

# Mock Data System Documentation

---

## Overview

---

CareLinkAI has a centralized mock data system that allows admins to toggle between real production data and mock/demo data. This is useful for:

- **Product Demonstrations:** Show the platform's capabilities without exposing real user data
- **Development & Testing:** Work with realistic data during development
- **Onboarding:** Help new users understand features with sample data
- **Screenshots & Marketing:** Generate consistent demo content for promotional materials

## How It Works

---

### Toggle Mechanism

Mock mode can be enabled through **three methods** (in order of priority):

1. **Cookie** (Highest Priority)
  - Cookie name: `carelink_mock_mode`
  - Set via Admin Tools UI or URL query parameter
  - Persists for 7 days
  - HttpOnly for security
2. **Environment Variables**
  - `SHOW_SITE MOCKS=1` or `SHOW_SITE MOCKS=true`
  - `NEXT_PUBLIC_SHOW MOCK_DASHBOARD=1` or `NEXT_PUBLIC_SHOW MOCK_DASHBOARD=true`
3. **Development Mode** (Fallback)
  - When `NODE_ENV=development`, APIs return mock data as a fallback if no real data exists
  - This ensures developers always have data to work with

## Admin Controls

Admins can control mock mode through:

### 1. Admin Tools UI

- Navigate to: `/admin/tools`
- Find "Demo Data (Mock Mode)" section
- Click "Enable" or "Disable" buttons
- Status is displayed and updated in real-time

### 2. URL Query Parameter

- Append `?mock=1` to any URL to enable mock mode
- Append `?mock=0` to any URL to disable mock mode
- The middleware will set the cookie and redirect to the clean URL

### 3. API Endpoint

- GET `/api/mock-mode` - Check current status
- GET `/api/mock-mode?on=1` - Enable mock mode

- GET /api/mock-mode?on=0 - Disable mock mode
- Requires ADMIN role

## Implementation Details

### Core Utilities

#### Mock Mode Check ( src/lib/mockMode.ts )

```
import { isMockModeEnabled } from '@lib/mockMode';

// In API routes
export async function GET(request: Request) {
  const useMockData = isMockModeEnabled(request);

  if (useMockData) {
    // Return mock data
  }

  // Continue with real data logic
}
```

The `isMockModeEnabled()` function:

- Accepts a Next.js Request object
- Checks cookie first, then environment variables
- Returns `true` if mock mode is enabled, `false` otherwise
- Handles errors gracefully

### API Integration

All marketplace APIs respect the mock data toggle:

#### Provider API ( /api/marketplace/providers )

##### Mock Data Generator: src/lib/mock/providers.ts

```
import { generateMockProviders, filterMockProviders } from '@lib/mock/providers';

// Generate 12 mock providers
const mockProviders = generateMockProviders(12);

// Filter mock providers
const filtered = filterMockProviders(mockProviders, {
  q: 'search term',
  serviceType: 'transportation',
  city: 'San Francisco',
  state: 'CA',
  verified: 'true'
});
```

##### Mock Provider Fields:

- `businessName` : Realistic business names (e.g., "CarePlus Transport")
- `serviceTypes` : Array of services (transportation, meal-prep, housekeeping, etc.)
- `coverageArea` : Cities and states covered (Bay Area focused)
- `yearsInBusiness` : Random 3-22 years
- `isVerified` : 70% verified, 30% pending

- `credentialCount` : 1-5 credentials
- `website` : Some have websites, some don't
- `photoUrl` : Random avatar images

#### API Behavior:

1. Checks if mock mode is enabled
2. If enabled, generates and returns mock providers with proper filtering/pagination
3. If disabled, queries database for real providers
4. On error in development/mock mode, returns mock data as fallback
5. Sets `X-Mock-Data: true` header and `_mockMode: true` in response when returning mock data

### Caregiver API ( `/api/marketplace/caregivers` )

**Mock Data Generator:** Built-in `generateMockCaregivers()` function

```
const mockCaregivers = generateMockCaregivers(12, specialtyOptions);
```

#### Mock Caregiver Fields:

- `name` : Realistic first/last name combinations
- `city / state` : Bay Area cities
- `hourlyRate` : \$20-40/hr range
- `yearsExperience` : 1-15 years
- `specialties` : 2-4 specialties per caregiver
- `backgroundCheckStatus` : Mix of CLEAR, PENDING, NOT\_STARTED
- `photoUrl` : Random profile photos from randomuser.me
- `ratingAverage` : 3.5-5.0 stars
- `reviewCount` : 5-54 reviews
- `badges` : Based on experience, reviews, and background check

#### API Behavior:

1. Checks if mock mode is enabled via cookie/env
2. If enabled AND no real results found, returns mock caregivers
3. On error in development/mock mode, returns mock data
4. Real data is always preferred when available and mock mode is off

## Frontend Components

Components can check mock mode status via:

### Server Components (App Router)

```
import { cookies } from 'next/headers';

export default function ServerComponent() {
  const mockCookie = cookies().get('carelink_mock_mode')?.value?.toString().trim().toLowerCase() || '';
  const showMock = ['1', 'true', 'yes', 'on'].includes(mockCookie);

  if (showMock) {
    // Show mock data UI
  }
}
```

## Client Components

```
'use client';

export default function ClientComponent() {
  const [mockMode, setMockMode] = useState(false);

  useEffect(() => {
    fetch('/api/runtime/mocks')
      .then(res => res.json())
      .then(data => setMockMode(data.show));
  }, []);

  // Use mockMode state
}
```

## Feature Coverage

### ✓ Features Respecting Mock Data Toggle

Feature	API Endpoint	Mock Data Generator	Status
<b>Providers</b>	/api/marketplace/providers	src/lib/mock/providers.ts	✓ DONE
<b>Caregivers</b>	/api/marketplace/caregivers	Built-in function	✓ DONE
<b>Operator Dashboard</b>	/operator/page.tsx	Built-in	✓ DONE
<b>Homes Listings</b>	/operator/homes/page.tsx	Built-in	✓ DONE
<b>Residents</b>	/operator/residents/page.tsx	Built-in	✓ DONE

### 🔄 Features To Add Mock Support

When implementing new features, follow this pattern:

1. **Create Mock Data Generator** (if needed)

- Place in `src/lib/mock/[feature].ts`
- Export generation and filtering functions
- Use realistic data that mirrors production

2. **Update API Route**

```
``typescript
import { isMockModeEnabled } from '@lib/mockMode';
import { generateMock[Feature] } from '@lib/mock/[feature]';
```

```
export async function GET(request: Request) {
  const useMockData = isMockModeEnabled(request);
```

```
    if (useMockData) {
      // Generate and return mock data
    }

    // Real data logic
```

```
  }
  ...
```

### 1. Add Response Headers

```
typescript
headers: {
  'X-Mock-Data': 'true',
  'Cache-Control': 'no-store', // Don't cache mock data
}
```

### 2. Include Mock Indicator

```
typescript
{
  data: mockData,
  _mockMode: true,
  _fallback: false, // true if this is an error fallback
}
```

## Testing

### Manual Testing

#### 1. Enable Mock Mode

```
```bash
# Visit admin tools
open http://localhost:3000/admin/tools

# Or append query parameter
open http://localhost:3000/marketplace?mock=1
```
```

#### 1. Check Provider API

```
```bash
# With mock mode enabled
curl -H "Cookie: carelink_mock_mode=1" http://localhost:3000/api/marketplace/providers

# Response should include _mockMode: true
```
```

#### 1. Check Caregiver API

```
bash
# With mock mode enabled
curl -H "Cookie: carelink_mock_mode=1" http://localhost:3000/api/marketplace/caregivers
```

## 2. Verify Header

```
```bash
curl -I -H "Cookie: carelink_mock_mode=1" http://localhost:3000/api/marketplace/providers

# Should include: X-Mock-Data: true
```
```

## 1. Disable Mock Mode

```
bash
# Visit admin tools and click "Disable"
# Or append query parameter
open http://localhost:3000/marketplace?mock=0
```

## Automated Testing

```
import { isMockModeEnabled } from '@lib/mockMode';

describe('Mock Mode', () => {
  it('should detect mock mode from cookie', () => {
    const request = new Request('http://localhost:3000', {
      headers: {
        'Cookie': 'carelink_mock_mode=1'
      }
    });

    expect(isMockModeEnabled(request)).toBe(true);
  });

  it('should return false when cookie is not set', () => {
    const request = new Request('http://localhost:3000');
    expect(isMockModeEnabled(request)).toBe(false);
  });
});
```

## Security Considerations

1. **Admin-Only Toggle:** Only users with ADMIN role can toggle mock mode via `/api/mock-mode`
2. **HttpOnly Cookie:** The `carelink_mock_mode` cookie is HttpOnly, preventing client-side JavaScript access
3. **No Cache for Mock Data:** Mock data responses include `Cache-Control: no-store` to prevent caching
4. **Clear Indicators:** Responses include `_mockMode: true` and `X-Mock-Data: true` header to clearly indicate mock data
5. **Production Safety:** Environment variables can be set to enforce production mode even if cookies are present

## Troubleshooting

### Issue: Mock mode enabled but seeing real data

#### Possible causes:

1. Cookie not being sent with request
2. API endpoint not checking mock mode
3. Caching issue

**Solutions:**

- Clear browser cookies and re-enable mock mode
- Check developer tools → Application → Cookies for `carelink_mock_mode=1`
- Verify API includes `isMockModeEnabled(request)` check
- Hard refresh page (Cmd+Shift+R or Ctrl+Shift+R)

**Issue: Can't disable mock mode****Possible causes:**

1. Environment variable overriding cookie
2. Browser cache issue

**Solutions:**

- Check `.env` file for `SHOW_SITE MOCKS=1` and remove/comment it
- Clear browser cookies manually
- Try incognito/private browsing mode

**Issue: Some features show mock data, others don't****Possible cause:**

- Feature hasn't been updated to respect mock data toggle

**Solution:**

- Follow the "Features To Add Mock Support" guide above
- Check the feature's API endpoint for `isMockModeEnabled()` usage

**Issue: "Providers (8)" showing instead of "Providers (5)"****Root cause identified:**

- Provider API was not respecting mock data toggle
- Mock mode was enabled, showing 12 mock providers
- After filtering, 8 mock providers were displayed

**Solution applied:**

- Updated Provider API to check `carelink_mock_mode` cookie
- Now respects admin toggle like other features
- When disabled, shows real seeded providers (5)

**Best Practices**

1. **Always Check Mock Mode:** Every new API endpoint should check `isMockModeEnabled(request)`
2. **Realistic Mock Data:** Mock data should mirror production data structure and values
3. **Clear Indicators:** Always include `_mockMode` and `X-Mock-Data` when returning mock data
4. **Don't Mix:** Don't mix mock and real data in a single response
5. **Fallback Strategy:** In development, use mock data as fallback when database is empty
6. **Filter Support:** Mock data generators should support the same filters as real APIs
7. **Pagination:** Mock data should respect pagination parameters
8. **Performance:** Cache mock data generation when possible (but don't cache responses)

**Future Enhancements**

- [ ] Add visual indicator in UI when mock mode is enabled
- [ ] Admin dashboard showing which features have mock support

- [ ] Seed database with mock data (instead of runtime generation)
- [ ] Mock data presets for different demo scenarios
- [ ] API to customize mock data parameters
- [ ] Automated tests for all mock data generators
- [ ] Mock data for messaging, appointments, and calendar features
- [ ] Mock data versioning for testing migrations

## Related Files







---

- `/src/lib/mockMode.ts` - Core utility functions
- `/src/lib/mock/providers.ts` - Provider mock data generator
- `/src/app/api/mock-mode/route.ts` - Admin toggle API
- `/src/app/api/runtime/mocks/route.ts` - Frontend status check API
- `/src/app/admin/tools/page.tsx` - Admin UI for toggling
- `/src/app/api/marketplace/providers/route.ts` - Provider API implementation
- `/src/app/api/marketplace/caregivers/route.ts` - Caregiver API implementation
- `/src/middleware.ts` - URL query parameter handling

## Changelog

---

### 2025-12-06 - Mock Data Toggle Integration

-  Created centralized `isMockModeEnabled()` utility function
-  Created `generateMockProviders()` mock data generator
-  Updated Provider API to respect mock data toggle
-  Updated Caregiver API to check mock mode cookie (not just `NODE_ENV`)
-  Added comprehensive documentation
-  Fixed issue where “Providers (8)” was showing due to mock data not respecting toggle

---

**For questions or issues,** please refer to this documentation or contact the development team.