LeadGenX Dashboard - Testing Guide

Quick Start

1. Start the Backend API

```
cd /home/ubuntu/lead_generation_api/nodejs_space
yarn start:dev
```

Wait for the message:

```
🚀 LeadGenX API is running on http://localhost:3000
```

2. Start the Dashboard

```
cd /home/ubuntu/leadgenx-dashboard
npm run dev -- -p 3001
```

Wait for the message:

```
▲ Next.js 16.1.1 (Turbopack)
- Local:         http://localhost:3001
```

3. Open the Dashboard

Navigate to: **http://localhost:3001**

Test API Key

API Key: `lgx_Wjvg8Dlde2BXzAbQ4nBnqQnGLhCVJkBuXMit2LqUsrs`

Use this key to log into the dashboard.

Creating Your Own API Key

If you want to create a new user and API key:

Step 1: Register a User

```
curl -X POST http://localhost:3000/v1/auth/register \
  -H "Content-Type: application/json" \
  -d '{
    "email": "your@company.com",
    "password": "SecurePass123!",
    "full_name": "Your Name",
    "company_name": "Your Company"
  }'
```

Response:

```json
{
  "user": { ... },
  "token": "<SESSION_TOKEN>",
  "expiresAt": "..."
}
```

Save the `token` value.

Step 2: Create an API Key

```bash
curl -X POST http://localhost:3000/v1/auth/api-keys \
  -H "Content-Type: application/json" \
  -H "Cookie: session_token=<SESSION_TOKEN>" \
  -d '{
    "organizationSlug": "your-company",
    "name": "My Dashboard Key"
  }'
```

Response:

```json
{
  "id": "...",
  "key": "lgx_...",
  "name": "My Dashboard Key",
  ...
}
```

Copy the `key` value (starts with `lgx_`).

Test Scenarios

Scenario 1: Client Management

1. **Login** with API key
2. Navigate to **Clients**
3. Click **“Add Client”**
4. Fill in:
   - Name: “Acme Corporation”
   - Industry: “Technology”
   - Website: “https://acme.com”
   - Notes: “Fortune 500 tech company”
5. Click **“Create Client”**
6. Verify the client appears in the grid
7. Click the **edit icon** (pencil)
8. Update the industry to “Software”
9. Click **“Update Client”**
10. Verify changes are saved
11. Click the **delete icon** (trash)
12. Confirm deletion

13. Verify client is removed

Scenario 2: Campaign Creation & Execution

Part A: Create Campaign

1. Navigate to **Campaigns**
2. Click **“New Campaign”**
3. **Step 1 - Basic Info**:
   - Name: “SF Tech Startups Q1 2026”
   - Client: Select a client or leave blank
   - Vertical: “Technology”
   - Click **“Next”**
4. **Step 2 - Targeting**:
   - City: “San Francisco”
   - State: “CA”
   - Country: “US”
   - Radius: 25 miles
   - Categories: “software development, saas, artificial intelligence”
   - Min Rating: 4.0
   - Click **“Next”**
5. **Step 3 - Sources**:
   - Enable **“Google Places”**
   - Enable **“Reddit Intent”**
   - Click **“Next”**
6. **Step 4 - Configuration**:
   - Review scoring weights
   - Click **“Create Campaign”**
7. Verify campaign appears in the grid with “draft” status

Part B: Run Campaign

1. Click on the campaign card to open details
2. Click **“Run Campaign”**
3. Confirm the action
4. Wait for completion (progress indicator should show)
5. Verify:
   - KPI cards update with lead counts
   - “Leads” tab shows discovered leads
   - “Run History” tab shows the completed run

Part C: View Campaign Leads

1. In the campaign detail page, click the **“Leads”** tab
2. Verify lead cards show:
   - Business name
   - Location
   - Contact badges (phone, email, website)
   - Rating and reviews
   - Campaign score

3. Click on a lead card to see more details
4. Click **“Visit”** button to open business website

Scenario 3: Lead Filtering & Search

1. Navigate to **Leads** (all leads page)
2. Verify all discovered leads across campaigns are displayed
3. Use the **search bar** to filter by business name
4. Type “coffee” and verify only coffee shops appear
5. Clear the search
6. Change the filter dropdown to **“Lead-Ready Only”**
7. Verify only leads with contact information are shown
8. Click on a lead card
9. Verify contact action buttons work:
   - **“Call”** opens tel: link
   - **“Visit”** opens website in new tab

Scenario 4: Campaign Settings

1. Open a campaign detail page
2. Click the **“Settings”** tab
3. Verify all campaign configuration is displayed:
   - Targeting details (vertical, location, radius)
   - Data sources (Google, Reddit)
   - Scoring weights breakdown
4. Verify the information matches what was entered during campaign creation

Scenario 5: Run History

1. Run the same campaign multiple times:
   - Click **“Run Campaign”**
   - Wait for completion
   - Repeat 2-3 times
2. Navigate to the **“Run History”** tab
3. Verify all runs are listed with:
   - Timestamp
   - Status (success/failed)
   - Stats (discovered, enriched, ready)
   - Logs preview
4. Verify runs are sorted by most recent first

Scenario 6: Empty States

1. **New Account**: Create a fresh API key and login
2. Verify empty states are shown for:
   - Clients page (“No clients yet” with CTA)
   - Campaigns page (“No campaigns yet” with CTA)
   - Leads page (“No leads found” message)
3. Verify CTAs work and open creation dialogs

Scenario 7: Responsive Design

1. Resize browser window to different widths:
   - Desktop (1920px)
   - Tablet (768px)
   - Mobile (375px)
2. Verify:
   - Sidebar remains accessible
   - Grid layouts adjust (3 cols → 2 cols → 1 col)
   - Cards maintain readability
   - Buttons and forms are usable

Scenario 8: Error Handling

1. **Invalid API Key**:
   - Logout
   - Try to login with “invalid_key”
   - Verify error message is shown

2. **Network Error**:
   - Stop the backend API
   - Try to perform an action in the dashboard
   - Verify error is handled gracefully
   - Restart the backend
   - Retry the action

3. **Validation Errors**:
   - Try to create a client with an empty name
   - Verify validation message is displayed

Scenario 9: Loading States

1. **Page Load**:
   - Refresh the clients page
   - Verify skeleton loaders are shown during data fetch
   - Verify smooth transition to actual content

2. **Action Loading**:
   - Run a campaign
   - Verify the button shows “Running…” during execution
   - Verify button is disabled during the action

Scenario 10: Logout

1. Click **“Logout”** in the sidebar
2. Verify you’re redirected to the login page
3. Verify the API key is cleared from localStorage
4. Try to navigate to a protected page (e.g., `/clients`)
5. Verify you’re redirected back to login

API Endpoint Testing

You can also test the API directly:

Health Check

```
curl http://localhost:3000/health
```

List Clients

```
curl -H "X-API-Key: lgx_Wjvg8Dlde2BXzAbQ4nBnqQnGLhCVJkBuXMit2LqUsrs" \
  http://localhost:3000/v1/clients
```

Create Client

```
curl -X POST http://localhost:3000/v1/clients \
  -H "Content-Type: application/json" \
  -H "X-API-Key: lgx_Wjvg8Dlde2BXzAbQ4nBnqQnGLhCVJkBuXMit2LqUsrs" \
  -d '{
    "name": "Test Client",
    "industry": "Technology",
    "website": "https://example.com"
  }'
```

List Campaigns

```
curl -H "X-API-Key: lgx_Wjvg8Dlde2BXzAbQ4nBnqQnGLhCVJkBuXMit2LqUsrs" \
  http://localhost:3000/v1/campaigns
```

Troubleshooting

Dashboard won’t start

- Ensure Node.js 18+ is installed: `node --version`
- Delete `node_modules` and reinstall: `npm install`
- Check if port 3001 is already in use: `lsof -i :3001`

Backend won’t start

- Check DATABASE_URL is set correctly
- Verify PostgreSQL is running
- Check if port 3000 is already in use: `lsof -i :3000`
- View logs: `tail -f /tmp/backend.log`

API key not working

- Ensure the key starts with `lgx_`
- Verify the key is active in the database
- Check the organization_id matches
- Try creating a new API key

Dashboard shows empty pages

- Open browser DevTools (F12)
- Check Console for errors
- Check Network tab for failed API requests

- Verify NEXT_PUBLIC_API_BASE_URL is set correctly in `.env.local`

Leads not appearing after campaign run

- Check backend logs for errors: `tail -f /tmp/backend.log`
- Verify Google Places API key is configured (if using)
- Check campaign configuration (targeting might be too specific)
- Try running the campaign again

Performance Testing

Large Dataset Test

1. Create 50+ clients
2. Create 20+ campaigns
3. Run campaigns to generate 500+ leads
4. Test:
   - Page load times
   - Search performance
   - Filter responsiveness
   - Scrolling smoothness

Concurrent Users

1. Open dashboard in 3 different browsers
2. Login with different API keys
3. Perform actions simultaneously
4. Verify no race conditions or conflicts

Security Testing

Authentication

1. Try accessing `/clients` without logging in
2. Verify redirect to login page
3. Try using an expired/invalid API key
4. Verify error handling

XSS Protection

1. Try entering malicious scripts in:
   - Client name: `<script>alert('XSS')</script>`
   - Campaign name: `<img src=x onerror=alert('XSS')>`
2. Verify scripts are not executed (should be escaped)

CSRF Protection

1. API uses API keys (not cookies for API auth)
2. Human auth uses HttpOnly cookies
3. Verify CORS is configured properly on backend

Browser Compatibility

Test on:
- ✅ Chrome (latest)
- ✅ Firefox (latest)
- ✅ Safari (latest)
- ✅ Edge (latest)

Accessibility Testing

1. **Keyboard Navigation**:
   - Tab through all interactive elements
   - Verify focus indicators are visible
   - Test form submission with Enter key

2. **Screen Reader** (optional):
   - Use VoiceOver (Mac) or NVDA (Windows)
   - Verify labels and descriptions are read correctly

3. **Color Contrast**:
   - Use browser DevTools to check contrast ratios
   - Verify all text meets WCAG AA standards

Success Criteria

The dashboard is working correctly if:
- ✅ All pages load without errors
- ✅ Authentication works (login/logout)
- ✅ CRUD operations work for clients
- ✅ CRUD operations work for campaigns
- ✅ Campaign execution completes successfully
- ✅ Leads are displayed and filterable
- ✅ No console errors in browser DevTools
- ✅ Responsive design works on all screen sizes
- ✅ Loading states show during async operations
- ✅ Error messages display when actions fail

Report Issues

If you encounter any bugs:
1. Note the exact steps to reproduce
2. Check browser console for errors
3. Check backend logs: `tail -f /tmp/backend.log`
4. Document the expected vs actual behavior
5. Include screenshots if applicable