



Issue Tracking System


Web Application





Project Abstract

Build a centralized, user-friendly system to raise tickets, assign priorities/types, track progress, collaborate via comments, and measure performance.



Technology Stack

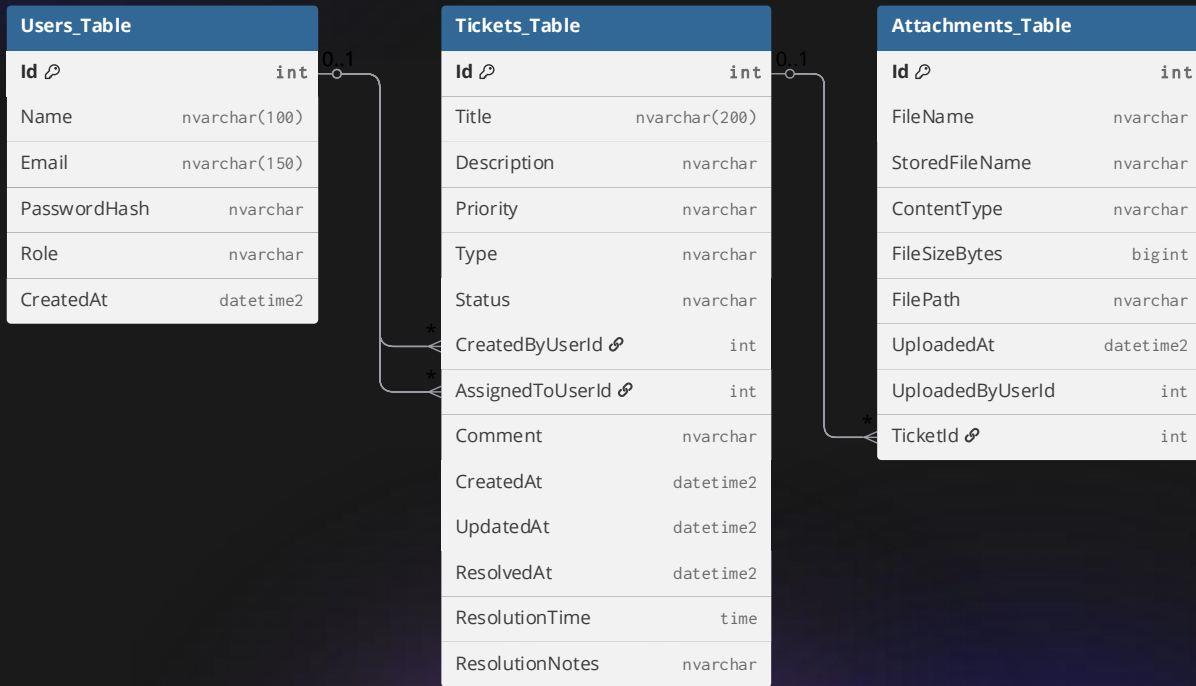


<u>Frontend</u>	Angular 20, TypeScript, Tailwind CSS
<u>Backend</u>	ASP.NET Core (.NET 8/9), Repository Pattern, EF Core, Middleware
<u>Database</u>	SSMS SQL Server with EF Core Migrations
<u>Auth</u>	JWT-based authentication, role-based authorization
<u>Tools</u>	NPM, dotnet CLI, Docker (optional), Postman





ER Diagram



Key Features



Authentication & Roles

JWT login/registration, roles (User, Admin and Representative) with guarded routes and API authorization.

User Management

Create, read, update, delete users; profile/password update; representative listing.

Ticket Management

Create ticket with priority/type (software/hardware)
Search/filter/sort; pagination
Update status, assign/reassign, add comments.
Upload Attachments and download per ticket.



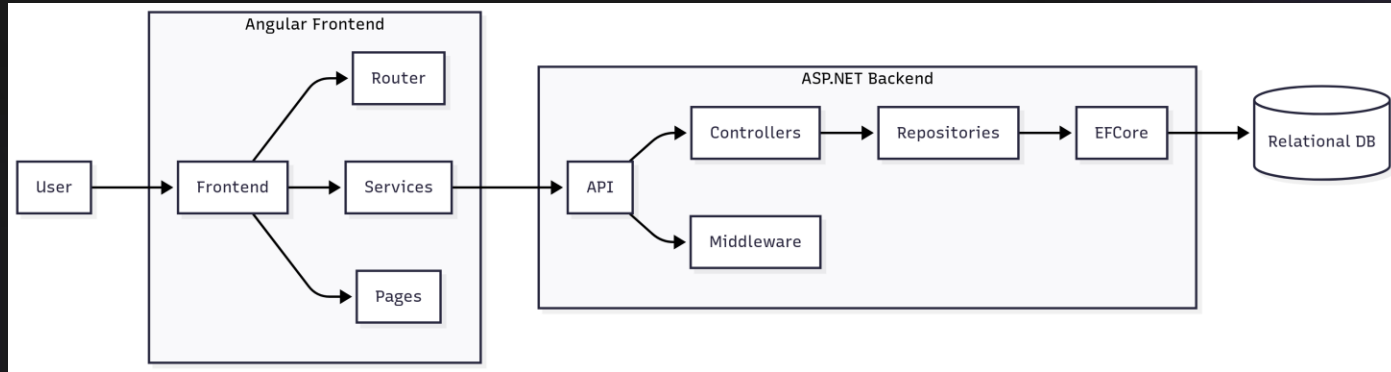


Testing Approach

- NUnit unit tests targeting repository layer (TicketRepo, UserRepo).
- Each test uses a fresh EF Core InMemoryDatabase keyed by a new Guid to avoid cross-test data bleed.
- User CRUD correctness and business rules (deletion constraints).
- Retrieves the correct user among multiple records
- Name, password hash, and role updates are saved and returned.
- Returns true when user row is removed with no tickets.



Overall Workflow





Project Conclusion

- Problem: Performance Insights and File Uploads.
- Modules: Auth, Users, Tickets, Attachments, Dashboard, KPIs, Error Handling each designed with DTOs, controllers, repositories, and guarded routes.
- Future Improvements: Containerizing the application with scalable micro services.

