# Authentication System - Complete Fix Summary

**Date:** November 26, 2025
**Project:** CDPA Lab - Clinical Diagnosis Procedure Analysis Lab
**Status:** ☑ COMPLETED

---

## Problem 1: Duplicate Endpoint Definitions

### What was wrong:

- Authentication endpoints defined in **both** `Program.cs` (minimal APIs) **and** `AuthController.cs` (controller-based)
- Caused route conflicts and confusion

### How we fixed it:

- Removed all minimal API endpoints from `Program.cs` (`/auth/register`, `/auth/login`)
- Kept only `AuthController.cs` for register and login
- Kept `/auth/me` as a minimal API in `Program.cs` but fixed it

---

## Problem 2: Route Prefix Mismatch

### What was wrong:

- `AuthController` had `[Route("api/auth")]` prefix
- Endpoints were actually at `/api/api/auth/register` (double `/api`)

### How we fixed it:

- Changed `[Route("api/auth")]` to `[Route("auth")]`
- Correct URLs: `/auth/register`, `/auth/login`, `/auth/me`

---

## Problem 3: Missing DTO Classes in Traditional Namespace

### What was wrong:

- `AuthDtos.cs` used **file-scoped namespace** (`namespace Api.Models;`)
- Caused accessibility issues with public classes
- C# compiler couldn't properly resolve the classes

### How we fixed it:

Converted to **traditional namespace braces**:

```
namespace Api.Models
{
    public class RegisterRequest { ... }
    public class LoginRequest { ... }
    public class AuthResponse { ... }
}
```

---

## Problem 4: Missing Service Registrations

### What was wrong:

- `JwtService` wasn't registered in the dependency injection container
- `AddControllers()` wasn't called in `Program.cs`

### How we fixed it:

```
builder.Services.AddControllers();           // Added
builder.Services.AddScoped<JwtService>();    // Added
```

---

## Problem 5: Claim Type Mismatch in Token Validation (MAIN ISSUE)

### What was wrong:

- `JwtService` creates claim using: `new Claim(JwtRegisteredClaimNames.Sub, user.Id.ToString())`
- But `/auth/me` tried to retrieve it using: `http.User.FindFirstValue(JwtRegisteredClaimNames.Sub)`
- These were **not matching** due to how ASP.NET maps claim types

### Debug output showed:

```
DEBUG: userIdClaim =
DEBUG: All claims =
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier=019ac16f-
d949-7f99-b73c-52a3c69df642
DEBUG: userIdClaim is empty!
```

### How we fixed it:

#### Changed from:

```
var userIdClaim = http.User.FindFirstValue(JwtRegisteredClaimNames.Sub);
```

To:

```
var userIdClaim = http.User.FindFirstValue(ClaimTypes.NameIdentifier);
```

This correctly extracts the user ID from the JWT token's `sub` claim.

---

## Problem 6: Port Conflicts

### What was wrong:

- Port 5172 was already in use by a previous API instance
- New API couldn't start

### How we fixed it:

- Killed all `dotnet` processes
- Started fresh API on port 5172

---

## The Complete Flow (After Fixes)

### 1. REGISTER

```
POST /auth/register
{ email, password, fullName, role }
↓
✓ HashPassword (PBKDF2 with 100k iterations)
✓ Create AppUser
✓ Save to database
✓ Generate JWT token
✓ Return 200 + token
```

### 2. LOGIN

```
POST /auth/login
{ email, password }
↓
✓ Find user by email
✓ VerifyPassword (compare PBKDF2 hashes)
✓ Generate JWT token
✓ Return 200 + token
```

### 3. PROTECTED ENDPOINT (/auth/me)

```
GET /auth/me
Authorization: Bearer eyJhbGci...
↓
✓ Validate JWT signature
✓ Check expiration
✓ Extract ClaimTypes.NameIdentifier (user ID)
✓ Query database for user
✓ Return 200 + user data
```

---

## Key Files Modified

| File | Change |
|---|---|
| `AuthController.cs` | Changed route from `api/auth` to `auth` |
| `Program.cs` | Added `AddControllers()`, `AddScoped<JwtService>()`, fixed `/auth/me` claim extraction |
| `AuthDtos.cs` | Converted file-scoped namespace to traditional braces |
| `JwtService.cs` | Converted file-scoped namespace to traditional braces |
| `PasswordService.cs` | Converted file-scoped namespace to traditional braces |

---

## Root Cause Analysis

The **core issue** was the **claim type mismatch**:

- JWT standard uses `"sub"` (subject) claim for user ID
- ASP.NET maps this to claim type: `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier`
- Looking it up with `JwtRegisteredClaimNames.Sub` didn't work
- Looking it up with `ClaimTypes.NameIdentifier` works correctly

**This is a common .NET JWT authentication gotcha!**

---

## Verification Results

✅ Register endpoint working (200 OK)
✅ Login endpoint working (200 OK)
✅ Token generation working
✅ Protected `/auth/me` endpoint working (200 OK)
✅ Authentication validation working

---

## Technology Stack Used

- **Framework:** ASP.NET Core 9.0

- **Database:** PostgreSQL
- **Authentication:** JWT (JSON Web Tokens)
- **Password Hashing:** PBKDF2 (100,000 iterations)
- **API Testing:** Postman
- **Language:** C#

---

## Security Features Implemented

✓ Password hashing with PBKDF2 and 100k iterations
✓ Random salt (16 bytes) per password
✓ JWT tokens signed with HMAC-SHA256
✓ Fixed-time comparison for password verification
✓ Generic error messages (no account enumeration)
✓ Token expiration (2 hours)
✓ Claim validation

---

**End of Report**