

# Socket.IO Smoke Test Documentation

**Generated by:** DeepAgent on 2025-11-13

**Task:** T7 - Realtime Socket Smoke Test with JWT Authentication



## Overview

The Socket.IO smoke test suite provides comprehensive validation of the real-time communication infrastructure. It tests both the main namespace ( / ) and community namespaces ( /community/:id ) with JWT authentication.

## What It Validates

### Authentication Flow

- JWT token validation
- Rejection of unauthenticated connections
- Token extraction from auth headers

### Main Namespace ( / )

- Basic connection establishment
- Ping/pong health checks
- Presence update events
- User room joining

### Community Namespace ( /community/:id )

- Dynamic namespace connections
- Community join/leave events
- Message broadcasting
- Typing indicators
- Multi-user interactions

### Error Handling

- Graceful timeout handling
- Connection error reporting
- Event validation

## Quick Start

### Prerequisites

#### 1. Socket.IO server must be running:

```
bash
cd apps/socket
pnpm dev
```

#### 2. Environment variables (optional):

```
bash
```

```

export JWT_SECRET="your-secret-key"
export SOCKET_URL="http://localhost:4001"
export TEST_COMMUNITY="test-community"

```

## Running the Smoke Test

```

# From the monorepo root
cd apps/socket

# Run smoke test (default settings)
pnpm smoke-test

# Run with verbose output
pnpm smoke-test:verbose

# Run with custom URL
pnpm smoke-test --url=http://localhost:4001

# Run with custom community
pnpm smoke-test --community=my-community

# Combine options
pnpm smoke-test --url=http://localhost:4001 --community=sf-music --verbose

```



## Test Suite Details

### Test 1: Main Namespace Connection

**Purpose:** Verify basic Socket.IO connection with JWT authentication

**Steps:**

1. Generate valid JWT token
2. Connect to main namespace ( / )
3. Verify successful connection
4. Disconnect cleanly

**Expected Result:** Connection established within timeout period

**Failure Scenarios:**

- Server not running
- Invalid JWT secret mismatch
- Network issues

### Test 2: Authentication Rejection (No Token)

**Purpose:** Verify server properly rejects unauthenticated connections

**Steps:**

1. Attempt connection without JWT token
2. Expect connection rejection

**Expected Result:** `connect_error` with "Authentication token required" message

**Failure Scenarios:**

- Server accepts connection without token (security issue!)
  - Wrong error message
- 

**Test 3: Ping/Pong Health Check****Purpose:** Verify basic bidirectional communication**Steps:**

1. Connect with valid token
2. Emit `ping` event
3. Wait for `pong` response

**Expected Result:** Pong received with timestamp**Failure Scenarios:**

- Pong not received (handler not working)
  - Timeout on slow connections
- 

**Test 4: Presence Update Event****Purpose:** Verify user presence tracking works**Steps:**

1. Connect to main namespace
2. Emit `presence:update` with status “online”
3. Wait for `presence:changed` broadcast

**Expected Result:** Presence change broadcast received**Failure Scenarios:**

- Event handler not registered
  - Broadcast not reaching clients
- 

**Test 5: Community Namespace Connection****Purpose:** Verify dynamic namespace creation and connection**Steps:**

1. Generate JWT token
2. Connect to `/community/:communityId` namespace
3. Verify successful connection

**Expected Result:** Connection to community namespace succeeds**Failure Scenarios:**

- Namespace not properly configured
  - Authentication fails on namespace
-

## Test 6: Community Join Event

**Purpose:** Verify explicit community join functionality

**Steps:**

1. Connect to community namespace
2. Emit `join-community` event with community ID and location
3. Wait for `join-community:success` acknowledgment

**Expected Result:** Success response with member count

**Failure Scenarios:**

- Join handler not working
  - Room not properly joined
- 

## Test 7: Community Message Event

**Purpose:** Verify message broadcasting in communities

**Steps:**

1. Connect to community namespace
2. Emit `community-message` with test content
3. Wait for `community-message:new` broadcast
4. Validate message content and user ID

**Expected Result:** Message broadcast with correct content and sender info

**Failure Scenarios:**

- Message not broadcast
  - Message content corrupted
  - User data missing
- 

## Test 8: Typing Indicators

**Purpose:** Verify real-time typing indicators between multiple users

**Steps:**

1. Create two test users with JWT tokens
2. Connect both to same community namespace
3. User 1 emits `typing:start`
4. User 2 receives `typing:user` event

**Expected Result:** Typing event received by other users

**Failure Scenarios:**

- Event not received by other users
  - Typing state not properly tracked
-

## JWT Token Generation

The smoke test includes a token generator utility at `test/utils/generate-test-token.ts`.

### Using the Token Generator

```
import { generateTestToken } from './utils/generate-test-token';

// Generate token with default test user
const token = generateTestToken();

// Generate token with custom user data
const customToken = generateTestToken({
  userId: 'custom-user-123',
  email: 'custom@test.com',
  username: 'customuser',
  expiresIn: '1h',
});

// Generate expired token (for negative testing)
import { generateExpiredToken } from './utils/generate-test-token';
const expiredToken = generateExpiredToken();

// Generate token with wrong secret (for negative testing)
import { generateInvalidToken } from './utils/generate-test-token';
const invalidToken = generateInvalidToken();

// Generate multiple tokens for multi-user tests
import { generateMultipleTestTokens } from './utils/generate-test-token';
const tokens = generateMultipleTestTokens(5); // Creates 5 test users
```

### Manual Token Generation

If you need to generate a token manually for debugging:

```
# Using Node.js REPL
node -e "
const jwt = require('jsonwebtoken');
const token = jwt.sign({
  sub: 'test-user-123',
  email: 'test@example.com',
  username: 'testuser'
}, 'super-secret-key', { expiresIn: '7d' });
console.log(token);
"
```

# Understanding Test Output

## Success Output

### Socket.IO Smoke Test Suite

- Server URL: <http://localhost:4001>
- Test Community: test-community
- Timeout: 10000ms per test
  
- 💡 Main Namespace Connection
  - ✓ Connected successfully (234ms)
  
- 💡 Authentication Rejection (No Token)
  - ✓ Correctly rejected: Authentication token required
  
- 💡 Ping/Pong Health Check
  - ✓ Pong received (156ms)
  
- 💡 Presence Update Event
  - ✓ Presence updated (189ms)
  
- 💡 Community Namespace Connection (test-community)
  - ✓ Connected to community namespace (245ms)
  
- 💡 Community Join Event
  - ✓ Joined community successfully (198ms)
  
- 💡 Community Message Event
  - ✓ Message sent and received (212ms)
  
- 💡 Typing Indicators
  - ✓ Typing indicator received (334ms)

### Test Summary

 Results: 8/8 passed  
 Average duration: 221ms

✓ All tests passed!

## Failure Output

- 💡 Main Namespace Connection
  - ✗ Connection failed: `connect ECONNREFUSED`
  
- ✗ Failed tests (1):
  - Main Namespace Connection
    - Error: `connect ECONNREFUSED`

## Verbose Output

Use `--verbose` or `-v` flag for detailed logging:

```
pnpm smoke-test:verbose
```

This shows:

- Token generation details
- Connection URLs
- Socket IDs
- Event data payloads
- Timing information

## Configuration

### Environment Variables

Variable	Default	Description
SOCKET_URL	http://localhost:4001	Socket.IO server URL
TEST_COMMUNITY	test-community	Community ID for testing
JWT_SECRET	super-secret-key	JWT signing secret (must match server)
NODE_ENV	development	Environment mode

### Command Line Options

Option	Description	Example
--url=<url>	Override server URL	--url=http://localhost:4001
--community=<id>	Override test community	--community=sf-music
--verbose or -v	Enable verbose output	--verbose

### Timeout Configuration

Default timeout is 10 seconds per test. To change:

Edit `test/smoke-test-client.ts`:

```
const TIMEOUT_MS = 20000; // 20 seconds
```

## Troubleshooting

---

### Problem: “Connection timeout”

**Cause:** Server not running or unreachable

**Solution:**

```
# Start the socket server
cd apps/socket
pnpm dev

# Verify it's running
curl http://localhost:4001/socket.io/
```

---

### Problem: “Authentication token required”

**Cause:** JWT\_SECRET mismatch between client and server

**Solution:**

```
# Check server secret
cat apps/socket/.env | grep JWT_SECRET

# Make sure test uses same secret
export JWT_SECRET="your-secret-from-env"
pnpm smoke-test
```

---

### Problem: “Test timeout” on specific tests

**Cause:** Event handlers not properly registered or slow network

**Solutions:**

1. **Check server logs** for errors
2. **Increase timeout** in smoke-test-client.ts
3. **Run with verbose mode** to see what's happening:

```
bash
  pnpm smoke-test:verbose
```

---

### Problem: “Message content or userId mismatch”

**Cause:** Community namespace handlers not working correctly

**Solution:**

1. Check `src/namespaces/communities.ts` for proper event handling
2. Verify user data is attached to socket in auth middleware
3. Check server logs for errors

## Problem: “Socket 2 connection failed” (Typing Indicators test)

**Cause:** Multiple connections or namespace issues

**Solution:**

1. Ensure server supports multiple connections
  2. Check for connection limits
  3. Verify namespace middleware is not blocking
-



## CI/CD Integration

### GitHub Actions Example

```

name: Socket.IO Smoke Test

on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]

jobs:
  smoke-test:
    runs-on: ubuntu-latest

    services:
      postgres:
        image: postgis/postgis:15-3.3
        env:
          POSTGRES_PASSWORD: postgres
        options: >-
          --health-cmd pg_isready
          --health-interval 10s
          --health-timeout 5s
          --health-retries 5

    steps:
      - uses: actions/checkout@v3

      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '20'

      - name: Install pnpm
        run: npm install -g pnpm

      - name: Install dependencies
        run: pnpm install

      - name: Start Socket server
        run: |
          cd apps/socket
          pnpm dev &
          sleep 5
        env:
          JWT_SECRET: ${{ secrets.JWT_SECRET }}

      - name: Run smoke tests
        run: |
          cd apps/socket
          pnpm smoke-test

```



### Writing Custom Tests

You can extend the smoke test suite by adding new test functions:

```

async function testCustomFeature(): Promise<boolean> {
  const testName = 'My Custom Feature';
  logTest(testName);
  const startTime = Date.now();

  return new Promise<boolean>((resolve) => {
    const token = generateTestToken();
    const socket = io(SOCKET_URL, {
      auth: { token },
      transports: ['websocket'],
      reconnection: false,
    });

    const timeout = setTimeout(() => {
      socket.disconnect();
      const duration = Date.now() - startTime;
      logError('Test timeout');
      recordResult(testName, false, duration, 'Timeout');
      resolve(false);
    }, TIMEOUT_MS);

    socket.on('connect', () => {
      // Your test logic here
      socket.emit('my-event', { data: 'test' });
    });

    socket.on('my-response', (data) => {
      clearTimeout(timeout);
      const duration = Date.now() - startTime;
      logSuccess(`Feature works (${duration}ms)`);

      socket.disconnect();
      recordResult(testName, true, duration);
      resolve(true);
    });

    socket.on('connect_error', (error) => {
      clearTimeout(timeout);
      const duration = Date.now() - startTime;
      logError(`Connection failed: ${error.message}`);
      recordResult(testName, false, duration, error.message);
      resolve(false);
    });
  });
}

// Add to test array in runAllTests()
const tests = [
  // ... existing tests
  testCustomFeature,
];

```

## 🎯 Exit Codes

The smoke test script uses standard exit codes:

<b>Exit Code</b>	<b>Meaning</b>	<b>Description</b>
0	Success	All tests passed
1	Failure	One or more tests failed

Use in scripts:

```
pnpm smoke-test
if [ $? -eq 0 ]; then
  echo "✓ Smoke tests passed"
else
  echo "✗ Smoke tests failed"
  exit 1
fi
```

## Related Documentation

- [Socket.IO Server Documentation](#) (./src/README.md)
- [JWT Authentication](#) (./src/middleware/auth.ts)
- [Community Namespaces](#) (./src/namespaces/communities.ts)
- [Event Handlers](#) (./src/handlers/index.ts)
- [Logger Utility](#) (./src/utils/logger.ts)

## Contributing

When adding new Socket.IO features:

1. ✓ Add corresponding smoke test
2. ✓ Update this documentation
3. ✓ Test with `pnpm smoke-test:verbose`
4. ✓ Update integration tests if needed

## License

Part of the UPRISE NEXT project. See main README for license information.