Ad Manager User Guide

Complete guide to using the New Stars Radio Ad Server for managing advertisers, campaigns, and ad creatives.

Table of Contents

Overview

The Ad Manager is a complete system for:

- **Managing Advertisers**: Store information about companies/individuals who want to advertise
- **Creating Campaigns**: Set up ad campaigns with budgets, targeting, and date ranges
- **Uploading Creatives**: Add banner images for your campaigns
- **Serving Ads**: Automatically select and serve ads to users based on targeting
- **Tracking Performance**: Monitor impressions, clicks, and click-through rates

Key Concepts

- **Advertiser**: A company or individual who wants to advertise
- **Campaign**: An advertising campaign with a budget, date range, and targeting options
- **Creative**: An actual banner image/ad that belongs to a campaign
- **Impression**: When an ad is shown to a user
- **Click**: When a user clicks on an ad

Getting Started

Prerequisites

1. Make sure the ad server is running:

```
cd ad-server
docker-compose up -d
```

2. Access the API documentation at: `http://localhost:8000/docs`
3. Default admin credentials:
    - Email: `admin@newstarsradio.com`
    - Password: `changeme123`

Authentication

Most operations require authentication. You'll need to log in and use the JWT token for subsequent requests.

Step 1: Login

Using cURL:

```
curl -X POST "http://localhost:8000/api/v1/auth/login" \
  -H "Content-Type: application/json" \
  -d '{
    "email": "admin@newstarsradio.com",
    "password": "changeme123"
  }'
```

Response:

```
{
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "token_type": "bearer"
}
```

Save the token - you'll need it for all authenticated requests!

Step 2: Verify Your Session

Using cURL:

```
curl -X GET "http://localhost:8000/api/v1/auth/me" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Response:

```
{
  "id": "uuid-here",
  "email": "admin@newstarsradio.com",
  "full_name": "Admin User",
  "role": "admin",
  "is_active": true
}
```

Managing Advertisers

Advertisers are the companies or individuals who want to advertise on your platform.

Create an Advertiser

Using cURL:

```
curl -X POST "http://localhost:8000/api/v1/advertisers" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -H "Content-Type: application/json" \
  -d '{
    "name": "John Smith",
    "email": "john@example.com",
    "phone": "+1-555-0123",
    "company_name": "Smith's Bakery"
  }'
```

Response:

```
{
  "id": "advertiser-uuid",
  "name": "John Smith",
  "email": "john@example.com",
  "phone": "+1-555-0123",
  "company_name": "Smith's Bakery",
  "status": "active",
  "created_at": "2025-11-29T21:00:00",
  "updated_at": "2025-11-29T21:00:00"
}
```

Save the advertiser ID - you'll need it when creating campaigns!

List All Advertisers

```
curl -X GET "http://localhost:8000/api/v1/advertisers" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Get a Specific Advertiser

```
curl -X GET "http://localhost:8000/api/v1/advertisers/ADVERTISER_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Update an Advertiser

```
curl -X PUT "http://localhost:8000/api/v1/advertisers/ADVERTISER_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -H "Content-Type: application/json" \
  -d '{
    "phone": "+1-555-9999",
    "status": "inactive"
  }'
```

Delete an Advertiser

```
curl -X DELETE "http://localhost:8000/api/v1/advertisers/ADVERTISER_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Creating Campaigns

Campaigns define when, where, and how ads will be shown. Each campaign belongs to an advertiser.

Campaign Statuses

- **`draft`**: Campaign is being set up (won't serve ads)
- **`active`**: Campaign is live and serving ads
- **`paused`**: Campaign is temporarily stopped
- **`completed`**: Campaign has finished

Campaign Priority

Priority ranges from **1-10** (higher = more priority). When multiple campaigns are eligible, higher priority campaigns are selected first.

Create a Campaign

Using cURL:

```
curl -X POST "http://localhost:8000/api/v1/campaigns" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -H "Content-Type: application/json" \
  -d '{
    "advertiser_id": "ADVERTISER_UUID",
    "name": "Summer Sale 2025",
    "start_date": "2025-12-01T00:00:00",
    "end_date": "2025-12-31T23:59:59",
    "priority": 7,
    "impression_budget": 10000,
    "target_cities": ["New York", "Los Angeles"],
    "target_states": ["NY", "CA"]
  }'
```

Key Fields:

- `advertiser_id`: UUID of the advertiser (from previous step)
- `start_date` / `end_date`: When the campaign runs (ISO 8601 format)
- `priority`: 1-10, higher = more priority
- `impression_budget`: Maximum number of times ads can be shown
- `target_cities`: Optional array of city names for targeting
- `target_states`: Optional array of state codes (e.g., "NY", "CA")

Response:

```
{
  "id": "campaign-uuid",
  "advertiser_id": "advertiser-uuid",
  "name": "Summer Sale 2025",
  "status": "draft",
  "start_date": "2025-12-01T00:00:00",
  "end_date": "2025-12-31T23:59:59",
  "priority": 7,
  "impression_budget": 10000,
  "impressions_served": 0,
  "target_cities": ["New York", "Los Angeles"],
  "target_states": ["NY", "CA"],
  "created_at": "2025-11-29T21:00:00",
  "updated_at": "2025-11-29T21:00:00"
}
```


Save the campaign ID - you'll need it for creatives!

List Campaigns

```
# List all campaigns
curl -X GET "http://localhost:8000/api/v1/campaigns" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"

# Filter by status
curl -X GET "http://localhost:8000/api/v1/campaigns?status_filter=active" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"

# Filter by advertiser
curl -X GET "http://localhost:8000/api/v1/campaigns?advertiser_id=ADVERTISER_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Update Campaign Status

To activate a campaign (make it serve ads):

```
curl -X PUT "http://localhost:8000/api/v1/campaigns/CAMPAIGN_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -H "Content-Type: application/json" \
  -d '{
    "status": "active"
  }'
```

Update Campaign Details

```
curl -X PUT "http://localhost:8000/api/v1/campaigns/CAMPAIGN_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -H "Content-Type: application/json" \
  -d '{
    "priority": 9,
    "impression_budget": 20000
  }'
```

Managing Ad Creatives

Creatives are the actual banner images that users see. Each creative belongs to a campaign.

Upload a Creative (File Upload)

Using cURL:

```
curl -X POST "http://localhost:8000/api/v1/creatives?
campaign_id=CAMPAIGN_UUID&name=Banner%201&click_url=https://example.com&alt_text=Summer%20Sale" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -F "image_file=@/path/to/your/banner.jpg"
```

Parameters:

- `campaign_id`: UUID of the campaign

- `name`: Name for this creative
- `click_url`: Where users go when they click the ad
- `alt_text`: Alternative text for accessibility (optional)
- `image_file`: The image file to upload

Supported formats: `.jpg`, `.jpeg`, `.png`, `.gif`, `.webp`

Response:

```
{
  "id": "creative-uuid",
  "campaign_id": "campaign-uuid",
  "name": "Banner 1",
  "image_url": "/static/ads/campaign-uuid_banner.jpg",
  "image_width": 728,
  "image_height": 90,
  "click_url": "https://example.com",
  "alt_text": "Summer Sale",
  "status": "active",
  "created_at": "2025-11-29T21:00:00",
  "updated_at": "2025-11-29T21:00:00"
}
```

Create Creative with Image URL

If your image is already hosted elsewhere:

```
curl -X POST "http://localhost:8000/api/v1/creatives/with-url" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -H "Content-Type: application/json" \
  -d '{
    "campaign_id": "CAMPAIGN_UUID",
    "name": "Banner 2",
    "image_url": "https://example.com/banner.jpg",
    "image_width": 728,
    "image_height": 90,
    "click_url": "https://example.com",
    "alt_text": "Summer Sale Banner"
  }'
```

List Creatives

```
# List all creatives
curl -X GET "http://localhost:8000/api/v1/creatives" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"

# Filter by campaign
curl -X GET "http://localhost:8000/api/v1/creatives?campaign_id=CAMPAIGN_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Update Creative Status

```
curl -X PUT "http://localhost:8000/api/v1/creatives/CREATIVE_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  -H "Content-Type: application/json" \
  -d '{
    "status": "inactive"
  }'
```

Serving Ads

This is how your radio app requests ads to show to users. **No authentication required** for serving ads.

Request an Ad

Using cURL:

```
curl -X GET "http://localhost:8000/api/v1/ads/serve?
user_id=user123&city=New%20York&state=NY&width=728&height=90"
```

Parameters:

- `user_id`: Unique identifier for the user/device (required)
- `city`: User's city for targeting (optional)
- `state`: User's state for targeting (optional)
- `width`: Banner width in pixels (optional)
- `height`: Banner height in pixels (optional)

Response:

```
{
  "creative_id": "creative-uuid",
  "campaign_id": "campaign-uuid",
  "image_url": "/static/ads/campaign-uuid_banner.jpg",
  "click_url": "https://example.com",
  "alt_text": "Summer Sale",
  "width": 728,
  "height": 90,
  "impression_id": "impression-uuid"
}
```

How Ad Selection Works:

1. System finds campaigns that are:
    - Status = `active`
    - Current date is between `start_date` and `end_date`
    - `impressions_served` < `impression_budget`
    - Match targeting (if city/state provided)
2. Sorts by priority (highest first)
3. Selects a random creative from the highest priority campaign
4. Records an impression
5. Returns the ad

Track a Click

When a user clicks an ad:

```
curl -X POST "http://localhost:8000/api/v1/ads/click" \
  -H "Content-Type: application/json" \
  -d '{
    "user_id": "user123",
    "creative_id": "creative-uuid",
    "campaign_id": "campaign-uuid",
    "impression_id": "impression-uuid"
  }'
```

Response: `204 No Content` (success)

Tracking & Analytics

Monitor how your campaigns are performing.

Get Campaign Statistics

```
curl -X GET "http://localhost:8000/api/v1/reports/campaigns/CAMPAIGN_UUID/stats" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Response:

```
{
  "campaign_id": "campaign-uuid",
  "campaign_name": "Summer Sale 2025",
  "impressions": 5420,
  "clicks": 127,
  "click_through_rate": 2.34,
  "impressions_served": 5420,
  "budget_remaining": 4580,
  "budget_utilized_percentage": 54.2
}
```

Get Statistics with Date Range

```
curl -X GET "http://localhost:8000/api/v1/reports/campaigns/CAMPAIGN_UUID/stats?start_date=2025-12-
01T00:00:00&end_date=2025-12-15T23:59:59" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Get All Campaigns Statistics

```
curl -X GET "http://localhost:8000/api/v1/reports/campaigns/stats" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Get Creative Statistics

```
curl -X GET "http://localhost:8000/api/v1/reports/creatives/CREATIVE_UUID/stats" \
  -H "Authorization: Bearer YOUR_TOKEN_HERE"
```

Response:

```
{
  "creative_id": "creative-uuid",
  "creative_name": "Banner 1",
  "impressions": 3200,
  "clicks": 85,
  "click_through_rate": 2.66
}
```

Complete Workflow Example

Here's a complete example of setting up an ad campaign from start to finish:

1. Login

```
TOKEN=$(curl -s -X POST "http://localhost:8000/api/v1/auth/login" \
  -H "Content-Type: application/json" \
  -d '{"email":"admin@newstarsradio.com","password":"changeme123"}' \
  | jq -r '.access_token')
```

2. Create Advertiser

```
ADVERTISER_ID=$(curl -s -X POST "http://localhost:8000/api/v1/advertisers" \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "name": "Jane Doe",
    "email": "jane@example.com",
    "company_name": "Doe Enterprises"
  }' | jq -r '.id')
```

3. Create Campaign

```
CAMPAIGN_ID=$(curl -s -X POST "http://localhost:8000/api/v1/campaigns" \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d "{
    \"advertiser_id\": \"$ADVERTISER_ID\",
    \"name\": \"Holiday Special\",
    \"start_date\": \"2025-12-01T00:00:00\",
    \"end_date\": \"2025-12-31T23:59:59\",
    \"priority\": 8,
    \"impression_budget\": 5000,
    \"target_states\": [\"NY\", \"CA\"]
  }" | jq -r '.id')
```

4. Activate Campaign

```
curl -X PUT "http://localhost:8000/api/v1/campaigns/$CAMPAIGN_ID" \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"status": "active"}'
```

5. Upload Creative

```
CREATIVE_ID=$(curl -s -X POST "http://localhost:8000/api/v1/creatives?
campaign_id=$CAMPAIGN_ID&name=Holiday%20Banner&click_url=https://example.com&alt_text=Holiday%20Special" \
  -H "Authorization: Bearer $TOKEN" \
  -F "image_file=@banner.jpg" | jq -r '.id')
```

6. Serve an Ad (from your app)

```
curl -X GET "http://localhost:8000/api/v1/ads/serve?user_id=user123&state=NY"
```

7. Track Click (when user clicks)

```
curl -X POST "http://localhost:8000/api/v1/ads/click" \
  -H "Content-Type: application/json" \
  -d "{
    \"user_id\": \"user123\",
    \"creative_id\": \"$CREATIVE_ID\",
    \"campaign_id\": \"$CAMPAIGN_ID\"
  }"
```

8. Check Statistics

```
curl -X GET "http://localhost:8000/api/v1/reports/campaigns/$CAMPAIGN_ID/stats" \
  -H "Authorization: Bearer $TOKEN"
```

Using the Interactive API Docs

The easiest way to explore and test the API is through the interactive Swagger documentation:

1. **Start the server:**

   ```
   cd ad-server
   docker-compose up -d
   ```

2. **Open your browser:**
   Navigate to `http://localhost:8000/docs`

3. **Authenticate:**

   - Click the **"Authorize"** button (green button with lock icon)
   - Click **"Authorize"** in the popup
   - Enter: `Bearer YOUR_TOKEN_HERE` (get token from login endpoint first)
   - Click **"Authorize"** and **"Close"**

4. **Try endpoints:**

   - Click on any endpoint to expand it
   - Click **"Try it out"**

- Fill in the parameters
- Click **"Execute"**
- See the response below

Tips for Using the Docs:

- **Login first**: Use the `/auth/login` endpoint to get a token
- **Authorize**: Always authorize before trying protected endpoints
- **Copy IDs**: When creating resources, copy the returned IDs for use in other endpoints
- **Check schemas**: Click "Schema" to see required fields and formats

---

Common Tasks

Pause a Campaign

```
curl -X PUT "http://localhost:8000/api/v1/campaigns/CAMPAIGN_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"status": "paused"}'
```

Increase Campaign Budget

```
curl -X PUT "http://localhost:8000/api/v1/campaigns/CAMPAIGN_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"impression_budget": 20000}'
```

Deactivate a Creative

```
curl -X PUT "http://localhost:8000/api/v1/creatives/CREATIVE_UUID" \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"status": "inactive"}'
```

Check Campaign Budget Usage

```
curl -X GET "http://localhost:8000/api/v1/reports/campaigns/CAMPAIGN_UUID/stats" \
  -H "Authorization: Bearer YOUR_TOKEN" | jq '.budget_utilized_percentage'
```

Troubleshooting

"No ads available at this time"

- Check that campaign status is `active`
- Verify campaign dates include today
- Ensure `impressions_served` < `impression_budget`
- Make sure campaign has at least one active creative
- Check targeting matches (if city/state provided)

"Campaign not found"

- Verify the campaign UUID is correct
- Check that you're authenticated
- Ensure the campaign wasn't deleted

"Unauthorized" errors

- Make sure you're including the `Authorization: Bearer TOKEN` header
- Check that your token hasn't expired (default: 60 minutes)
- Re-login to get a new token

File upload fails

- Check file size (max 5MB)
- Verify file extension is allowed (.jpg, .jpeg, .png, .gif, .webp)
- Ensure you're using `multipart/form-data` format

Best Practices

1. **Always activate campaigns** after creating them (status = "draft" by default)
2. **Set realistic budgets** - campaigns stop serving when budget is reached
3. **Use targeting wisely** - empty targeting = ads shown to everyone
4. **Monitor performance** - check stats regularly to optimize campaigns
5. **Test before going live** - use draft status to set up campaigns first
6. **Multiple creatives** - add multiple creatives per campaign for variety
7. **Priority management** - use higher priority (8-10) for important campaigns

---

Next Steps

- Explore the API documentation at `http://localhost:8000/docs`
- Set up your first campaign
- Integrate ad serving into your radio app
- Monitor performance and optimize campaigns

For more help, check the main README.md or the API documentation.