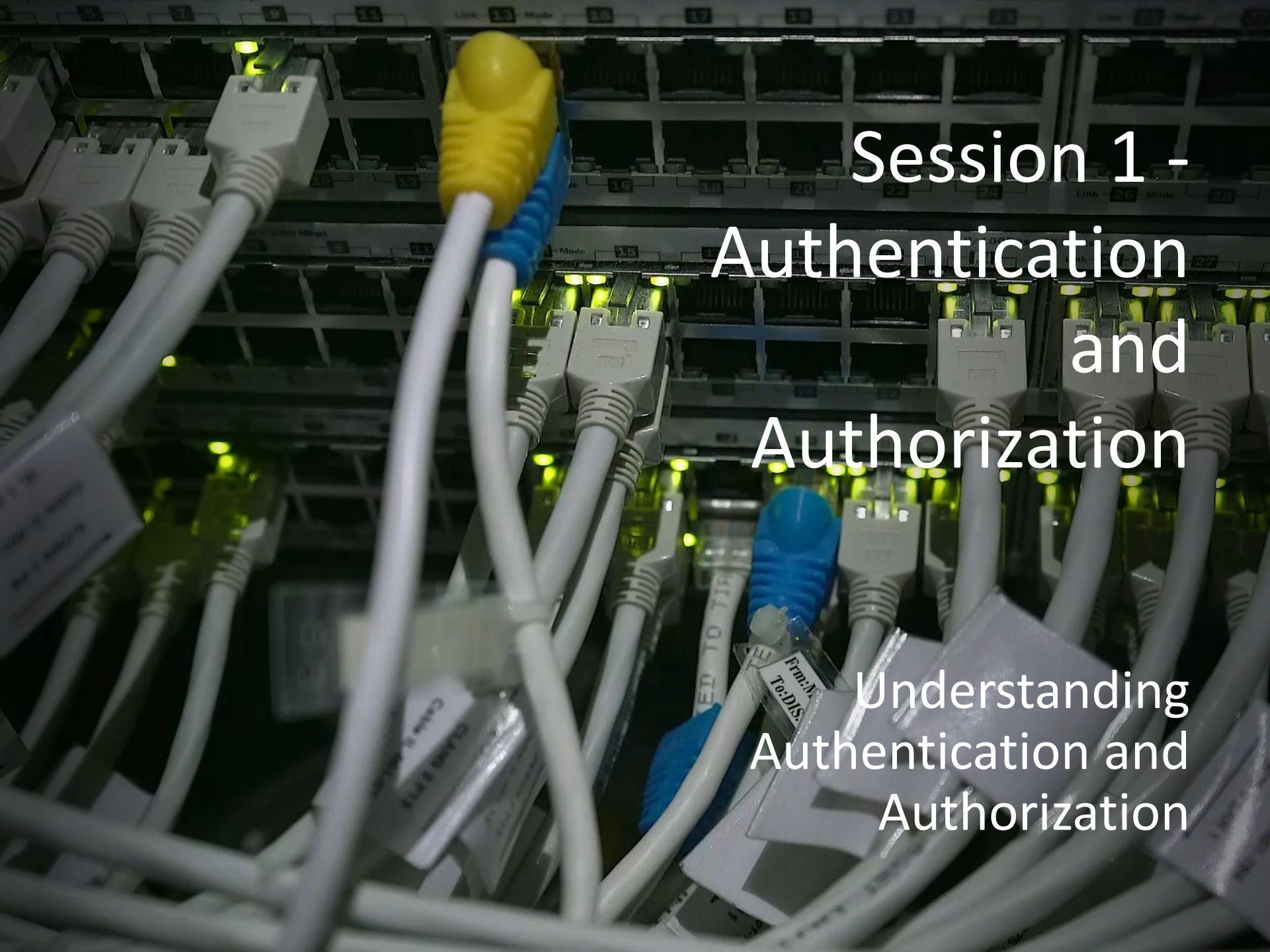


Session 1 - Authentication and Authorization

Understanding Authentication and Authorization

Blazor Authentication

- Blazor, Microsoft's web framework for building interactive UIs with C#, offers robust authentication capabilities essential for securing modern web applications. This summary outlines key aspects of Blazor authentication, its implementation, and its benefits for your organization.

Integration with ASP.NET Core Identity

- Leverages the proven ASP.NET Core Identity system
- Provides comprehensive user management:
  - Registration
  - Login
  - Password reset

Flexible Authentication Options

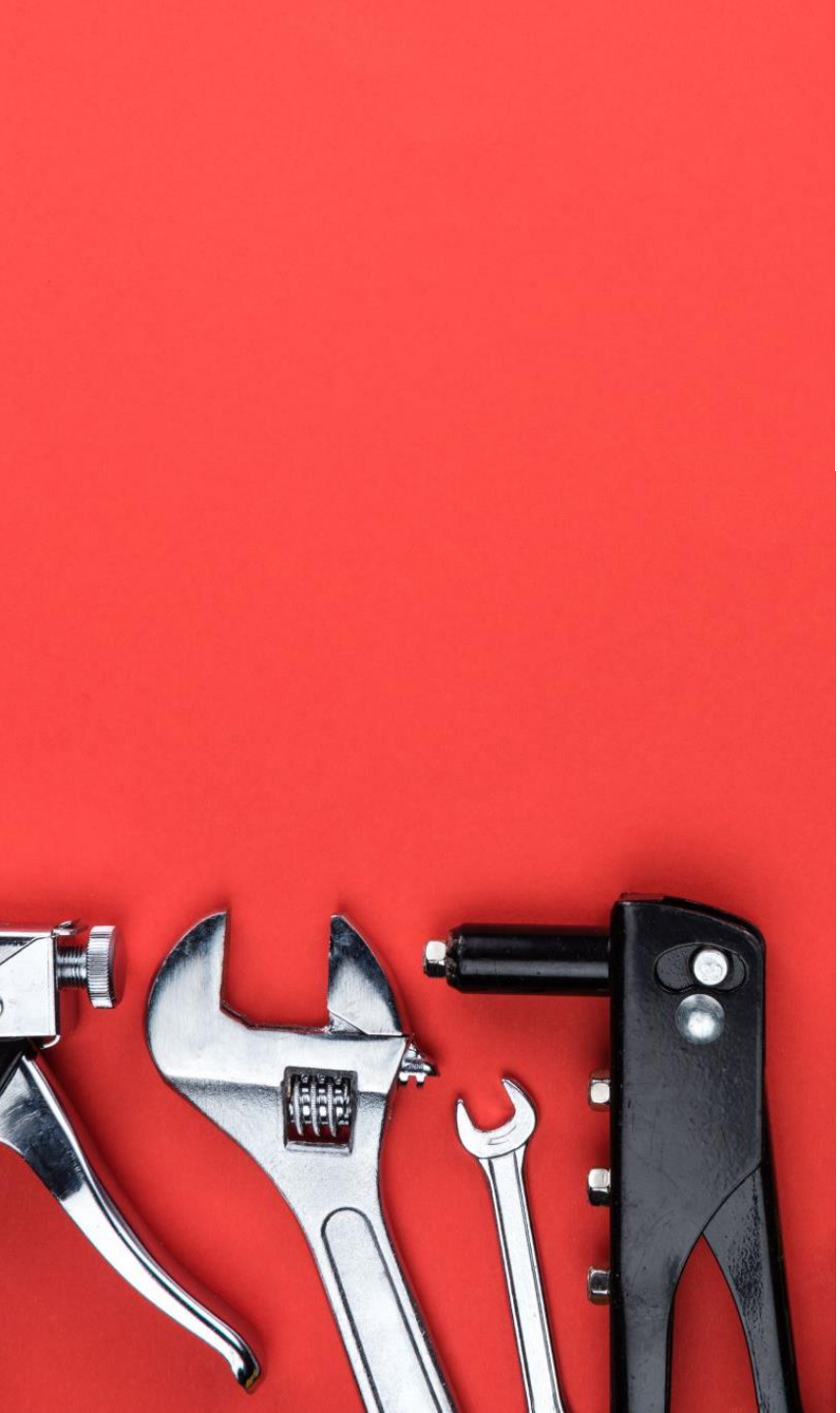
- Supports various authentication providers:
  - Local accounts
  - Microsoft
  - Google
  - Facebook
- Enables Single Sign-On (SSO) capabilities

Client-Side and Server-Side Authentication

- Seamless authentication in both Blazor Server and Blazor WebAssembly apps
- Secure token-based authentication for client-side Blazor WebAssembly

Role-Based and Policy-Based Authorization

- Fine-grained access control using roles and custom policies
- Easy implementation of complex authorization rules



Built-in Components

- <AuthorizeView> for conditional UI rendering based on authentication state
- [Authorize] attribute for securing components and pages

Take Home Reading

- Please review the following slides for a high-level overview of authentication for further information.

Introduction

OVERVIEW OF AUTHENTICATION AND AUTHORIZATION

IMPORTANCE IN ASP.NET CORE APPLICATIONS

What is Authentication?

DEFINITION: VERIFYING THE IDENTITY OF A USER

ANALOGY: CHECKING INTO A HOTEL WITH ID

EXAMPLE: RECEPTIONIST ASKS FOR ID TO VERIFY IDENTITY

Claims-Based Authentication

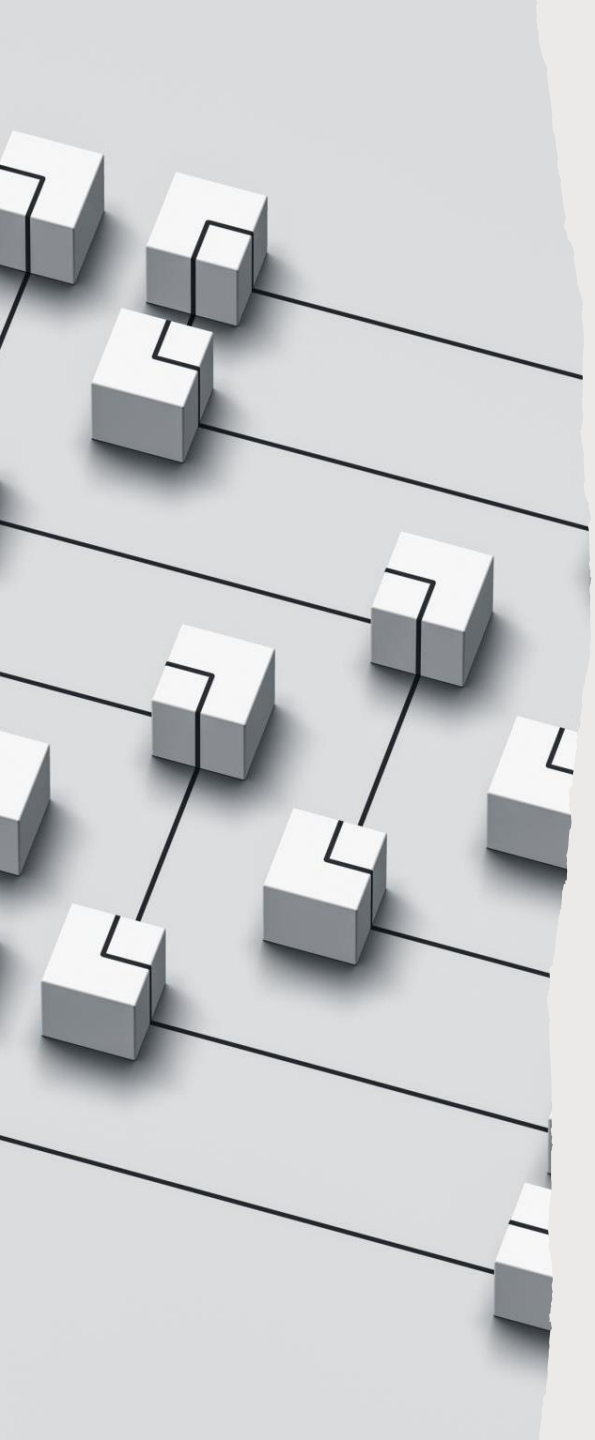
- **Explanation of Identity Claims**
  - Identity claims are pieces of information about a user that are used to verify their identity. These claims can include personal details such as name, address, and birth date. In the context of authentication, claims are used to ensure that the person requesting access is who they say they are.
- **Importance of Claims in Proving Identity**
  - Claims play a crucial role in authentication because they provide verifiable information that can be checked against known data. For example, when a user logs in, their password is checked against the stored password for that username. If the password matches, the identity claims associated with that username (such as name and email) are used to confirm the user's identity.
- **Real-World Analogy: Passport with Multiple Identity Claims**
  - A passport is a real-world example of a document that contains multiple identity claims. It includes your name, photograph, date of birth, nationality, and other details that together provide a comprehensive proof of your identity. When you travel, border control checks these claims to verify that you are the rightful holder of the passport.

Identity Cookies

- **How Identity Cookies Work**
  - Identity cookies are small pieces of data stored on the client's browser to identify a user between requests.
  - They contain encrypted information about the user's identity, which the server can read and verify.
  - When a user logs in, the server creates an identity cookie and sends it to the client's browser.
  - The browser includes this cookie in subsequent requests to the server, allowing the server to recognize the user.

Role of Identity Cookies in ASP.NET Core

- In ASP.NET Core, identity cookies are integral to the authentication mechanism.
- They facilitate persistent user sessions, meaning users do not need to log in again on each request.
- Identity cookies store information like user ID, roles, and claims, which are used for authentication and authorization.
- ASP.NET Core Identity handles the creation, management, and security of these cookies.



ASP.NET Core Identity Framework

- Features of ASP.NET Core Identity
- Password management
- Two-factor authentication
- Built-in user interface (login screen)
- Extending and customizing the identity framework

Features of ASP.NET Core Identity

- **Comprehensive User Management**: Supports CRUD (Create, Read, Update, Delete) operations for user accounts.
- **Role Management**: Enables defining and managing roles to control user permissions.
- **Claims-Based Authentication**: Utilizes claims to represent user information and manage access control.
- **Security Features**: Includes password hashing, account lockout, and token-based security mechanisms.
- **Scalability**: Compatible with various data stores like SQL Server, MySQL, PostgreSQL, and more

Password Management

- **Secure Storage**: Passwords are hashed using secure algorithms (e.g., PBKDF2, bcrypt, Argon2) before storage.
- **Password Policies**: Enforces rules such as minimum length, complexity, and expiration.
- **Password Reset**: Allows users to reset forgotten passwords through email verification.
- **Password History**: Tracks past passwords to prevent reuse of recent ones.

Two-Factor Authentication

- **Enhanced Security**: Adds an extra layer of security by requiring a second form of verification (e.g., SMS, email, authenticator apps).
- **User-Friendly Setup**: Provides a simple setup process for enabling two-factor authentication.
- **Backup Codes**: Generates backup codes for use if users lose access to their primary two-factor method.
- **Integration with External Providers**: Supports seamless integration with external two-factor authentication providers.

Built-In User Interface i.e. Login Screen

- **Out-of-the-Box UI**: Includes pre-built screens for login, registration, and password recovery.
- **Customizable Templates**: Can be tailored to match the application's branding and design.
- **Localization Support**: Allows for localization to support different languages and regions.
- **Responsive Design**: Ensures the UI works well on desktops, tablets, and smartphones.

Extending and Customizing the Identity Framework

- **Custom User Properties**: Extend the identity model to include additional user properties.
- **Custom Stores**: Implement custom storage providers, such as NoSQL databases or cloud storage.
- **Custom Token Providers**: Create custom token providers for tasks like password reset and email confirmation.
- **Middleware Integration**: Integrate with other middleware to enhance authentication and authorization.
- **Event Handling**: Hook into identity events (e.g., user creation, password change) to trigger custom actions.

Beyond Basic Authentication

- Using identity providers
- OpenID Connect and OAuth
- Centralizing authentication for multiple applications
- Benefits of using external identity providers

What is Authorization?

DEFINITION: CONTROLLING WHAT A USER CAN DO

ANALOGY: HOTEL KEY PROVIDING ACCESS TO SPECIFIC AREAS

EXAMPLE: ACCESS TO ROOMS, POOL, AND RESTRICTED AREAS

Understanding Authorization Controlling - What a User Can Do

- **Authorization** is determining what an authenticated user is allowed to do within an application.
- It involves setting permissions and access controls to resources based on the user's identity and roles.
- Unlike authentication, which verifies the identity of a user, authorization ensures that users have the appropriate permissions to access specific resources or perform certain actions.

Analogy: Hotel Key Providing Access to Specific Areas

- **Authorization**: Similar to a hotel key that grants access to specific areas within the hotel.
- **Personal Access**: The key allows you to access your room and other permitted areas like the pool or gym.
- **Restricted Access**: The key does not grant access to other guests' rooms or staff-only areas.

Real-World Application in Software

- **Role-Based Access Control (RBAC)**: Users are assigned roles, each with specific permissions.

- **Admin**:
  - Manage users
  - Configure settings
  - Access all areas of the application

- **Manager**:
  - Access reports
  - Manage teams
  - Cannot configure system settings

- **User**:
  - View and edit their own data
  - Cannot access administrative functions

Implementing Authorization
Direct and Indirect Authorization

- **Direct Authorization:**
- Authorization checks are performed directly within the application code.
- Typically involves checking user roles or claims explicitly in controllers or services.

- **Indirect Authorization:**
- Authorization decisions are offloaded to external services or APIs.
- The application sends user claims to an authorization service that returns permissions.

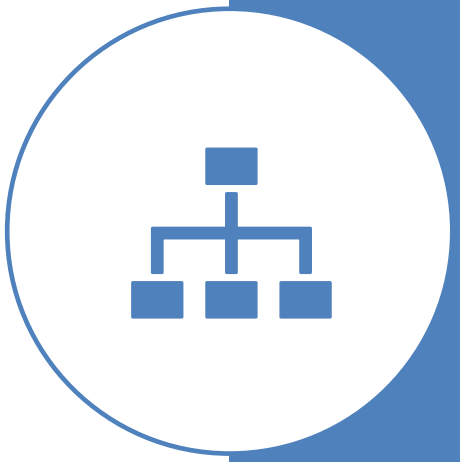
- **Role-Based Access Control (RBAC):**
- Assign roles to users and restrict access based on these roles.

Implementing Authorization
Using Claims to Obtain Authorization Data

- **Claims-Based Authorization:**
- Claims are information about the user, such as roles, departments, or other attributes.
- Claims are included in the user's identity token and can be used to make authorization decisions.

- **Custom Claims:**
- You can create custom claims to represent additional user information required for authorization.
- Adding custom claims during user authentication allows for more granular access control.

- **Claims-Based Authorization:**
- Use claims to enforce granular access control.



Implementing Authorization
Using Claims to Obtain Authorization Data

- **Policy-Based Authorization:**
- Policies define a set of requirements that must be met for authorization.
- Policies can be configured in the Startup.cs file and applied to controllers or actions.

- **Resource-Based Authorization:**
- Resource-based authorization checks if the user has access to specific resources.
- Often used when the authorization requirements depend on the resource being accessed.

- **Custom Authorization Policies:**
- Create custom policies to handle complex authorization requirements.

Statement Regarding Slide Accuracy and Potential Revisions

- Please note that the content of these PowerPoint slides is accurate to the best of my knowledge at the time of presentation. However, as new research and developments emerge, or to enhance the learning experience, these slides may be subject to updates or revisions in the future. I encourage you to stay engaged with the course materials and any announcements for the most current information