



# Module 2 Day 8

Database Security

# What makes an application?

- Program Data

- ✓ Variables & .NET Data Types
- ✓ Arrays
- ✓ More Collections (list, dictionary, stack, queue)
- ✓ Classes and objects (OOP)

- Program Logic

- ✓ Statements and expressions
- ✓ Conditional logic (if)
- ✓ Repeating logic (for, foreach, do, while)
- ✓ Methods (functions / procedures)
- ✓ Classes and objects (OOP)
- ❑ Frameworks (MVC)

- Input / Output

- User

- ✓ Console read / write
  - ❑ HTML / CSS
  - ❑ Front-end frameworks (HTML / CSS / JavaScript)

- Storage

- ✓ File I/O
  - ✓ Relational database
  - ❑ APIs

# Database Security

- Permissions

- Database users are granted access to resources (e.g., tables)
- Access can be fairly granular
  - Select, Insert, Update, Delete, Alter
- Set using DCL (Data Control Language)

GRANT SELECT, INSERT, UPDATE, DELETE ON employees TO smithj

- Stored Procedures

- Like the scripts we have been writing
- Compiled
- Can be Executed using SqlCommand

# Database Security Best Practices

- Avoid creating DB logins for individual users if possible
  - Create logins for an application instead
  - User logs on to the application (if needed)
  - Application logs on to the database
- Avoid storing DB credentials in plain text
  - Use encryption or a password vault
- Use stored procedures to access / update data
  - Grant users EXECUTE permission on the procedures
  - Grant NO permission directly on the tables

# SQL Injection

- A very common type of cyber-attack
- Allows a malicious user to read, update or delete data they should not have access to
- Caused by string concatenation in your program
- Prevented by using parameters
  - The bottom line: **Use Parameterized Queries!**
- <https://informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/>



Let's  
Code