "title": "Fresh Vegetables Bundle",
      "foodType": "Vegetables",
      "quantity": 5,
      "unit": "kg",
      "category": "Produce",
      "expiresAt": "2025-04-18T15:30:00Z",
      "location": "123 Main St, Anytown",
      "description": "Assorted fresh vegetables including carrots, broccoli, and bell peppers",
      "donorId": "u12345",
      "donorName": "Green Grocers",
      "status": "Pending Approval",
      "createdAt": "2025-04-14T10:20:30Z"
    }
  ]
}
```

```
],  
  "total": 1,  
  "page": 1,  
  "totalPages": 1  
}  
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves a list of all donation listings with optional filtering and pagination.

### ### 2.3 Get Donation Details

**\*\*Endpoint URL\*\*:** `/api/donations/{id}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ``

Authorization: Bearer {token}

` ``

**\*\*Response Body Schema\*\*:**

` `` `json

{

"id": "don123",

```
"title": "Fresh Vegetables Bundle",
"foodType": "Vegetables",
"quantity": 5,
"unit": "kg",
"category": "Produce",
"expiresAt": "2025-04-18T15:30:00Z",
"location": "123 Main St, Anytown",
"description": "Assorted fresh vegetables including carrots, broccoli, and bell peppers",
"donor": {
  "id": "u12345",
  "name": "Green Grocers",
  "phone": "(555) 123-4567",
  "address": "123 Main St, Anytown"
},
"recipient": {
  "id": "u67890",
  "name": "Community Food Bank",
  "phone": "(555) 987-6543",
  "address": "456 Oak Ave, Townsville"
},
"ngo": {
  "id": "ngo789",
  "name": "Food Rescue Initiative",
  "phone": "(555) 456-7890"
},
"status": "In Transit",
"createdAt": "2025-04-14T10:20:30Z",
"timestamps": {
  "created": "2025-04-14T10:20:30Z",
  "accepted": "2025-04-14T12:30:00Z",
  "readyForPickup": "2025-04-14T15:00:00Z",
```

```
"inTransit": "2025-04-14T16:30:00Z"
},
"logistics": {
  "pickupAddress": "123 Main St, Cityville",
  "dropoffAddress": "456 Oak Ave, Townsville",
  "deliveryType": "NGO Logistics",
  "eta": "2025-04-14T18:00:00Z"
}
}
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have access to this donation
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves detailed information about a specific donation, including donor, recipient, and logistics information.

### ### 2.4 Update Donation Status

**\*\*Endpoint URL\*\*:** `/api/donations/{id}/status`

**\*\*HTTP Method\*\*:** `PUT`

**\*\*Request Headers\*\*:**

...

Content-Type: application/json

Authorization: Bearer {token}

...

**\*\*Request Body Schema\*\*:**

```json

```
{  
  "status": "Ready for Pickup"  
}
```

```

**\*\*Response Body Schema\*\*:**

```json

```
{  
  "id": "don123",  
  "status": "Ready for Pickup",  
  "updatedAt": "2025-04-14T14:30:00Z",  
  "timestamps": {  
    "readyForPickup": "2025-04-14T14:30:00Z"  
  },  
  "message": "Donation status updated successfully"  
}
```

...

**\*\*Authentication Requirements\*\*:** JWT Token Required (Donor, Beneficiary, NGO or Admin role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Status updated successfully
- `400 Bad Request` : Invalid status value
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to update this donation



- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Updates the status of a donation. Different roles can set different statuses based on the workflow: Donors can mark as "Ready for Pickup", NGOs can mark as "In Transit", Beneficiaries can mark as "Delivered", and Admins can set any status.

### ### 2.5 Get Donor Donations

**\*\*Endpoint URL\*\***: `/api/donations/donor/{userId}`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\***:

`` `

Authorization: Bearer {token}

`` `

**\*\*Request Parameters\*\***:

- `status` (optional): Filter by donation status
- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\***:

`` `json

{

  "donations": [

    {

      "id": "don123",

      "title": "Fresh Vegetables Bundle",

      "foodType": "Vegetables",

      "quantity": 5,

```
"unit": "kg",
"category": "Produce",
"expiresAt": "2025-04-18T15:30:00Z",
"status": "Pending Approval",
"createdAt": "2025-04-14T10:20:30Z"
}
],
"total": 1,
"page": 1,
"totalPages": 1
}
` ` `
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Donor or Admin role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these donations
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all donations created by a specific donor with optional filtering and pagination.

### ### 2.6 Claim Donation

**\*\*Endpoint URL\*\*:** `/api/donations/claim`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

```
{
  "donationId": "don123",
  "beneficiaryId": "u67890",
  "claimNotes": "Will pick up on Thursday afternoon"
}
```

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "id": "don123",
  "status": "Accepted",
  "recipientId": "u67890",
  "recipientName": "Community Food Bank",
  "claimedAt": "2025-04-14T11:45:00Z",
  "message": "Donation claimed successfully"
}
```

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary or NGO role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Donation claimed successfully

- `400 Bad Request` : Invalid request or donation not available

- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have appropriate role
- `404 Not Found` : Donation not found
- `409 Conflict` : Donation already claimed
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Claims an available donation on behalf of a beneficiary or NGO, updating the status to "Accepted".

### ### 2.7 Get Claimed Donations

**\*\*Endpoint URL\*\*:** `/api/donations/claimed/{beneficiaryId}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

`` `

Authorization: Bearer {token}

`` `

**\*\*Request Parameters\*\*:**

- `status` (optional): Filter by donation status
- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\*:**

`` `json

{

"donations": [

{

"id": "don123",

```

    "title": "Fresh Vegetables Bundle",
    "foodType": "Vegetables",
    "quantity": 5,
    "unit": "kg",
    "category": "Produce",
    "expiresAt": "2025-04-18T15:30:00Z",
    "donorName": "Green Grocers",
    "status": "Accepted",
    "claimedAt": "2025-04-14T11:45:00Z"
  }
],
"total": 1,
"page": 1,
"totalPages": 1
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary, NGO or Admin role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these donations
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all donations claimed by a specific beneficiary or NGO with optional filtering and pagination.

### ### 2.8 Browse Available Donations

**\*\*Endpoint URL\*\*:** `/api/donations/available`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Request Parameters\*\*:**

- `category` (optional): Filter by food category
- `distance` (optional): Maximum distance in km
- `latitude` (optional): Latitude for location-based search
- `longitude` (optional): Longitude for location-based search
- `expiryRange` (optional): Filter by expiration timeframe
- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\*:**

` ` `json

{

  "donations": [

    {

      "id": "don123",

      "title": "Fresh Vegetables Bundle",

      "type": "Vegetables",

      "quantity": 5,

      "unit": "kg",

      "expiresAt": "2025-04-18T15:30:00Z",

      "status": "Available",

      "donorName": "Green Grocers",

      "distance": "1.2 km",

```
    "location": "123 Main St, Anytown"
  }
],
"total": 1,
"page": 1,
"totalPages": 1
}
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary or NGO role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves a list of available donations that can be claimed, with optional filtering by category, distance, and expiration timeframe.

### ### 2.9 Request Donation

**\*\*Endpoint URL\*\*:** `/api/donations/request`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` ``

Content-Type: application/json

Authorization: Bearer {token}

` ``

**\*\*Request Body Schema\*\*:**

```
```json
{
  "donationId": "don123",
  "beneficiaryId": "u67890",
  "requestNotes": "Need for community meal program"
}
```
```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "id": "req456",
  "donationId": "don123",
  "beneficiaryId": "u67890",
  "status": "Pending",
  "requestedAt": "2025-04-14T13:20:00Z",
  "message": "Donation request submitted successfully"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary or NGO role)

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Request submitted successfully
- `400 Bad Request` : Invalid request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have appropriate role
- `404 Not Found` : Donation not found
- `409 Conflict` : Donation already requested or unavailable
- `500 Internal Server Error` : Server error



**\*\*Description\*\*:** Submits a request for a specific donation, which must be approved by the donor before it's claimed.

### ### 2.10 Submit Donation Review

**\*\*Endpoint URL\*\*:** `/api/donations/{donationId}/review`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

{

"rating": 5,

"comment": "The food was fresh and the pickup process was smooth. Thank you!"

}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "rev789",

"donationId": "don123",

"reviewerId": "u67890",

"reviewerName": "Community Food Bank",

```
"reviewerRole": "Beneficiary",  
"rating": 5,  
"comment": "The food was fresh and the pickup process was smooth. Thank you!",  
"createdAt": "2025-04-14T18:45:00Z"  
}  
` `` `
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Donor, Beneficiary, or NGO role)

**\*\*Status Codes and Messages\*\*:**

- ` 201 Created ` : Review submitted successfully
- ` 400 Bad Request ` : Invalid review data
- ` 401 Unauthorized ` : Invalid or missing token
- ` 403 Forbidden ` : User not authorized to review this donation
- ` 404 Not Found ` : Donation not found
- ` 409 Conflict ` : User has already reviewed this donation
- ` 500 Internal Server Error ` : Server error

**\*\*Description\*\*:** Submits a review for a completed donation. The user must be a participant (donor, recipient, or NGO) in the donation process.

### ### 2.11 Delete Donation

**\*\*Endpoint URL\*\*:** `/api/donations/{id}`

**\*\*HTTP Method\*\*:** `DELETE`

**\*\*Request Headers\*\*:**

` `` `

Authorization: Bearer {token}

` `` `

**\*\*Response Body Schema\*\*:**

```
```\n{\n  "message": "Donation deleted successfully",\n  "id": "don123"\n}\n```\n
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Donor or Admin role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Donation deleted successfully
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User not authorized to delete this donation
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Deletes a donation listing. Donors can only delete their own donations that are in "Pending" or "Available" status, while admins can delete any donation.

### ### 2.12 Search Donations

**\*\*Endpoint URL\*\*:** `/api/donations/search`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```\n

Authorization: Bearer {token}

```\n

**\*\*Request Parameters\*\*:**

- `query` (optional): Text search query
- `category` (optional): Filter by food category
- `status` (optional): Filter by donation status
- `fromDate` (optional): Filter by date range start
- `toDate` (optional): Filter by date range end
- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\*:**

```
` `` `json
{
  "donations": [
    {
      "id": "don123",
      "title": "Fresh Vegetables Bundle",
      "type": "Vegetables",
      "quantity": 5,
      "unit": "kg",
      "expiresAt": "2025-04-18T15:30:00Z",
      "status": "Available",
      "donorName": "Green Grocers",
      "createdAt": "2025-04-14T10:20:30Z"
    }
  ],
  "total": 1,
  "page": 1,
  "totalPages": 1
}
` `` `
```

**\*\*Authentication Requirements\*\***: JWT Token Required

**\*\*Status Codes and Messages\*\***:

- `200 OK` : Successful request
- `400 Bad Request` : Invalid search parameters
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Searches for donations based on various criteria, including text search and filters.

### ### 2.13 Get Donation Statistics

**\*\*Endpoint URL\*\***: `/api/donations/stats/{userId}`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\***:

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Response Body Schema\*\***:

` ` `json

```
{  
  "totalDonations": 15,  
  "activeDonations": 3,  
  "completedDonations": 12,  
  "impactMetrics": {  
    "totalQuantity": 87,
```

```
"peopleFed": 348,
"foodSaved": "87 kg",
"co2Saved": "174 kg"
},
"topCategories": [
  {"name": "Vegetables", "count": 7},
  {"name": "Bakery", "count": 5},
  {"name": "Canned Goods", "count": 3}
],
"monthlyStats": [
  {"month": "January", "count": 2},
  {"month": "February", "count": 3},
  {"month": "March", "count": 5},
  {"month": "April", "count": 5}
]
}
` ` `
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User not authorized to view these statistics
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves statistical information about a user's donation activity, including counts, impact metrics, and trends over time.

I'll continue with documenting the User APIs for the FoodShare application. Let me first analyze the codebase to understand how the user-related APIs are implemented.  
# FoodShare API Documentation - User APIs

## ## 3. User APIs

### ### 3.1 User Registration

**\*\*Endpoint URL\*\*:** `/api/auth/register`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```\n

Content-Type: application/json

```\n

**\*\*Request Body Schema\*\*:**

```json

```
{
  "name": "John Smith",
  "email": "john.smith@example.com",
  "password": "securePassword123",
  "role": "donor",
  "phone": "+1234567890",
  "address": "123 Main St, Anytown, CA 12345, USA"
}
```

```\n

**\*\*Response Body Schema\*\*:**

```json

{

```
"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
"user": {
  "id": "u12345",
  "name": "John Smith",
  "email": "john.smith@example.com",
  "role": "donor",
  "status": "Active",
  "createdAt": "2025-04-14T10:20:30Z"
}
}
```

**\*\*Authentication Requirements\*\*:** None (Public API)

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Successfully registered
- `400 Bad Request` : Invalid input data or validation errors
- `409 Conflict` : Email already registered
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Registers a new user with the provided information. For NGO accounts, the status will be set to "Pending Verification" until approved by an admin.

### ### 3.2 User Login

**\*\*Endpoint URL\*\*:** `/api/auth/login`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

...



Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```json

```
{  
  "email": "john.smith@example.com",  
  "password": "securePassword123"  
}
```

```

**\*\*Response Body Schema\*\*:**

```json

```
{  
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",  
  "user": {  
    "id": "u12345",  
    "name": "John Smith",  
    "email": "john.smith@example.com",  
    "role": "donor",  
    "status": "Active",  
    "createdAt": "2025-04-14T10:20:30Z"  
  }  
}
```

```

**\*\*Authentication Requirements\*\*:** None (Public API)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful login
- `400 Bad Request` : Invalid input data

- `401 Unauthorized` : Invalid credentials
- `403 Forbidden` : Account disabled or pending verification
- `500 Internal Server Error` : Server error

**Description**: Authenticates a user with email and password credentials. Returns a JWT token and basic user information upon successful authentication.

### ### 3.3 User Logout

**Endpoint URL**: `/api/auth/logout`

**HTTP Method**: `POST`

**Request Headers**:

```

Authorization: Bearer {token}

```

**Request Body Schema**: Empty

**Response Body Schema**:

```json

```
{  
  "message": "Logout successful"  
}
```

```

**Authentication Requirements**: JWT Token Required

**Status Codes and Messages**:

- `200 OK` : Successfully logged out

- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**Description**: Invalidates the user's session token on the server side. The client should also remove the token from local storage.

### ### 3.4 User Profile

**Endpoint URL**: `/api/users/{userId}`

**HTTP Method**: `GET`

**Request Headers**:

```

Authorization: Bearer {token}

```

**Response Body Schema**:

```json

{

"id": "u12345",

"name": "John Smith",

"email": "john.smith@example.com",

"role": "donor",

"phone": "+1234567890",

"address": "123 Main St, Anytown, CA 12345, USA",

"bio": "I'm passionate about reducing food waste and helping those in need.",

"verifiedUser": true,

"status": "Active",

"points": 2350,

"createdAt": "2025-03-14T10:20:30Z",

```
"stats": {  
  "donations": 27,  
  "received": 0,  
  "daysActive": 45  
}  
}  
` ``
```

**\*\*Authentication Requirements\*\***: JWT Token Required

**\*\*Status Codes and Messages\*\***:

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view this profile
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Retrieves detailed profile information for a specific user. Users can only view their own profile unless they are admins.

### ### 3.5 Update User Profile

**\*\*Endpoint URL\*\***: `/api/users/{userId}`

**\*\*HTTP Method\*\***: `PUT`

**\*\*Request Headers\*\***:

` ``

Authorization: Bearer {token}

Content-Type: application/json

` ``

**\*\*Request Body Schema\*\*:**

```
```json
{
  "name": "John A. Smith",
  "phone": "+1234567890",
  "address": "456 Oak Ave, Newtown, CA 12345, USA",
  "bio": "I'm passionate about reducing food waste and helping those in need through regular donations."
}
```
```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "id": "u12345",
  "name": "John A. Smith",
  "email": "john.smith@example.com",
  "phone": "+1234567890",
  "address": "456 Oak Ave, Newtown, CA 12345, USA",
  "bio": "I'm passionate about reducing food waste and helping those in need through regular donations.",
  "updatedAt": "2025-04-14T15:30:00Z"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successfully updated
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token

- `403 Forbidden` : User doesn't have permission to update this profile
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**Description**: Updates a user's profile information. Users can only update their own profile information, except for admins who can update any profile.

### ### 3.6 Get User Statistics

**Endpoint URL**: `/api/users/{userId}/stats`

**HTTP Method**: `GET`

**Request Headers**:

```

Authorization: Bearer {token}

```

**Response Body Schema**:

```json

{

"userId": "u12345",

"stats": {

"donations": 27,

"received": 0,

"foodSaved": "135 kg",

"impactPoints": 2350,

"daysActive": 45,

"lastActivity": "2025-04-12T10:15:30Z"

},

"recentActivity": [

```
{
  "type": "donation",
  "id": "don789",
  "title": "Fresh Vegetables Bundle",
  "date": "2025-04-10T14:30:00Z",
  "status": "Completed"
},
{
  "type": "reward",
  "id": "rwd456",
  "points": 150,
  "description": "Monthly donor achievement badge",
  "date": "2025-04-01T09:20:00Z"
}
]
}
```

...

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these statistics
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves statistics and activity information for a specific user. Users can only view their own statistics unless they are admins.

### ### 3.7 Get Reward History

**\*\*Endpoint URL\*\*:** `/api/rewards/{userId}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Request Parameters\*\*:**

- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\*:**

` ` `json

```
{
  "userId": "u12345",
  "totalPoints": 2350,
  "rewards": [
    {
      "id": "rwd456",
      "points": 150,
      "type": "achievement",
      "description": "Monthly donor achievement badge",
      "date": "2025-04-01T09:20:00Z"
    },
    {
      "id": "rwd455",
      "points": 200,
      "type": "donation",
```



```
    "description": "Completed donation #don789",
    "date": "2025-03-25T14:30:00Z"
  }
],
"pagination": {
  "total": 15,
  "page": 1,
  "limit": 10,
  "pages": 2
}
}
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these rewards
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves the reward point history for a specific user. Shows how points were earned and spent over time.

### ### 3.8 Add Reward Points (Admin)

**\*\*Endpoint URL\*\*:** `/api/rewards/{userId}/add`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```json

{

"points": 150,

"reason": "Community contribution recognition"

}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "rwd457",

"userId": "u12345",

"points": 150,

"totalPoints": 2500,

"description": "Community contribution recognition",

"date": "2025-04-14T15:30:00Z",

"addedBy": "admin"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Points added successfully

- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows administrators to add reward points to a user's account for various reasons such as special achievements or community contributions.

### ### 3.9 List All Users (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/users`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `role` (optional): Filter by user role
- `status` (optional): Filter by user status
- `search` (optional): Search by name or email
- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\*:**

```json

{

  "users": [

```
{
  "id": "u12345",
  "name": "John Smith",
  "email": "john.smith@example.com",
  "role": "donor",
  "status": "Active",
  "joined": "2025-03-14T10:20:30Z",
  "lastActive": "2025-04-12T10:15:30Z"
},
{
  "id": "u12346",
  "name": "Food Rescue Inc",
  "email": "info@foodrescue.org",
  "role": "ngo",
  "status": "Pending Verification",
  "joined": "2025-04-10T09:15:00Z",
  "lastActive": "2025-04-10T09:20:00Z"
}
],
"pagination": {
  "total": 248,
  "page": 1,
  "limit": 20,
  "pages": 13
}
}
...
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Retrieves a list of all users in the system with pagination and filtering options. Only accessible by admins.

### ### 3.10 Search Users (Admin)

**\*\*Endpoint URL\*\***: `/api/admin/users/search`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\***:

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\***:

- `query` : Search query string
- `role` (optional): Filter by user role
- `status` (optional): Filter by user status
- `fromDate` (optional): Filter by join date range start
- `toDate` (optional): Filter by join date range end
- `page` (optional): Page number for pagination
- `limit` (optional): Number of results per page

**\*\*Response Body Schema\*\***:

```json

{

```
"users": [  
  {  
    "id": "u12345",  
    "name": "John Smith",  
    "email": "john.smith@example.com",  
    "role": "donor",  
    "status": "Active",  
    "joined": "2025-03-14T10:20:30Z",  
    "lastActive": "2025-04-12T10:15:30Z"  
  },  
  {  
    "id": "u67890",  
    "name": "Jane Doe",  
    "email": "jane.doe@example.com",  
    "role": "donor",  
    "status": "Active",  
    "joined": "2025-03-15T10:20:30Z",  
    "lastActive": "2025-04-12T10:15:30Z"  
  },  
  {  
    "id": "u11111",  
    "name": "John Smith",  
    "email": "john.smith@example.com",  
    "role": "donor",  
    "status": "Active",  
    "joined": "2025-03-14T10:20:30Z",  
    "lastActive": "2025-04-12T10:15:30Z"  
  }  
],  
"pagination": {  
  "total": 3,  
  "page": 1,  
  "limit": 20,  
  "pages": 1  
}  
}
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid search parameters
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows admins to search for users based on various criteria and filters.

### ### 3.11 Change User Role (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/users/{userId}/role`

**\*\*HTTP Method\*\*:** `PATCH`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```json

{

"role": "beneficiary"

}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "u12345",

"name": "John Smith",

"email": "john.smith@example.com",

"role": "beneficiary",

"status": "Active",

"updatedAt": "2025-04-14T15:30:00Z",

"message": "User role updated successfully"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Role changed successfully
- `400 Bad Request` : Invalid role value
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows administrators to change a user's role between donor, beneficiary, NGO, and admin.

### ### 3.12 Update User Status (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/users/{userId}/status`

**\*\*HTTP Method\*\*:** `PATCH`

**\*\*Request Headers\*\*:**

...

Authorization: Bearer {token}

Content-Type: application/json

...

**\*\*Request Body Schema\*\*:**

```json

{

"status": "Disabled"

}



```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "id": "u12345",
  "name": "John Smith",
  "email": "john.smith@example.com",
  "role": "donor",
  "status": "Disabled",
  "updatedAt": "2025-04-14T15:30:00Z",
  "message": "User status updated successfully"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Status updated successfully
- `400 Bad Request` : Invalid status value
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows administrators to change a user's status between Active, Pending, and Disabled.

### ### 3.13 Verify NGO (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/users/{userId}/verify`

**\*\*HTTP Method\*\*:** `PATCH`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```json

{

"isVerified": true

}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "u12346",

"name": "Food Rescue Inc",

"email": "info@foodrescue.org",

"role": "ngo",

"status": "Active",

"verifiedUser": true,

"updatedAt": "2025-04-14T15:30:00Z",

"message": "NGO verification status updated successfully"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Verification status updated successfully
- `400 Bad Request` : Invalid request data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows administrators to verify NGO accounts after reviewing their documentation. This changes the NGO status from "Pending Verification" to "Active" and marks them as verified.

**### 3.14 Deactivate User (Admin)**

**\*\*Endpoint URL\*\*:** `/api/admin/users/{userId}`

**\*\*HTTP Method\*\*:** `DELETE`

**\*\*Request Headers\*\*:**

...

Authorization: Bearer {token}

...

**\*\*Response Body Schema\*\*:**

```json

```
{  
  "id": "u12345",  
  "message": "User account successfully deactivated",  
  "deactivatedAt": "2025-04-14T15:30:00Z"  
}
```

...

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Account deactivated successfully
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows administrators to deactivate a user account. This doesn't permanently delete the account but changes its status to "Disabled" and prevents the user from logging in.

### ### 3.15 Refresh Token

**\*\*Endpoint URL\*\*:** `/api/auth/refresh-token`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

```

**\*\*Request Body Schema\*\*:**

```json

```
{  
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."  
}
```

```

**\*\*Response Body Schema\*\*:**

```
```\n{\n  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",\n  "expiresAt": "2025-04-15T15:30:00Z"\n}\n```\n
```

**\*\*Authentication Requirements\*\*:** Valid but potentially expired JWT Token

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successfully refreshed token
- `401 Unauthorized` : Invalid token
- `403 Forbidden` : Refresh not allowed (token revoked)
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Refreshes an expired or soon-to-expire JWT token without requiring the user to log in again. The client should replace the old token with the new one for subsequent API calls.

Let me continue with documenting the Notification APIs for the FoodShare application. I'll analyze the codebase first to understand how the notification system works.# FoodShare API Documentation - Notification APIs

After analyzing the notification service implementation in the codebase, I'll now document the Notification APIs available in the FoodShare application.

## ## 4. Notification APIs

### ### 4.1 Send In-App Notification

**\*\*Endpoint URL\*\*:** `/api/notifications`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

{

"userId": "u12345",

"title": "Donation Matched!",

"message": "Your donation has been matched with Food Rescue Initiative.",

"type": "success",

"linkTo": "/donations/don123",

"metadata": {

"donationId": "don123",

"recipientId": "u67890"

```
}  
}  
` ``
```

**\*\*Response Body Schema\*\*:**

```
` `` json  
{  
  "id": "not789",  
  "userId": "u12345",  
  "title": "Donation Matched!",  
  "message": "Your donation has been matched with Food Rescue Initiative.",  
  "type": "success",  
  "isRead": false,  
  "createdAt": "2025-04-14T15:30:00Z",  
  "linkTo": "/donations/don123",  
  "metadata": {  
    "donationId": "don123",  
    "recipientId": "u67890"  
  }  
}  
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Notification created successfully
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : Insufficient permissions
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Creates a new in-app notification for a specific user. The notification can include a link and additional metadata for context.

### ### 4.2 Get User Notifications

**\*\*Endpoint URL\*\*:** `/api/notifications/user/{userId}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `limit` (optional): Maximum number of notifications to return (default: 20)
- `offset` (optional): Number of notifications to skip (default: 0)
- `unreadOnly` (optional): Only return unread notifications if true

**\*\*Response Body Schema\*\*:**

```json

[

{

"id": "not789",

"userId": "u12345",

"title": "Donation Matched!",

"message": "Your donation has been matched with Food Rescue Initiative.",

"type": "success",

"isRead": false,

"createdAt": "2025-04-14T15:30:00Z",

"linkTo": "/donations/don123",



```
"metadata": {
  "donationId": "don123"
},
{
  "id": "not788",
  "userId": "u12345",
  "title": "Donation Expiring Soon!",
  "message": "Your donation 'Fresh Vegetables Bundle' will expire in less than an hour.",
  "type": "warning",
  "isRead": true,
  "createdAt": "2025-04-14T14:00:00Z",
  "linkTo": "/donations/don122",
  "metadata": {
    "donationId": "don122"
  }
}
]
```

...

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these notifications
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves a list of notifications for a specific user with optional filtering for unread notifications and pagination support.

### ### 4.3 Mark Notification as Read

**\*\*Endpoint URL\*\*:** `/api/notifications/{notificationId}/read`

**\*\*HTTP Method\*\*:** `PATCH`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "not789",

"isRead": true,

"updatedAt": "2025-04-14T15:35:00Z"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Notification marked as read successfully
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to update this notification
- `404 Not Found` : Notification not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Marks a specific notification as read. This endpoint should be called when a user clicks on or views a notification.

#### ### 4.4 Mark All Notifications as Read

**\*\*Endpoint URL\*\*:** `/api/notifications/user/{userId}/read-all`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

...

Authorization: Bearer {token}

...

**\*\*Response Body Schema\*\*:**

```json

{

"success": true,

"count": 5,

"message": "All notifications marked as read"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Notifications marked as read successfully
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to update these notifications
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Marks all unread notifications for a specific user as read. This endpoint is typically called when a user clicks "Mark all as read" in the notification panel.

### ### 4.5 Get Notification Statistics

**\*\*Endpoint URL\*\*:** `/api/notifications/user/{userId}/stats`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "total": 15,
  "unread": 3,
  "categories": {
    "success": 5,
    "warning": 3,
    "info": 7,
    "error": 0
  },
  "recentActivity": "2025-04-14T15:30:00Z"
}
```

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these statistics
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves notification statistics for a specific user, including the total count, unread count, and breakdown by notification type.

### ### 4.6 Subscribe to Real-Time Notifications

**\*\*Endpoint URL\*\*:** `/api/notifications/subscribe`

**\*\*HTTP Method\*\*:** `GET` (WebSocket connection)

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Response\*\*:** WebSocket connection established

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `101 Switching Protocols` : WebSocket connection established successfully
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to subscribe
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Establishes a WebSocket connection to receive real-time notifications. New notifications will be pushed to the client as they are created, without requiring polling.

### ### 4.7 Send Email Notification

**\*\*Endpoint URL\*\*:** `/api/notifications/email`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

{

"toEmail": "recipient@example.com",

"subject": "Your FoodShare donation has been matched",

"htmlBody": "<div><h1>Donation Matched!</h1><p>Your donation has been matched with Food Rescue Initiative.</p></div>",

"templateParams": {

"donorName": "John Smith",

"recipientName": "Food Rescue Initiative"

}

}

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "success": true,
  "messageId": "email-1713171000000",
  "status": 200
}
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin or System role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Email sent successfully
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to send emails
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Sends an email notification to a specified recipient. This endpoint is typically used by the system for sending transactional emails like donation matches, expiration alerts, etc.

### ### 4.8 Send System Notification

**\*\*Endpoint URL\*\*:** `/api/notifications/system`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` ``

Content-Type: application/json

Authorization: Bearer {token}

` ``

**\*\*Request Body Schema\*\*:**

```
```json
{
  "title": "Scheduled Maintenance",
  "message": "FoodShare will be undergoing scheduled maintenance on April 20th from 2-4 AM EST.",
  "type": "info",
  "roles": ["admin", "donor", "ngo", "beneficiary"]
}
```
```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "success": true,
  "notificationId": "sys123",
  "recipientCount": 248,
  "createdAt": "2025-04-14T15:30:00Z"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : System notification sent successfully
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Creates and sends a system-wide notification to all users or to users with specific roles. This is used for announcements, alerts, and other system-wide communications.



### ### 4.9 Get System Notifications

**\*\*Endpoint URL\*\*:** `/api/notifications/system`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

`` `

Authorization: Bearer {token}

`` `

**\*\*Request Parameters\*\*:**

- `limit` (optional): Maximum number of notifications to return (default: 20)
- `offset` (optional): Number of notifications to skip (default: 0)
- `type` (optional): Filter by notification type (info, warning, error)

**\*\*Response Body Schema\*\*:**

`` `json

[

{

"id": "sys123",

"title": "Scheduled Maintenance",

"message": "FoodShare will be undergoing scheduled maintenance on April 20th from 2-4 AM EST.",

"type": "info",

"createdAt": "2025-04-14T15:30:00Z",

"targetRoles": ["admin", "donor", "ngo", "beneficiary"],

"createdBy": "admin",

"priority": "medium"

},

```
{
  "id": "sys122",
  "title": "Multiple Food Donations Approaching Expiration",
  "message": "5 donations are expiring in the next 3 hours.",
  "type": "warning",
  "createdAt": "2025-04-14T14:00:00Z",
  "targetRoles": ["admin"],
  "priority": "high"
}
]
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves a list of system notifications for admin users to monitor system alerts and announcements.

### ### 4.10 Notify Donation Match

**\*\*Endpoint URL\*\*:** `/api/notifications/donation-match`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` ``

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

```
{
  "donorId": "u12345",
  "recipientId": "u67890",
  "donationDetails": {
    "id": "don123",
    "title": "Fresh Vegetables Bundle",
    "quantity": "10 kg",
    "expiresAt": "2025-04-18T15:30:00Z",
    "location": "123 Main St, Anytown"
  }
}
```

}

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "success": true,
  "notifications": [
    {
      "type": "in-app",
      "userId": "u12345",
      "notificationId": "not789",
      "success": true
    },
    {
```

{

```

    "type": "in-app",
    "userId": "u67890",
    "notificationId": "not790",
    "success": true
  },
  {
    "type": "email",
    "recipient": "donor@example.com",
    "success": true,
    "messageId": "email-1713171000000"
  }
]
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin or System role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Notifications sent successfully
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission
- `404 Not Found` : Donor or recipient not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Sends notifications to both donor and recipient when a donation is matched. This includes both in-app notifications and email notifications if email addresses are available.

### ### 4.11 Notify Donation Expiration

**\*\*Endpoint URL\*\*:** `/api/notifications/donation-expiration`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` ` `

Content-Type: application/json

Authorization: Bearer {token}

` ` `

**\*\*Request Body Schema\*\*:**

` ` `json

{

  "donationId": "don123",

  "expirationDate": "2025-04-18T15:30:00Z"

}

` ` `

**\*\*Response Body Schema\*\*:**

` ` `json

{

  "success": true,

  "notified": true,

  "hoursLeft": 2,

  "notifications": [

    {

      "type": "in-app",

      "userId": "u12345",

      "notificationId": "not791",

      "success": true

    },

    {

```

    "type": "in-app",
    "userId": "admin",
    "notificationId": "not792",
    "success": true
  },
  {
    "type": "email",
    "recipient": "donor@example.com",
    "success": true,
    "messageId": "email-1713171000001"
  }
]
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin or System role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Process completed successfully (may not have sent notifications if not within threshold)
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Checks if a donation is nearing expiration and sends notifications to relevant parties if it is within the notification threshold (e.g., 3 hours before expiration). Notifications are sent to the donor and to admins.

This completes the documentation for the Notification APIs in the FoodShare application, which allow users to receive real-time updates about their donations, matches, and other important events.



Let me continue with documenting the remaining APIs for the FoodShare application. I'll focus on documenting the Map Service APIs next. # FoodShare API Documentation - Location/Map APIs

Based on my analysis of the mapService.js file in the codebase, I'll now document the Location/Map APIs available in the FoodShare application.

## ## 5. Location/Map APIs

### ### 5.1 Geocode Address

**\*\*Endpoint URL\*\*:** `/api/map/geocode`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `address` (required): The address to convert to coordinates

**\*\*Response Body Schema\*\*:**

```json

{

"lat": 37.7749,

"lng": -122.4194,

"fullAddress": "123 Main St, San Francisco, CA 94103, USA"

}

```



**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid or missing address
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Address could not be geocoded
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Converts a human-readable address into geographic coordinates (latitude and longitude). Also returns a standardized full address.

### ### 5.2 Reverse Geocode

**\*\*Endpoint URL\*\*:** `/api/map/reverse-geocode`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Request Parameters\*\*:**

- `lat` (required): Latitude coordinate
- `lng` (required): Longitude coordinate

**\*\*Response Body Schema\*\*:**

` ` `json

{

"address": "123 Main St, San Francisco, CA 94103, USA"

```
}  
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid or missing coordinates
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Location could not be reverse geocoded
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Converts geographic coordinates (latitude and longitude) to a human-readable address.

### ### 5.3 Calculate Distance

**\*\*Endpoint URL\*\*:** `/api/map/distance`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```
` ``
```

Authorization: Bearer {token}

```
` ``
```

**\*\*Request Parameters\*\*:**

- `startLat` (required): Starting point latitude
- `startLng` (required): Starting point longitude
- `endLat` (required): Ending point latitude
- `endLng` (required): Ending point longitude

- `unit` (optional): Unit of measurement ('km' or 'mi', defaults to 'km')

**\*\*Response Body Schema\*\*:**

```json

{

"distance": 12.43,

"unit": "km"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request

- `400 Bad Request` : Invalid or missing coordinates

- `401 Unauthorized` : Invalid or missing token

- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Calculates the direct distance between two geographic coordinates using the Haversine formula, which accounts for the Earth's curvature.

### ### 5.4 Get Route Map

**\*\*Endpoint URL\*\*:** `/api/map/route`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `startLat` (required): Starting point latitude
- `startLng` (required): Starting point longitude
- `endLat` (required): Ending point latitude
- `endLng` (required): Ending point longitude
- `mode` (optional): Transportation mode ('driving', 'walking', 'bicycling', 'transit', defaults to 'driving')

**\*\*Response Body Schema\*\*:**

```
`` `json
{
  "distance": 13.8,
  "duration": 24,
  "polyline": "enc:gfo~@bk_eGxBkANe@Ba@Ai@le@O_@MSKKKGMe@Kw@K_A...",
  "coordinates": [
    { "lat": 37.7749, "lng": -122.4194 },
    { "lat": 37.7710, "lng": -122.4187 },
    /* Additional coordinates */
  ],
  "steps": [
    {
      "instructions": "Head south on Market St",
      "distance": 0.2,
      "duration": 1
    }
    /* Additional steps */
  ]
}
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid or missing coordinates
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Route not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Calculates a route between two geographic coordinates and returns detailed information including distance, duration, turn-by-turn instructions, and path coordinates for displaying on a map.

### ### 5.5 Get Estimated Time of Arrival

**\*\*Endpoint URL\*\*:** `/api/map/eta`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

...

Authorization: Bearer {token}

...

**\*\*Request Parameters\*\*:**

- `startLat` (required): Starting point latitude
- `startLng` (required): Starting point longitude
- `endLat` (required): Ending point latitude
- `endLng` (required): Ending point longitude
- `departureTime` (optional): Departure time in ISO format (defaults to current time)

**\*\*Response Body Schema\*\*:**

```
` ``json
{
  "duration": 24,
  "distance": 13.8,
  "estimatedArrival": "2025-04-14T16:00:00Z",
  "trafficLevel": "moderate",
  "formattedETA": "24 minutes (13.8 km)"
}
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid or missing coordinates
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : ETA calculation failed
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Calculates the estimated time of arrival between two geographic coordinates, accounting for current or predicted traffic conditions.

### ### 5.6 Get Delivery Location

**\*\*Endpoint URL\*\*:** `/api/map/delivery-location/{deliveryId}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ``

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "lat": 37.7749,
  "lng": -122.4194,
  "address": "123 Delivery Street, San Francisco, CA",
  "timestamp": "2025-04-14T15:30:00Z",
  "eta": "10 minutes",
  "speed": "25 km/h",
  "deliveryStatus": "In Transit"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to track this delivery
- `404 Not Found` : Delivery not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves the current location and status of a delivery in progress. This can be used for real-time tracking of food donations being transported.

### ### 5.7 Find Nearby Donations

**\*\*Endpoint URL\*\*:** `/api/map/nearby-donations`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Request Parameters\*\*:**

- `lat` (required): User's current latitude
- `lng` (required): User's current longitude
- `radius` (optional): Search radius in kilometers (default: 10)
- `limit` (optional): Maximum number of results (default: 20)
- `status` (optional): Filter by donation status (default: "Available")
- `category` (optional): Filter by food category

**\*\*Response Body Schema\*\*:**

` ` ` json

```
{
  "donations": [
    {
      "id": "don123",
      "title": "Fresh Vegetables Bundle",
      "foodType": "Vegetables",
      "quantity": 5,
      "unit": "kg",
      "lat": 37.7710,
      "lng": -122.4187,
      "address": "456 Market St, San Francisco, CA 94103",
      "distance": 1.2,
      "expiresAt": "2025-04-18T15:30:00Z"
    },
  ],
}
```



```
{
  "id": "don124",
  "title": "Bakery Items",
  "foodType": "Baked Goods",
  "quantity": 10,
  "unit": "items",
  "lat": 37.7830,
  "lng": -122.4065,
  "address": "789 Mission St, San Francisco, CA 94103",
  "distance": 2.4,
  "expiresAt": "2025-04-15T12:00:00Z"
}
],
"total": 2,
"center": {
  "lat": 37.7749,
  "lng": -122.4194
},
"radius": 10
}
...

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary or NGO role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid or missing coordinates
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have appropriate role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Finds available food donations within a specified radius of the user's location. Results are sorted by proximity.

### ### 5.8 Generate Static Map

**\*\*Endpoint URL\*\*:** `/api/map/static`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

...

Authorization: Bearer {token}

...

**\*\*Request Parameters\*\*:**

- `lat` (required): Center latitude of the map
- `lng` (required): Center longitude of the map
- `zoom` (optional): Zoom level (1-20, default: 14)
- `width` (optional): Image width in pixels (default: 600)
- `height` (optional): Image height in pixels (default: 400)
- `markers` (optional): Array of marker positions in format "lat,lng" (can specify multiple)

**\*\*Response\*\*:** PNG/JPEG image file

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid parameters
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Generates a static map image centered at the specified coordinates with optional markers. Useful for embedding maps in emails, reports, or other non-interactive contexts.

This completes the documentation for the Location/Map APIs in the FoodShare application, providing various features for geocoding, routing, distance calculations, and location-based search functionality.

I'll continue with documenting the Review and Stats APIs for the FoodShare application. Let me first analyze the codebase to understand these features. # FoodShare API Documentation - Review APIs

Based on my analysis of the codebase, I'll now document the Review APIs available in the FoodShare application.

## ## 6. Review APIs

The FoodShare application includes a comprehensive review system that allows users to provide feedback on donations, rate their experiences, and help build trust within the platform's community. Here are the API endpoints for managing reviews:

### ### 6.1 Submit Review

**\*\*Endpoint URL\*\*:** ``/api/donations/{donationId}/review``

**\*\*HTTP Method\*\*:** ``POST``

**\*\*Request Headers\*\*:**

`` ` ``

Content-Type: application/json

Authorization: Bearer {token}

`` ` ``

**\*\*Request Body Schema\*\*:**

`` ` `json`

`{`

`"rating": 5,`

`"comment": "The vegetables were very fresh and the pickup process was smooth. Thank you for your generosity!",`

`"reviewerRole": "Beneficiary"`

`}`

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "id": "rev101",
  "donationId": "don123",
  "reviewerId": "u12345",
  "reviewerName": "Jane Smith",
  "reviewerRole": "Beneficiary",
  "rating": 5,
  "comment": "The vegetables were very fresh and the pickup process was smooth. Thank you for your generosity!",
  "date": "2025-04-14T15:30:00Z",
  "status": "Active"
}
```

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Review submitted successfully
- `400 Bad Request` : Invalid input data or rating not provided
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User not authorized to review this donation
- `404 Not Found` : Donation not found
- `409 Conflict` : User has already reviewed this donation
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows a user to submit a review for a donation they were involved with. Donors can review beneficiaries and beneficiaries can review donors after a donation has been completed.

### ### 6.2 Get Reviews for Donation

**\*\*Endpoint URL\*\*:** `/api/donations/{donationId}/reviews`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

`` `

Authorization: Bearer {token}

`` `

**\*\*Request Parameters\*\*:**

- `page` (optional): Page number for pagination (default: 1)

- `limit` (optional): Number of reviews per page (default: 10)

**\*\*Response Body Schema\*\*:**

`` `json

{

  "reviews": [

    {

      "id": "rev101",

      "donationId": "don123",

      "reviewerId": "u12345",

      "reviewerName": "Jane Smith",

      "reviewerRole": "Beneficiary",

      "rating": 5,

      "comment": "The vegetables were very fresh and the pickup process was smooth. Thank you for your generosity!",

      "date": "2025-04-14T15:30:00Z",

      "status": "Active"

```
},
{
  "id": "rev102",
  "donationId": "don123",
  "reviewerId": "u67890",
  "reviewerName": "Green Grocers",
  "reviewerRole": "Donor",
  "rating": 4,
  "comment": "The food bank was very professional and picked up the items promptly.",
  "date": "2025-04-14T16:20:00Z",
  "status": "Active"
}
],
"pagination": {
  "total": 2,
  "page": 1,
  "limit": 10,
  "pages": 1
}
}
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all reviews associated with a specific donation. This includes reviews from both the donor and the recipient.

### ### 6.3 Get User Reviews

**\*\*Endpoint URL\*\*:** `/api/users/{userId}/reviews`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `type` (optional): Type of reviews to retrieve ("given", "received", or "all")
- `page` (optional): Page number for pagination (default: 1)
- `limit` (optional): Number of reviews per page (default: 10)

**\*\*Response Body Schema\*\*:**

```json

{

"reviews": [

{

"id": "rev101",

"donationId": "don123",

"donationTitle": "Fresh Vegetables Bundle",

"reviewerId": "u12345",

"reviewerName": "Jane Smith",

"reviewerRole": "Beneficiary",

"recipientId": "u67890",



```
"recipientName": "Green Grocers",
"recipientRole": "Donor",
"rating": 5,
"comment": "The vegetables were very fresh and the pickup process was smooth. Thank you
for your generosity!",
"date": "2025-04-14T15:30:00Z",
"status": "Active"
}
],
"pagination": {
"total": 1,
"page": 1,
"limit": 10,
"pages": 1
},
"stats": {
"averageRating": 4.8,
"totalReviews": 15,
"ratingDistribution": {
"5": 10,
"4": 4,
"3": 1,
"2": 0,
"1": 0
}
}
}
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these reviews
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all reviews associated with a specific user, either reviews they've given, reviews they've received, or both. Also includes statistics about the user's review history.

### ### 6.4 Update Review

**\*\*Endpoint URL\*\*:** `/api/reviews/{reviewId}`

**\*\*HTTP Method\*\*:** `PUT`

**\*\*Request Headers\*\*:**

...

Content-Type: application/json

Authorization: Bearer {token}

...

**\*\*Request Body Schema\*\*:**

```json

```
{  
  "rating": 4,  
  "comment": "Updated comment: The vegetables were fresh but delivery was delayed."  
}
```

...

**\*\*Response Body Schema\*\*:**

```
` ``json
{
  "id": "rev101",
  "donationId": "don123",
  "reviewerId": "u12345",
  "reviewerName": "Jane Smith",
  "reviewerRole": "Beneficiary",
  "rating": 4,
  "comment": "Updated comment: The vegetables were fresh but delivery was delayed.",
  "date": "2025-04-14T15:30:00Z",
  "updatedAt": "2025-04-14T17:45:00Z",
  "status": "Active"
}
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- ` 200 OK ` : Review updated successfully
- ` 400 Bad Request ` : Invalid input data
- ` 401 Unauthorized ` : Invalid or missing token
- ` 403 Forbidden ` : User doesn't have permission to update this review
- ` 404 Not Found ` : Review not found
- ` 500 Internal Server Error ` : Server error

**\*\*Description\*\*:** Allows a user to update a review they previously submitted. Users can only update their own reviews.

### ### 6.5 Delete Review

**\*\*Endpoint URL\*\*:** `/api/reviews/{reviewId}`

**\*\*HTTP Method\*\*:** `DELETE`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "rev101",

"message": "Review successfully deleted",

"deletedAt": "2025-04-14T17:50:00Z"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Review deleted successfully
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to delete this review
- `404 Not Found` : Review not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows a user to delete a review they previously submitted. Users can only delete their own reviews, while administrators can delete any review.

### 6.6 Report Review

**\*\*Endpoint URL\*\*:** `/api/reviews/{reviewId}/report`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

{

"reason": "Inappropriate content",

"details": "This review contains offensive language."

}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "rep123",

"reviewId": "rev101",

"reporterId": "u12345",

"reason": "Inappropriate content",

"details": "This review contains offensive language.",

"status": "Pending",

"submittedAt": "2025-04-14T17:55:00Z"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Report submitted successfully
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Review not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows a user to report a review that they believe violates the platform's guidelines. An administrator will review the report and take appropriate action.

### 6.7 Moderate Review (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/reviews/{reviewId}/moderate`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` ` `

Content-Type: application/json

Authorization: Bearer {token}

` ` `

**\*\*Request Body Schema\*\*:**

` `` `json

```
{  
  "action": "approve",  
  "notes": "Review meets community guidelines.",  
  "notifyUser": true  
}
```

```

**\*\*Response Body Schema\*\*:**

```json

```
{  
  "id": "rev101",  
  "status": "Approved",  
  "moderatorId": "admin123",  
  "moderationNotes": "Review meets community guidelines.",  
  "moderatedAt": "2025-04-14T18:00:00Z"  
}
```

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Review moderated successfully
- `400 Bad Request` : Invalid input data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `404 Not Found` : Review not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Allows administrators to moderate reported reviews. The possible actions are "approve", "flag", or "remove".

### ### 6.8 Get Review Reports (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/reviews/reports`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `status` (optional): Filter by report status ("pending", "resolved", "all")
- `page` (optional): Page number for pagination (default: 1)
- `limit` (optional): Number of reports per page (default: 20)

**\*\*Response Body Schema\*\*:**

```json

```
{
  "reports": [
    {
      "id": "rep123",
      "reviewId": "rev101",
      "reviewContent": {
        "rating": 2,
        "comment": "This service was terrible...",
        "reviewerName": "John Doe"
      },
      "reporterId": "u12345",
      "reporterName": "Jane Smith",
      "reason": "Inappropriate content",
      "details": "This review contains offensive language.",
      "status": "Pending",
      "submittedAt": "2025-04-14T17:55:00Z"
    }
  ],
}
```



```
"pagination": {  
  "total": 1,  
  "page": 1,  
  "limit": 20,  
  "pages": 1  
}  
}  
` ``
```

**\*\*Authentication Requirements\*\***: JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\***:

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Retrieves a list of reported reviews for administrator moderation. Includes details about both the report and the reported review.

## # FoodShare API Documentation - Statistics APIs

Now, let's document the Statistics APIs for the FoodShare application:

### ## 7. Statistics APIs

#### ### 7.1 Get User Statistics

**\*\*Endpoint URL\*\***: `/api/users/{userId}/stats`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\*:**

```json

{

"userId": "u12345",

"role": "Donor",

"donations": {

"total": 42,

"active": 3,

"completed": 37,

"cancelled": 2

},

"claimed": {

"total": 0,

"active": 0,

"completed": 0

},

"impact": {

"foodSaved": 320,

"mealsProvided": 960,

"co2Reduced": 640

},

"points": 2350,

"rating": {

"average": 4.8,

"count": 37

```
},
"memberSince": "2024-11-15T10:30:00Z",
"lastActive": "2025-04-14T15:30:00Z"
}
` ` `
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these statistics
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves comprehensive statistics about a user's activity on the platform, including donations, claims, impact metrics, points, and ratings.

### ### 7.2 Get Donation Statistics

**\*\*Endpoint URL\*\*:** `/api/donations/{donationId}/stats`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Response Body Schema\*\*:**

` `` `json

```

{
  "id": "don123",
  "title": "Fresh Vegetables Bundle",
  "createdAt": "2025-04-10T10:30:00Z",
  "completedAt": "2025-04-12T15:45:00Z",
  "timeToMatch": 5.25,
  "timeToComplete": 53.25,
  "viewCount": 42,
  "requestCount": 3,
  "environmentalImpact": {
    "co2Saved": 15,
    "waterSaved": 2500,
    "landUseSaved": 10
  },
  "socialImpact": {
    "mealsProvided": 30,
    "beneficiariesServed": 1
  },
  "reviews": {
    "donorRating": 5,
    "recipientRating": 4
  }
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Donation not found

- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Retri

Let me continue with documenting the Review APIs for the FoodShare application based on the code analysis.

## # FoodShare API Documentation - Review APIs

### ## 6. Review APIs

Based on my analysis of the codebase, especially the `ReviewCard` component, `DonationDetailsPage`, and `DonationTrackingPage`, I'll document the Review APIs available in the FoodShare application.

#### ### 6.1 Submit Review

**\*\*Endpoint URL\*\*:** `/api/donations/{donationId}/review`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

{

"rating": 5,

"comment": "The food quality was excellent and the pickup process was smooth.",

"reviewType": "donor-to-beneficiary"

}

```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "id": "rev123",
  "donationId": "don456",
  "reviewerId": "u789",
  "reviewerName": "John Smith",
  "reviewerRole": "Beneficiary",
  "recipientId": "u012",
  "recipientName": "Fresh Grocers",
  "rating": 5,
  "comment": "The food quality was excellent and the pickup process was smooth.",
  "reviewType": "donor-to-beneficiary",
  "status": "Active",
  "createdAt": "2025-04-14T15:30:00Z"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Review submitted successfully
- `400 Bad Request` : Invalid review data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to review this donation
- `404 Not Found` : Donation not found
- `409 Conflict` : User has already submitted a review for this donation
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Submits a review for a completed donation. Either the donor can review the beneficiary/NGO, or the beneficiary/NGO can review the donor. The user must have been a participant in the donation transaction to submit a review.

### ### 6.2 Get Donation Reviews

**\*\*Endpoint URL\*\*:** `/api/donations/{donationId}/reviews`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Response Body Schema\*\*:**

` ` `json

[

{

"id": "rev123",

"donationId": "don456",

"reviewerId": "u789",

"reviewerName": "John Smith",

"reviewerRole": "Beneficiary",

"rating": 5,

"comment": "The food quality was excellent and the pickup process was smooth.",

"reviewType": "beneficiary-to-donor",

"status": "Active",

"createdAt": "2025-04-14T15:30:00Z"

},

{

"id": "rev124",

"donationId": "don456",

"reviewerId": "u012",



```
"reviewerName": "Fresh Grocers",
"reviewerRole": "Donor",
"rating": 4,
"comment": "Pickup was on time, very professional interaction.",
"reviewType": "donor-to-beneficiary",
"status": "Active",
"createdAt": "2025-04-15T09:15:00Z"
}
]
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all reviews associated with a specific donation.

### ### 6.3 Get User Reviews

**\*\*Endpoint URL\*\*:** `/api/users/{userId}/reviews`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ``

Authorization: Bearer {token}

` ``

**\*\*Request Parameters\*\*:**

- `role` (optional): Filter by review role ('reviewer' or 'recipient')
- `page` (optional): Page number for pagination
- `limit` (optional): Number of reviews per page
- `status` (optional): Filter by review status

**\*\*Response Body Schema\*\*:**

```json

```
{
  "reviews": [
    {
      "id": "rev123",
      "donationId": "don456",
      "donationTitle": "Fresh Vegetables Bundle",
      "reviewerId": "u789",
      "reviewerName": "John Smith",
      "recipientId": "u012",
      "recipientName": "Fresh Grocers",
      "rating": 5,
      "comment": "The food quality was excellent and the pickup process was smooth.",
      "reviewType": "beneficiary-to-donor",
      "status": "Active",
      "createdAt": "2025-04-14T15:30:00Z"
    }
  ],
  "pagination": {
    "total": 12,
    "page": 1,
    "limit": 10,
    "pages": 2
  }
}
```

```
}  
}  
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these reviews
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all reviews either written by or about a specific user, depending on the `role` parameter.

### ### 6.4 Update Review

**\*\*Endpoint URL\*\*:** `/api/reviews/{reviewId}`

**\*\*HTTP Method\*\*:** `PUT`

**\*\*Request Headers\*\*:**

```
` ``
```

Content-Type: application/json

Authorization: Bearer {token}

```
` ``
```

**\*\*Request Body Schema\*\*:**

```
` `` json
```

```
{
```

```
"rating": 4,
"comment": "Updated comment after further consideration."
}
` ` `
```

**\*\*Response Body Schema\*\*:**

```
` ` `json
{
  "id": "rev123",
  "donationId": "don456",
  "reviewerId": "u789",
  "rating": 4,
  "comment": "Updated comment after further consideration.",
  "updatedAt": "2025-04-16T10:20:30Z",
  "status": "Active"
}
` ` `
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Review updated successfully
- `400 Bad Request` : Invalid review data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to update this review
- `404 Not Found` : Review not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Updates an existing review. Users can only update their own reviews, and typically within a limited timeframe after submission (e.g., 24 hours).

### ### 6.5 Delete Review

**\*\*Endpoint URL\*\*:** `/api/reviews/{reviewId}`

**\*\*HTTP Method\*\*:** `DELETE`

**\*\*Request Headers\*\*:**

```

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "rev123",

"message": "Review successfully deleted"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Review deleted successfully
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to delete this review
- `404 Not Found` : Review not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Deletes a review. Users can only delete their own reviews, and typically within a limited timeframe after submission.

### ### 6.6 Report Review

**\*\*Endpoint URL\*\*:** `/api/reviews/{reviewId}/report`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

```
{  
  "reason": "inappropriate",  
  "details": "This review contains offensive language."  
}
```

```

**\*\*Response Body Schema\*\*:**

```json

```
{  
  "id": "rep456",  
  "reviewId": "rev123",  
  "reporterId": "u789",  
  "reason": "inappropriate",  
  "details": "This review contains offensive language.",  
  "status": "Pending",  
  "createdAt": "2025-04-14T15:30:00Z"  
}
```

`, `

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Report submitted successfully
- `400 Bad Request` : Invalid report data
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : Review not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Reports a review for moderation. This flags the review for administrator review without immediately removing it from view.

### ### 6.7 Moderate Review (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/reviews/{reviewId}/moderate`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

`, `

Content-Type: application/json

Authorization: Bearer {token}

`, `

**\*\*Request Body Schema\*\*:**

```json

{

"action": "approve",

"notes": "Review meets community guidelines."

```
}  
` ``
```

**\*\*Response Body Schema\*\*:**

```
` `` json  
{  
  "id": "rev123",  
  "status": "Approved",  
  "moderatedAt": "2025-04-14T16:00:00Z",  
  "moderatorId": "u001",  
  "moderatorName": "Admin User",  
  "notes": "Review meets community guidelines."  
}  
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- ` 200 OK ` : Review moderated successfully
- ` 400 Bad Request ` : Invalid moderation action
- ` 401 Unauthorized ` : Invalid or missing token
- ` 403 Forbidden ` : User doesn't have admin role
- ` 404 Not Found ` : Review not found
- ` 500 Internal Server Error ` : Server error

**\*\*Description\*\*:** Allows administrators to moderate a review - approving it, flagging it, or removing it from the system.

### 6.8 Get Reported Reviews (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/reviews/reported`



**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Request Parameters\*\*:**

- `status` (optional): Filter by report status ('pending', 'resolved')
- `page` (optional): Page number for pagination
- `limit` (optional): Number of reports per page

**\*\*Response Body Schema\*\*:**

` ` `json

{

  "reports": [

    {

      "id": "rep456",

      "reviewId": "rev123",

      "review": {

        "id": "rev123",

        "donationId": "don456",

        "reviewerId": "u789",

        "reviewerName": "John Smith",

        "rating": 2,

        "comment": "The food was of poor quality.",

        "status": "Flagged"

    },

    "reporterId": "u012",

    "reporterName": "Fresh Grocers",

```

    "reason": "disputed",
    "details": "This review is inaccurate. The food was freshly prepared that morning.",
    "status": "Pending",
    "createdAt": "2025-04-14T15:30:00Z"
  }
],
"pagination": {
  "total": 5,
  "page": 1,
  "limit": 10,
  "pages": 1
}
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves a list of reviews that have been reported, along with the report details, for administrator moderation.

## ## 7. Statistics APIs

Based on the codebase analysis, particularly the donation service and admin dashboard components, I'll document the Statistics APIs for the FoodShare application.

### ### 7.1 Get User Statistics

**\*\*Endpoint URL\*\*:** `/api/users/{userId}/stats`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Response Body Schema\*\*:**

` ` `json

{

  "donations": {

    "total": 27,

    "active": 3,

    "completed": 24

  },

  "received": {

    "total": 12,

    "active": 1,

    "completed": 11

  },

  "impact": {

    "foodSaved": 135,

    "unit": "kg",

    "mealsProvided": 450,

    "impactPoints": 2350

  },

  "ratings": {

    "average": 4.8,

```
"count": 25
},
"activity": {
  "firstActivity": "2025-01-15T10:20:30Z",
  "lastActivity": "2025-04-12T14:30:00Z",
  "daysActive": 88
}
}
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these statistics
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves comprehensive statistics for a specific user's activity on the platform, including donation metrics, impact metrics, and ratings.

### ### 7.2 Get Donation Statistics

**\*\*Endpoint URL\*\*:** `/api/donations/{donationId}/stats`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

...

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\*:**

```json

```
{
  "views": 42,
  "requests": 3,
  "timeToMatch": {
    "hours": 4,
    "minutes": 15
  },
  "ratings": {
    "donorRating": 5,
    "beneficiaryRating": 4,
    "averageRating": 4.5
  },
  "impact": {
    "foodSaved": 15,
    "unit": "kg",
    "mealsProvided": 50
  },
  "timeline": [
    {
      "status": "Created",
      "timestamp": "2025-04-10T09:00:00Z"
    },
    {
      "status": "Approved",
      "timestamp": "2025-04-10T10:30:00Z"
    },
    {
```

```

    "status": "Matched",
    "timestamp": "2025-04-10T13:15:00Z"
  },
  {
    "status": "Completed",
    "timestamp": "2025-04-10T16:45:00Z"
  }
]
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these statistics
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves detailed statistics about a specific donation, including metrics on how quickly it was matched and its impact.

### ### 7.3 Get Platform Statistics (Admin)

**\*\*Endpoint URL\*\*:** `/api/admin/stats`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` `` `

Authorization: Bearer {token}

```

**\*\*Request Parameters\*\*:**

- `period` (optional): Time period for metrics ('day', 'week', 'month', 'year', 'all')
- `startDate` (optional): Start date for custom period
- `endDate` (optional): End date for custom period

**\*\*Response Body Schema\*\*:**

```json

```
{
  "users": {
    "total": 248,
    "donors": 112,
    "beneficiaries": 96,
    "ngos": 40,
    "growth": {
      "rate": 8.3,
      "trend": [65, 59, 80, 81, 56, 55, 40]
    }
  },
  "donations": {
    "total": 437,
    "active": 53,
    "completed": 384,
    "growth": {
      "rate": 12.5,
      "trend": [45, 52, 49, 60, 55, 58, 62]
    }
  },
  "impact": {
```

```

    "foodSaved": 2750,
    "unit": "kg",
    "mealsProvided": 9167,
    "growth": {
      "rate": 15.2,
      "trend": [200, 250, 280, 310, 330, 350, 400]
    },
    "system": {
      "pendingApprovals": 8,
      "pendingReports": 3,
      "averageResponseTime": "4.2 hours"
    }
  },
  ...

```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Admin role only)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid date range
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have admin role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves comprehensive platform-wide statistics for administrative purposes, including user metrics, donation metrics, and overall impact.

### ### 7.4 Get Impact Report

**\*\*Endpoint URL\*\*:** `/api/impact-report`



**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Request Parameters\*\*:**

- `userId` (optional): Filter report by specific user
- `timeframe` (optional): Time period for report ('month', 'quarter', 'year')
- `format` (optional): Response format ('json', 'pdf', 'csv')

**\*\*Response Body Schema\*\* (JSON format):**

` ` `json

```
{  
  "user": {  
    "id": "u789",  
    "name": "John Smith",  
    "role": "Donor",  
    "joinedDate": "2025-01-15T10:20:30Z"  
  },  
  "timeframe": {  
    "period": "quarter",  
    "start": "2025-01-01T00:00:00Z",  
    "end": "2025-03-31T23:59:59Z"  
  },  
  "summary": {  
    "donations": 12,  
    "foodSaved": 60,  
    "unit": "kg",
```

```
"mealsProvided": 200,
"carbonOffset": 250,
"unit": "kg CO2e"
},
"comparison": {
  "percentile": 85,
  "communityAverage": {
    "donations": 8,
    "foodSaved": 40
  }
},
"badges": [
  {
    "name": "Food Rescue Hero",
    "description": "Saved over 50kg of food in a quarter",
    "earnedAt": "2025-03-15T14:20:00Z"
  }
],
"monthlyBreakdown": [
  {
    "month": "January",
    "donations": 3,
    "foodSaved": 15
  },
  {
    "month": "February",
    "donations": 4,
    "foodSaved": 20
  },
  {
    "month": "March",
```

```
"donations": 5,  
"foodSaved": 25  
}  
]  
}  
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid parameters
- `401 Unauthorized` : Invalid or missing token
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Generates a detailed impact report showing the environmental and social impact of a user's activities on the platform. The report can be filtered by timeframe and delivered in various formats.

This completes the documentation for the Review and Statistics APIs in the FoodShare application.

I'll continue with documenting the remaining APIs for the FoodShare application. Let me now document the Donation APIs. # FoodShare API Documentation - Donation APIs

Based on my analysis of the codebase, particularly the donation service and related components, I'll now document the Donation APIs available in the FoodShare application.

## ## 8. Donation APIs

### ### 8.1 Create Donation

**\*\*Endpoint URL\*\*:** `/api/donations/create`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

``

Content-Type: application/json

Authorization: Bearer {token}

``

**\*\*Request Body Schema\*\*:**

```json

{

"title": "Fresh Vegetables Bundle",

"foodType": "Vegetables",

"quantity": 15,

"unit": "kg",

"category": "Vegetables",

"expiresAt": "2025-04-20T00:00:00Z",

"location": "123 Main St, Cityville",

"description": "Assorted fresh vegetables including carrots, broccoli, and bell peppers",

"pickupTimeStart": "2025-04-15T09:00:00Z",

```
"pickupTimeEnd": "2025-04-15T17:00:00Z"
}
` ` `
```

**\*\*Response Body Schema\*\*:**

```
` ` `json
{
  "id": "don001",
  "title": "Fresh Vegetables Bundle",
  "foodType": "Vegetables",
  "quantity": 15,
  "unit": "kg",
  "category": "Vegetables",
  "expiresAt": "2025-04-20T00:00:00Z",
  "status": "Pending Approval",
  "location": "123 Main St, Cityville",
  "description": "Assorted fresh vegetables including carrots, broccoli, and bell peppers",
  "pickupTimeStart": "2025-04-15T09:00:00Z",
  "pickupTimeEnd": "2025-04-15T17:00:00Z",
  "donorId": "usr112",
  "createdAt": "2025-04-14T15:30:00Z"
}
` ` `
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Donor role)

**\*\*Status Codes and Messages\*\*:**

- ` 201 Created ` : Donation listing created successfully
- ` 400 Bad Request ` : Invalid or missing required fields
- ` 401 Unauthorized ` : Invalid or missing token
- ` 403 Forbidden ` : User doesn't have donor role

- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Creates a new food donation listing. All new donations are initially set to "Pending Approval" status and must be approved by an administrator before they become visible to beneficiaries.

### ### 8.2 Get All Donations

**\*\*Endpoint URL\*\***: `/api/donations/list`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\***:

`, ` ,`

Authorization: Bearer {token}

`, ` ,`

**\*\*Request Parameters\*\***:

- `status` (optional): Filter by donation status
- `category` (optional): Filter by food category
- `page` (optional): Page number for pagination
- `limit` (optional): Number of items per page
- `sortBy` (optional): Field to sort by (e.g., 'createdAt', 'expiresAt')
- `sortOrder` (optional): Sort direction ('asc' or 'desc')

**\*\*Response Body Schema\*\***:

`` `json

{

  "donations": [

    {

      "id": "don001",

      "title": "Fresh Vegetables Bundle",

```
"foodType": "Vegetables",
"quantity": 15,
"unit": "kg",
"category": "Vegetables",
"expiresAt": "2025-04-20T00:00:00Z",
"status": "Approved",
"location": "123 Main St, Cityville",
"donorName": "Green Grocers",
"createdAt": "2025-04-10T15:30:00Z"
},
{
  "id": "don002",
  "title": "Bakery Items",
  "foodType": "Baked Goods",
  "quantity": 10,
  "unit": "items",
  "category": "Baked Goods",
  "expiresAt": "2025-04-16T00:00:00Z",
  "status": "In Transit",
  "location": "456 Flour Ave, Local City",
  "donorName": "Daily Bread",
  "createdAt": "2025-04-12T10:15:00Z"
}
],
"pagination": {
  "total": 42,
  "page": 1,
  "limit": 10,
  "pages": 5
}
}
```

`, ``, `

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves a list of donation listings. Users with different roles can access different sets of donations: donors see their own donations, beneficiaries see available donations, and admins see all donations.

### ### 8.3 Get Donation by ID

**\*\*Endpoint URL\*\*:** `/api/donations/{id}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

`, ``, `

Authorization: Bearer {token}

`, ``, `

**\*\*Response Body Schema\*\*:**

`` `json

```
{
  "id": "don001",
  "title": "Fresh Vegetables Bundle",
  "foodType": "Vegetables",
  "quantity": 15,
  "unit": "kg",
```



```
"category": "Vegetables",
"expiresAt": "2025-04-20T00:00:00Z",
"status": "Approved",
"createdAt": "2025-04-10T15:30:00Z",
"description": "Assorted fresh vegetables including carrots, broccoli, and bell peppers",
"donor": {
  "id": "usr112",
  "name": "Green Grocers",
  "phone": "(555) 123-4567",
  "address": "123 Main St, Cityville"
},
"recipient": {
  "id": "usr456",
  "name": "Community Food Bank",
  "phone": "(555) 987-6543",
  "address": "456 Oak Ave, Townsville"
},
"ngo": {
  "id": "usr789",
  "name": "Food Rescue Initiative",
  "phone": "(555) 456-7890"
},
"logistics": {
  "pickupAddress": "123 Main St, Cityville",
  "dropoffAddress": "456 Oak Ave, Townsville",
  "deliveryType": "NGO Logistics",
  "eta": "2025-04-15T15:30:00Z"
},
"timestamps": {
  "created": "2025-04-10T15:30:00Z",
  "accepted": "2025-04-11T10:30:00Z",
```

```

    "readyForPickup": "2025-04-14T10:00:00Z",
    "inTransit": "2025-04-14T13:45:00Z",
    "delivered": null,
    "cancelled": null
  },
  "flags": [
    {
      "issue": "Incorrect quantity reported",
      "date": "2025-04-14T08:30:00Z"
    }
  ]
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view this donation
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves detailed information about a specific donation. The amount of information shown may vary based on the user's role and relationship to the donation.

### ### 8.4 Update Donation Status

**\*\*Endpoint URL\*\*:** `/api/donations/{id}/status`

**\*\*HTTP Method\*\*:** `PUT`

**\*\*Request Headers\*\*:**

```

Content-Type: application/json

Authorization: Bearer {token}

```

**\*\*Request Body Schema\*\*:**

```json

{

"status": "In Transit",

"notes": "Picked up by delivery driver at 2:30 PM"

}

```

**\*\*Response Body Schema\*\*:**

```json

{

"id": "don001",

"status": "In Transit",

"previousStatus": "Ready for Pickup",

"updatedAt": "2025-04-14T14:30:00Z",

"updatedBy": "usr789",

"timestamps": {

"inTransit": "2025-04-14T14:30:00Z"

},

"notes": "Picked up by delivery driver at 2:30 PM"

}

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Status updated successfully
- `400 Bad Request` : Invalid status transition
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to update this donation
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Updates the status of a donation. Different roles can make different status transitions: donors can mark donations as "Ready for Pickup", NGOs can mark donations as "In Transit", beneficiaries can mark donations as "Delivered", and admins can make any status change.

### ### 8.5 Get Donor's Donations

**\*\*Endpoint URL\*\*:** `/api/donations/donor/{userId}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

`, `

Authorization: Bearer {token}

`, `

**\*\*Request Parameters\*\*:**

- `status` (optional): Filter by donation status
- `page` (optional): Page number for pagination
- `limit` (optional): Number of items per page
- `sortBy` (optional): Field to sort by
- `sortOrder` (optional): Sort direction ('asc' or 'desc')

**\*\*Response Body Schema\*\*:**

```
```json
{
  "donations": [
    {
      "id": "don001",
      "title": "Fresh Vegetables Bundle",
      "quantity": "15 kg",
      "category": "Vegetables",
      "expiresAt": "2025-04-20T00:00:00Z",
      "status": "Approved",
      "recipientName": "Community Food Bank",
      "createdAt": "2025-04-10T15:30:00Z"
    }
  ],
  "pagination": {
    "total": 12,
    "page": 1,
    "limit": 10,
    "pages": 2
  },
  "stats": {
    "pending": 2,
    "approved": 5,
    "inTransit": 1,
    "completed": 4
  }
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these donations
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all donations created by a specific donor. Donors can only access their own donations, while admins can access any donor's donations.

### ### 8.6 Claim Donation

**\*\*Endpoint URL\*\*:** `/api/donations/claim`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

`, `,

Content-Type: application/json

Authorization: Bearer {token}

`, `,

**\*\*Request Body Schema\*\*:**

```json

```
{  
  "donationId": "don001",  
  "beneficiaryId": "usr456",  
  "notes": "Will pick up ourselves on Wednesday morning."  
}
```

```

**\*\*Response Body Schema\*\*:**

```
```\n\n{\n  "id": "don001",\n  "status": "Accepted",\n  "previousStatus": "Approved",\n  "claimedAt": "2025-04-14T15:30:00Z",\n  "beneficiaryId": "usr456",\n  "beneficiaryName": "Community Food Bank",\n  "message": "Donation successfully claimed"\n}\n```\n
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary or NGO role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Donation claimed successfully
- `400 Bad Request` : Invalid donation for claiming
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to claim donations
- `404 Not Found` : Donation not found
- `409 Conflict` : Donation already claimed
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Claims an approved donation, transitioning its status to "Accepted" and associating it with the claiming beneficiary or NGO.

### ### 8.7 Get Claimed Donations

**\*\*Endpoint URL\*\*:** `/api/donations/claimed/{beneficiaryId}`

**\*\*HTTP Method\*\*:** `GET`

**\*\*Request Headers\*\*:**

` ` `

Authorization: Bearer {token}

` ` `

**\*\*Request Parameters\*\*:**

- `status` (optional): Filter by donation status
- `page` (optional): Page number for pagination
- `limit` (optional): Number of items per page
- `sortBy` (optional): Field to sort by
- `sortOrder` (optional): Sort direction ('asc' or 'desc')

**\*\*Response Body Schema\*\*:**

` ` ` json

```
{
  "donations": [
    {
      "id": "don001",
      "title": "Fresh Vegetables Bundle",
      "donorName": "Green Grocers",
      "quantity": "15 kg",
      "category": "Vegetables",
      "expiresAt": "2025-04-20T00:00:00Z",
      "status": "In Transit",
      "location": "123 Main St, Cityville",
      "claimedAt": "2025-04-11T10:30:00Z"
    }
  ],
}
```



```
"pagination": {  
  "total": 8,  
  "page": 1,  
  "limit": 10,  
  "pages": 1  
}  
}  
` ``
```

**\*\*Authentication Requirements\*\***: JWT Token Required

**\*\*Status Codes and Messages\*\***:

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these donations
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Retrieves all donations claimed by a specific beneficiary or NGO. Beneficiaries and NGOs can only access their own claimed donations, while admins can access any user's claimed donations.

### ### 8.8 Get Available Donations

**\*\*Endpoint URL\*\***: `/api/donations/available`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\***:

` ``

Authorization: Bearer {token}

` ``

**\*\*Request Parameters\*\*:**

- `category` (optional): Filter by food category
- `distance` (optional): Maximum distance in kilometers from user's location
- `lat` (optional): User's latitude for distance calculation
- `lng` (optional): User's longitude for distance calculation
- `expiryRange` (optional): Filter by expiration timeframe
- `page` (optional): Page number for pagination
- `limit` (optional): Number of items per page
- `sortBy` (optional): Field to sort by
- `sortOrder` (optional): Sort direction ('asc' or 'desc')

**\*\*Response Body Schema\*\*:**

` `` `json

```
{  
  "donations": [  
    {  
      "id": "don101",  
      "title": "Fresh Organic Vegetables",  
      "type": "Vegetables",  
      "quantity": 8,  
      "unit": "kg",  
      "expiresAt": "2025-04-18T00:00:00Z",  
      "status": "Available",  
      "category": "Vegetables",  
      "location": "123 Green St, Local City",  
      "donorName": "Green Fields Farm",  
      "distance": "1.2 km"  
    }  
  ],  
  "pagination": {
```

```
"total": 15,  
"page": 1,  
"limit": 10,  
"pages": 2  
}  
}  
` ``
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary or NGO role)

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have appropriate role
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves all currently available (approved but unclaimed) donations, with options for filtering by various criteria.

### ### 8.9 Request Donation

**\*\*Endpoint URL\*\*:** `/api/donations/request`

**\*\*HTTP Method\*\*:** `POST`

**\*\*Request Headers\*\*:**

` ``

Content-Type: application/json

Authorization: Bearer {token}

` ``

**\*\*Request Body Schema\*\*:**

```
```json
{
  "donationId": "don001",
  "beneficiaryId": "usr456",
  "message": "We would like to distribute this food to our community members.",
  "requestType": "pickup"
}
```
```

**\*\*Response Body Schema\*\*:**

```
```json
{
  "id": "req789",
  "donationId": "don001",
  "beneficiaryId": "usr456",
  "donorId": "usr112",
  "status": "Pending",
  "message": "We would like to distribute this food to our community members.",
  "requestType": "pickup",
  "createdAt": "2025-04-14T15:30:00Z"
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required (Beneficiary or NGO role)

**\*\*Status Codes and Messages\*\*:**

- `201 Created` : Request submitted successfully
- `400 Bad Request` : Invalid request data
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to request donations

- `404 Not Found` : Donation not found
- `409 Conflict` : Donation already requested or claimed
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Submits a request for a donation before actually claiming it, allowing donors to review and approve requests from multiple beneficiaries.

### ### 8.10 Search Donations

**\*\*Endpoint URL\*\***: `/api/donations/search`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\***:

`,`, `

Authorization: Bearer {token}

`,`, `

**\*\*Request Parameters\*\***:

- `query` (required): Search term for donation title, description, or donor name
- `category` (optional): Filter by food category
- `status` (optional): Filter by donation status
- `locationName` (optional): Location name for proximity search
- `lat` (optional): Latitude for proximity search
- `lng` (optional): Longitude for proximity search
- `radius` (optional): Search radius in kilometers
- `fromDate` (optional): Start date for creation date filter
- `toDate` (optional): End date for creation date filter
- `page` (optional): Page number for pagination
- `limit` (optional): Number of items per page

**\*\*Response Body Schema\*\*:**

```
```json
{
  "donations": [
    {
      "id": "don101",
      "title": "Fresh Organic Vegetables",
      "type": "Vegetables",
      "quantity": 8,
      "unit": "kg",
      "expiresAt": "2025-04-18T00:00:00Z",
      "status": "Available",
      "category": "Vegetables",
      "location": "123 Green St, Local City",
      "donorName": "Green Fields Farm",
      "distance": "1.2 km",
      "match": 0.85
    }
  ],
  "pagination": {
    "total": 3,
    "page": 1,
    "limit": 10,
    "pages": 1
  }
}
```
```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `400 Bad Request` : Invalid search parameters
- `401 Unauthorized` : Invalid or missing token
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Performs a comprehensive search across donations with multiple filtering criteria. The search results are ranked by relevance to the query.

### ### 8.11 Delete Donation

**\*\*Endpoint URL\*\***: `/api/donations/{id}`

**\*\*HTTP Method\*\***: `DELETE`

**\*\*Request Headers\*\***:

```

Authorization: Bearer {token}

```

**\*\*Response Body Schema\*\***:

```json

{

"id": "don001",

"message": "Donation deleted successfully"

}

```

**\*\*Authentication Requirements\*\***: JWT Token Required

**\*\*Status Codes and Messages\*\***:

- `200 OK` : Donation deleted successfully

- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to delete this donation
- `404 Not Found` : Donation not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\***: Deletes a donation listing. Donors can only delete their own donations that haven't been claimed. Admins can delete any donation.

### ### 8.12 Get Donation Statistics

**\*\*Endpoint URL\*\***: `/api/donations/stats/{userId}`

**\*\*HTTP Method\*\***: `GET`

**\*\*Request Headers\*\***:

...

Authorization: Bearer {token}

...

**\*\*Response Body Schema\*\***:

```json

{

  "totalDonations": 27,

  "byStatus": {

    "pending": 3,

    "approved": 2,

    "inTransit": 1,

    "completed": 21

  },

  "byCategory": {

    "Vegetables": 8,



```

    "Fruits": 5,
    "Baked Goods": 7,
    "Canned Goods": 4,
    "Dairy": 3
  },
  "quantityDonated": 135.5,
  "estimatedServings": 450,
  "carbonFootprintSaved": 215.2,
  "timeline": [
    { "month": "Jan", "count": 3, "quantity": 15.2 },
    { "month": "Feb", "count": 4, "quantity": 20.1 },
    { "month": "Mar", "count": 5, "quantity": 25.8 },
    { "month": "Apr", "count": 6, "quantity": 30.3 }
  ],
  "activeDonations": 6,
  "pendingApprovals": 3
}
` ``

```

**\*\*Authentication Requirements\*\*:** JWT Token Required

**\*\*Status Codes and Messages\*\*:**

- `200 OK` : Successful request
- `401 Unauthorized` : Invalid or missing token
- `403 Forbidden` : User doesn't have permission to view these statistics
- `404 Not Found` : User not found
- `500 Internal Server Error` : Server error

**\*\*Description\*\*:** Retrieves donation-related statistics for a specific user. The statistics returned vary based on user role: donors see statistics about their donations, beneficiaries see statistics about received donations.

This completes the documentation for the Donation APIs in the FoodShare application. These APIs provide comprehensive functionality for creating, managing, searching, and tracking food donations throughout their lifecycle.

