

The background consists of a dense, dark blue hexagonal pattern, resembling a molecular lattice or a stylized architectural facade. The hexagons are slightly raised, creating a textured, geometric surface.

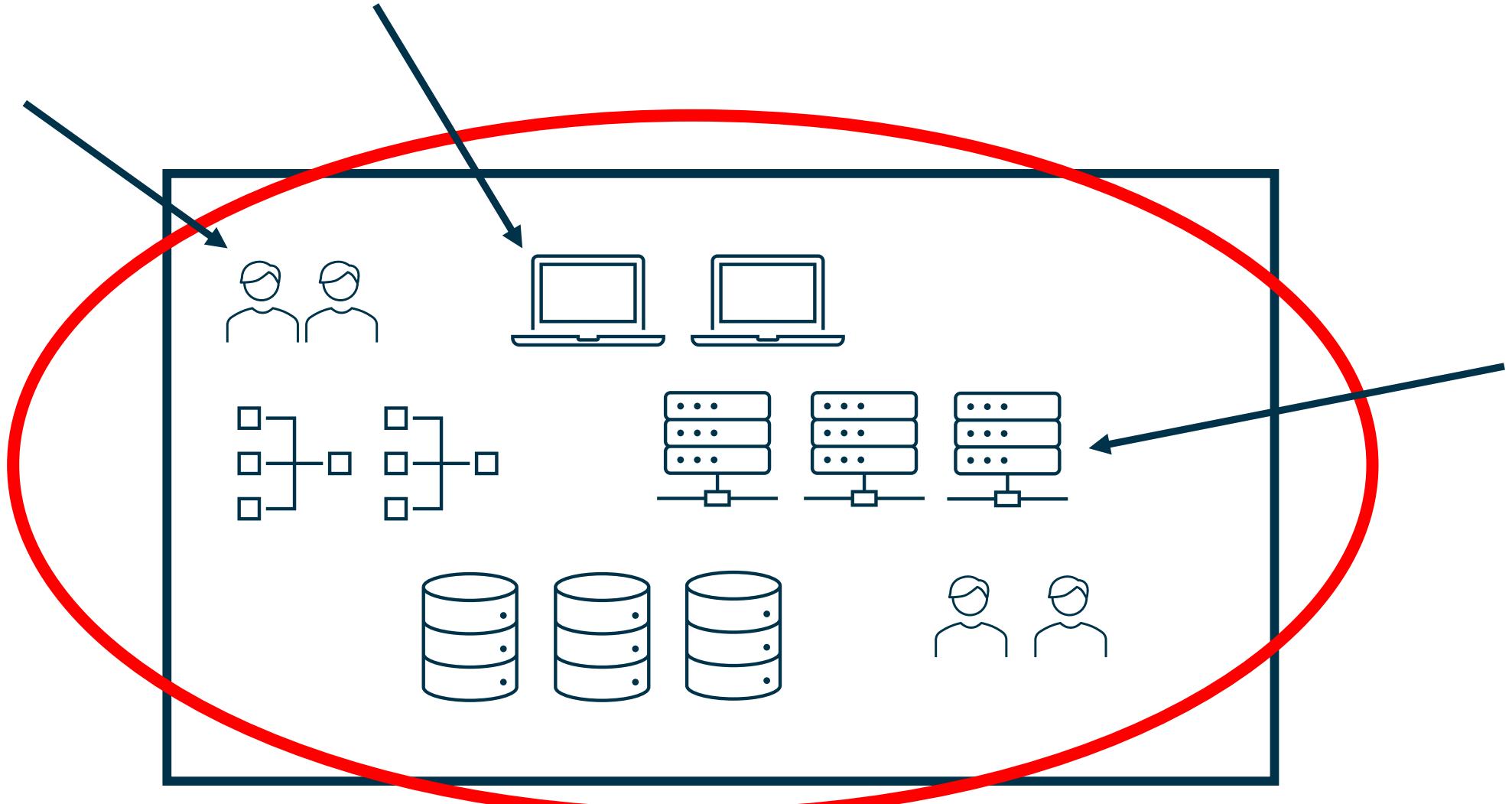
omega  
point.

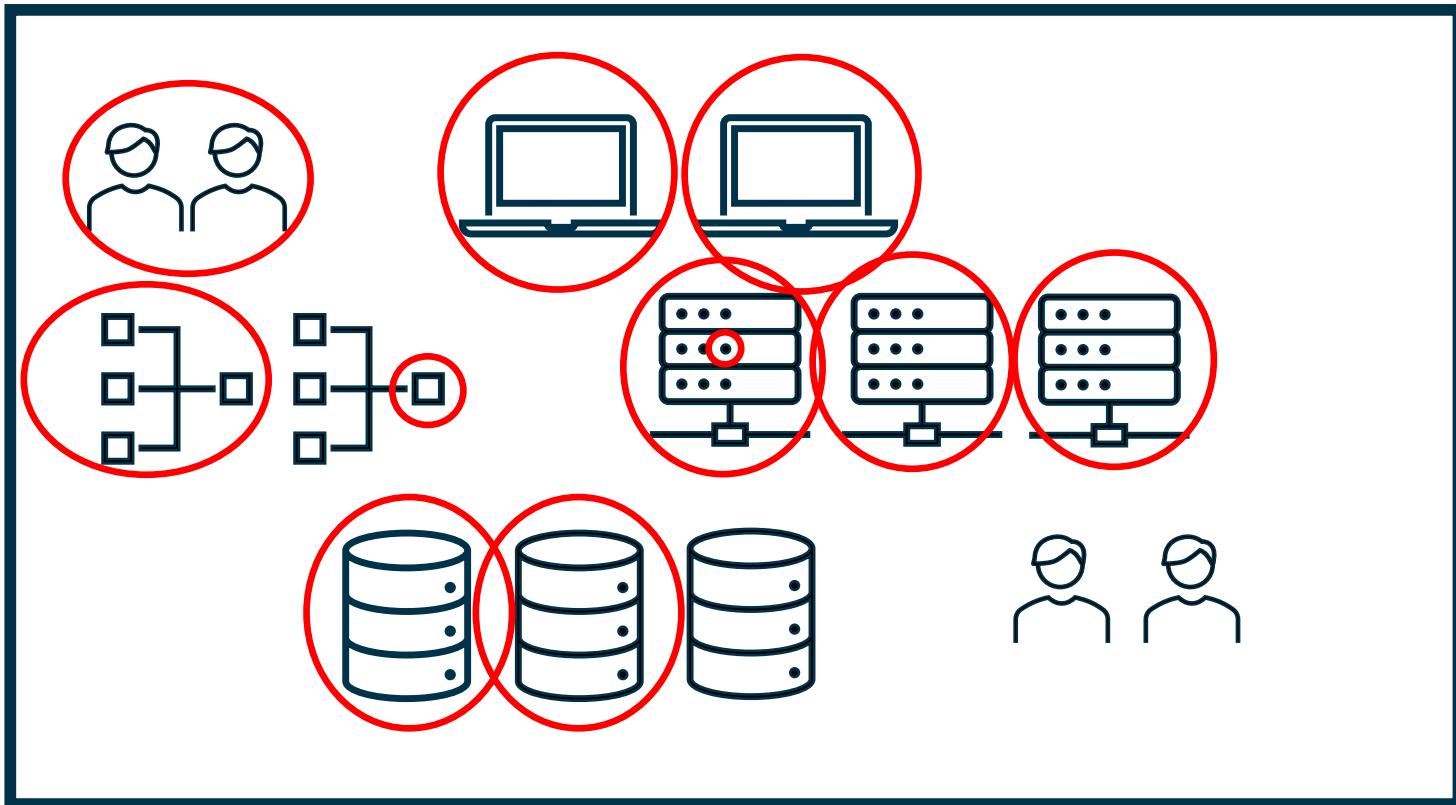




O.









«Secure By Design»

Domain driven security

Cybersecure digitalization

A portrait photograph of a young man with short brown hair, smiling at the camera. He is wearing a dark blue zip-up jacket with the white 'HH' logo on the chest. He is leaning against a wooden railing, with a blurred cityscape and lights visible in the background.

# Håkon Anders Strømsodd

- 25 år
- Sandefjord
- Datateknologi + AI (NTNU 2023)
- Sommerstudent (2022) og  
Konsulent (2023+) i Omegapoint Norge

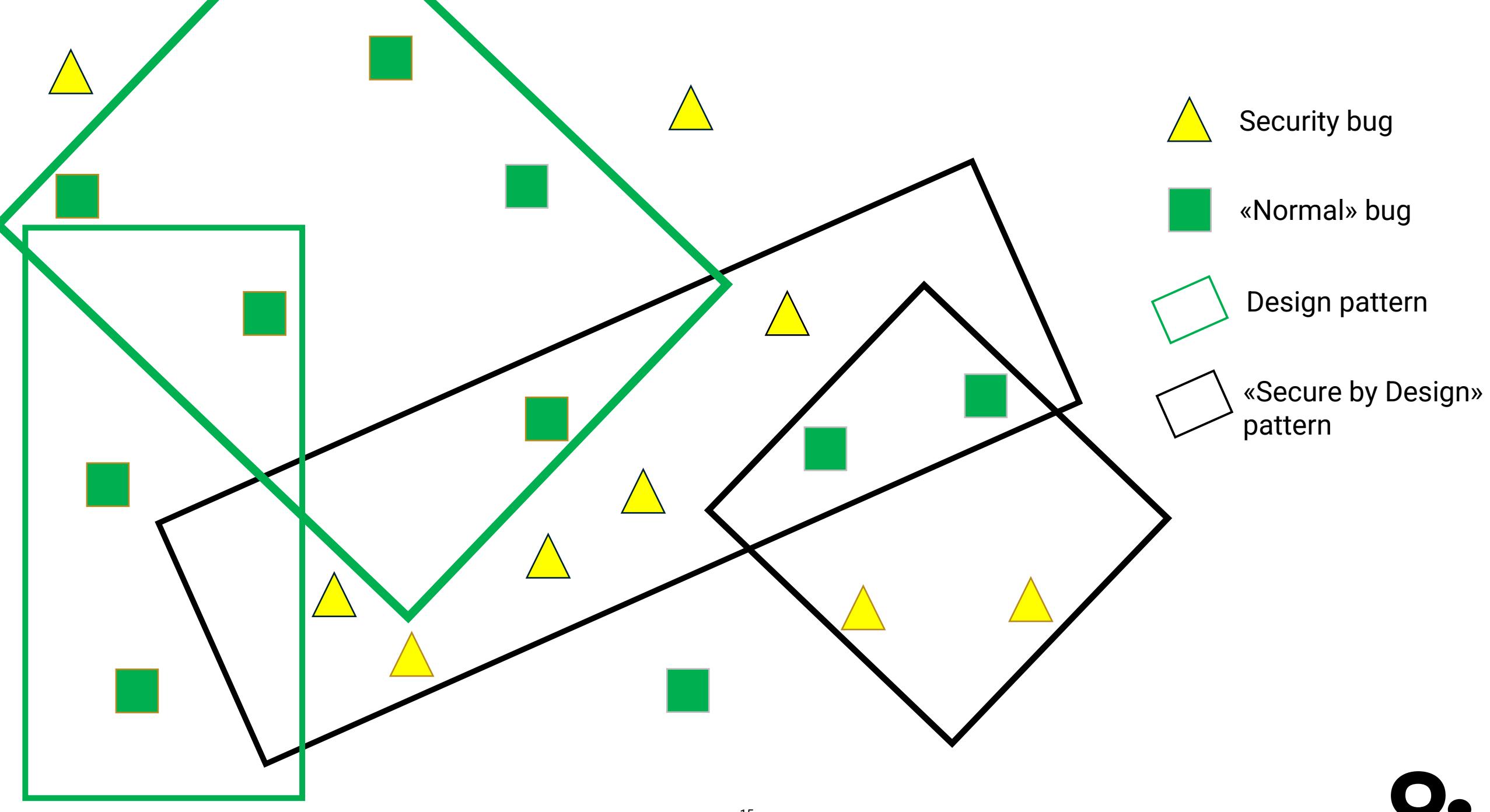
# Secure by Design

Dan Bergh Johnsson  
Daniel Deogun  
Daniel Sawano  
Foreword by Daniel Terhorst-North



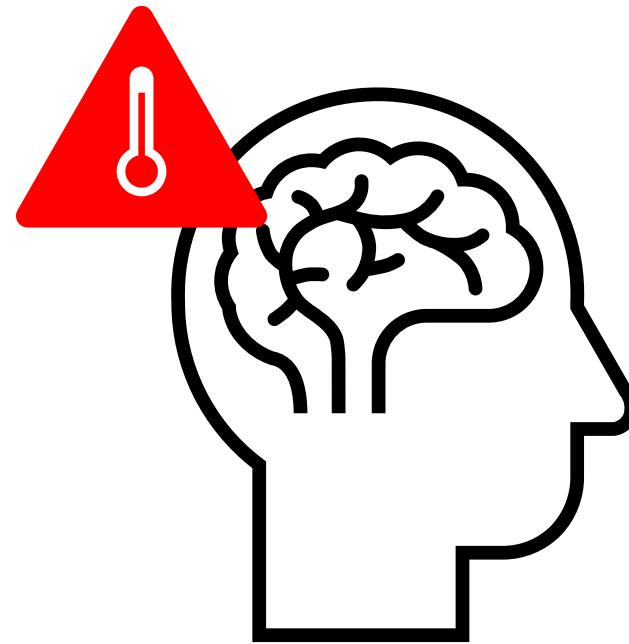
omega  
point.





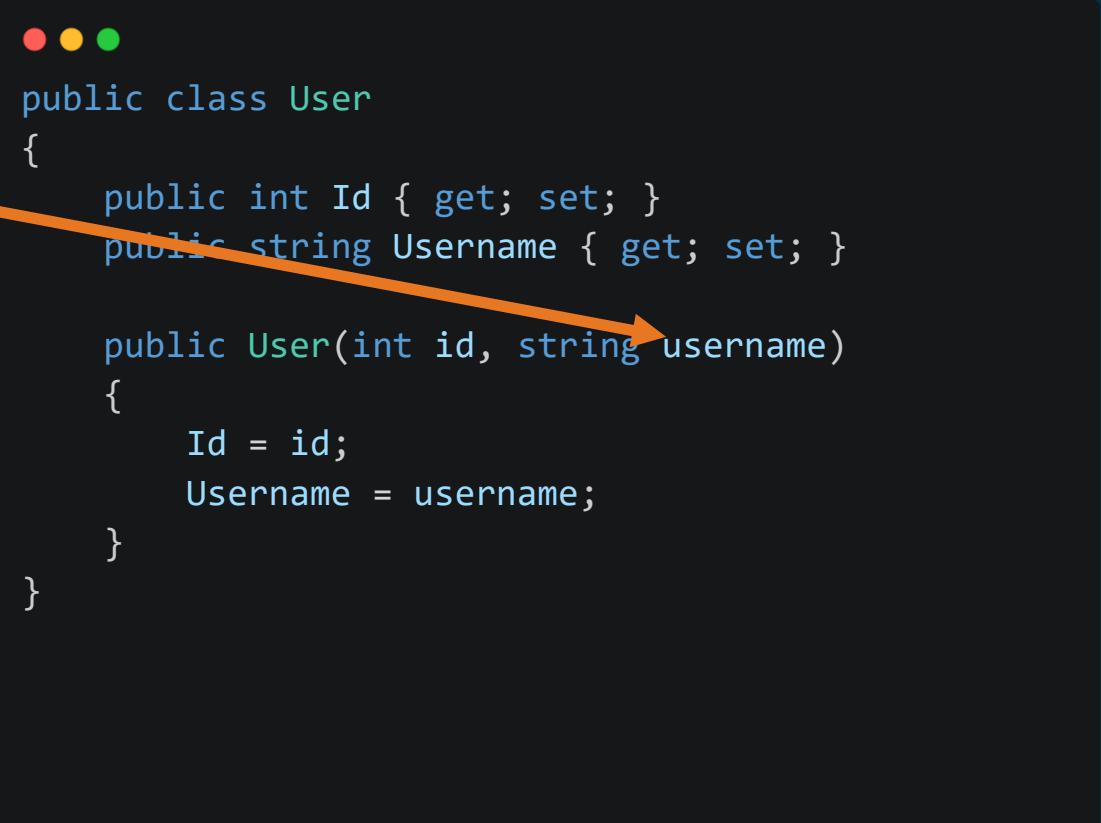
# “Traditional” approach

- ✓ OWASP
- ✓ XSS
- ✓ SQL Injection
- ✓ Attack vectors
- ✓ 0-day vulnerabilities
- ✓ ...



# “Traditional” approach

- Possible XSS and SQL injection
- <script>alert("Hacked");</script>



```
public class User
{
    public int Id { get; set; }
    public string Username { get; set; }

    public User(int id, string username)
    {
        Id = id;
        Username = username;
    }
}
```

# “Traditional” approach

```
●●●  
public class User  
{  
    public int Id { get; set; }  
    public string Username { get; set; }  
  
    public User(int id, string username)  
    {  
        Id = notNull(id);  
        Username = validateForXSSAndSQLI(username);  
    }  
}
```

# Secure by Design

```
●●●  
public class User  
{  
    private static readonly int USERNAME_MAX_LENGTH = 20;  
    private static readonly int USERNAME_MIN_LENGTH = 5;  
    private static readonly string USERNAME_ALLOWED_CHARS = "[a-zA-Z0-9_-]+";  
    private int Id { get; }  
    private string Username { get; }  
    public User(int id, string username)  
    {  
        var trimmedUsername = username.Trim();  
        lengthBetween(trimmedUsername, USERNAME_MAX_LENGTH, USERNAME_MIN_LENGTH);  
        matches(trimmedUsername, USERNAME_ALLOWED_CHARS);  
        Id = notNull(id);  
        Username = trimmedUsername;  
    }  
}
```



# Secure by Design

```
● ● ●  
public class Username  
{  
    private static readonly int USERNAME_MAX_LENGTH = 20;  
    private static readonly int USERNAME_MIN_LENGTH = 5;  
    private static readonly int USERNAME_ALLOWED_CHARS = "[a-zA-Z0-9_-]+";  
  
    public string Value { get; }  
  
    public Username(string username)  
    {  
        var trimmedUsername = username.Trim();  
        lengthBetween(trimmedUsername, USERNAME_MAX_LENGTH, USERNAME_MIN_LENGTH)  
        matches(trimmedUsername, USERNAME_ALLOWED_CHARS);  
        Value = trimmedUsername;  
    }  
}
```



# Secure by Design

```
•••  
public class User  
{  
    public int Id { get; }  
    public Username Username { get; }  
    public User(int id, Username username)  
    {  
        Id = notNull(id);  
        Username = username;  
    }  
}
```



# Postal Code





```
Person CreatePerson(string name, string address, int postalCode)
{
    var person = new Person(name, address + postalCode);
    db.Save(person);
    return person;
}
```

```
CreatePerson("Binneveien 13B", "Håkon Anders Strømsodd", 0774);
```



```
User CreatePerson(Name name, Address address, PostalCode postalCode)
{
    return new Person(name, address + postalCode);
}

CreatePerson(
    new Address("Binneveien 13B"),
    new Name("Håkon Anders Strømsodd"),
    new PostalCode(0484));
```



# Money!

```
●●●  
public class Money  
{  
    private static readonly string[] _allowedCurrencies =  
        new string[] {"NOK", "EUR", "USD", "SEK"};  
  
    public decimal Amount { get; }  
    public string Currency { get; }  
  
    public Money(decimal amount, string currency)  
    {  
        Amount = amount;  
        Currency = isOneOf(currency, _allowedCurrencies);  
    }  
}
```





```
public class Money
{
    //.....
    public static Money operator +(Money a, Money b)
    {
        if(a.Currency != b.Currency)
            throw new ArgumentException("Different currencies cannot be added");

        return new Money(a.Amount + b.Amount, a.Currency);
    }

    public static Money operator -(Money a, Money b)
    {
        if(a.Currency != b.Currency)
            throw new ArgumentException("Different currencies cannot be subed");

        return new Money(a.Amount - b.Amount, a.Currency);
    }
}
```



# Failures are not Exceptions

```
● ● ●  
public class Account  
{  
    //.....  
    public void Transfer(Money money, Account toAccount)  
    {  
        if (Balance() < money) {  
            throw new InsufficientFundsException();  
        }  
        //...  
  
        return;  
    }  
  
    Money Balance()  
    {  
        //...  
    }  
}
```

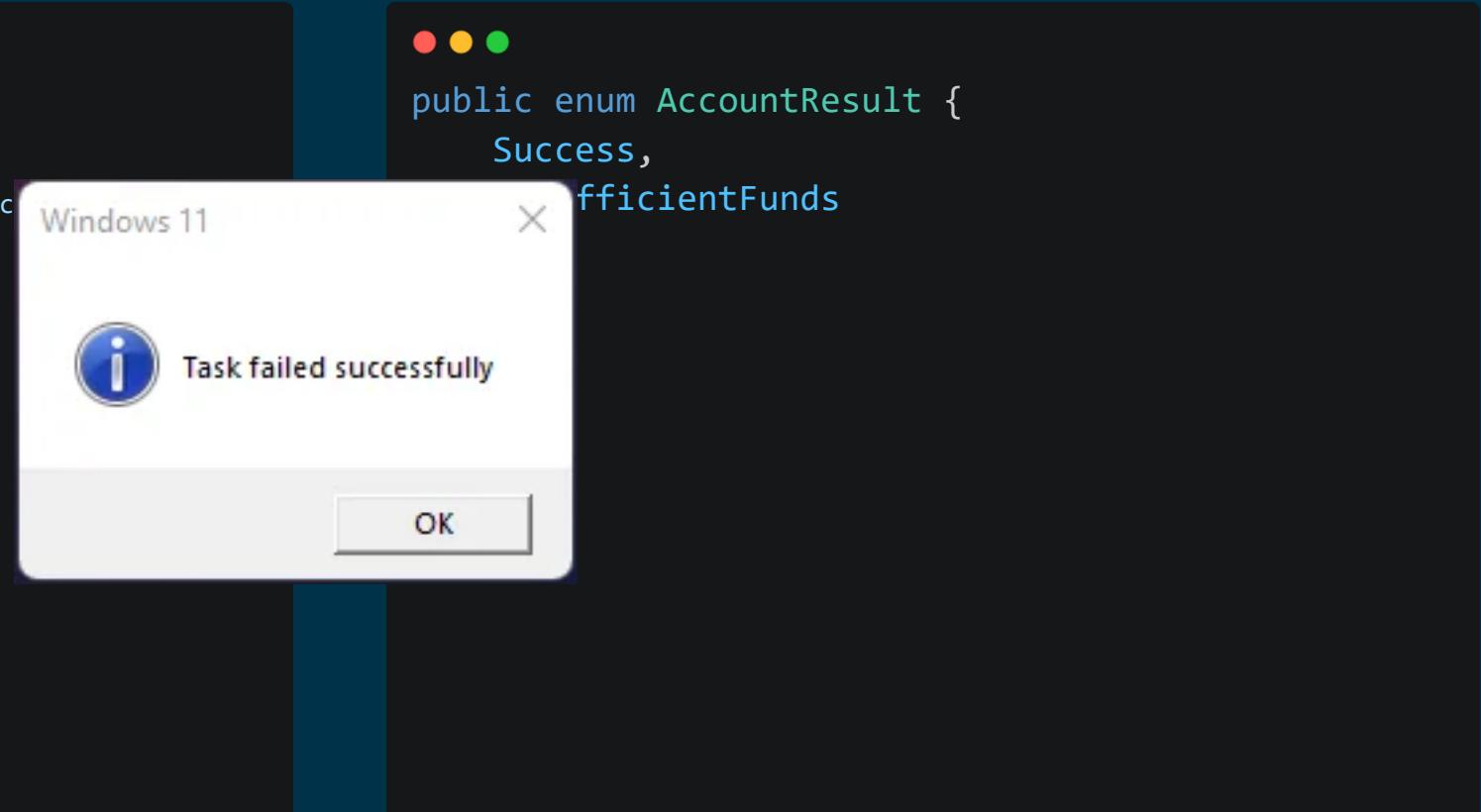


# Failures are not Exceptions

```
public class Account
{
    //....
    public Result Transfer(Money money, Account toAccount)
    {
        if (Balance() < money) {
            return AccountResult.InsufficientFunds;
        }
        //...

        return AccountResult.Success;
    }

    Money Balance()
    {
        //...
    }
}
```



# Handling permissions

- Don't do custom stuff everywhere
- Permissions are domain logic
- Three levels of authorization:
  - Functional level
  - Object level
  - Property level

```
●●●  
interface IProductPermissionService  
{  
    bool CanRead(Product product);  
    bool CanWrite(Product product);  
  
    //...  
}
```



# Secure by Design

Focus on good design

Less bugs

More secure systems

# Workshop Introduction

Secure by design



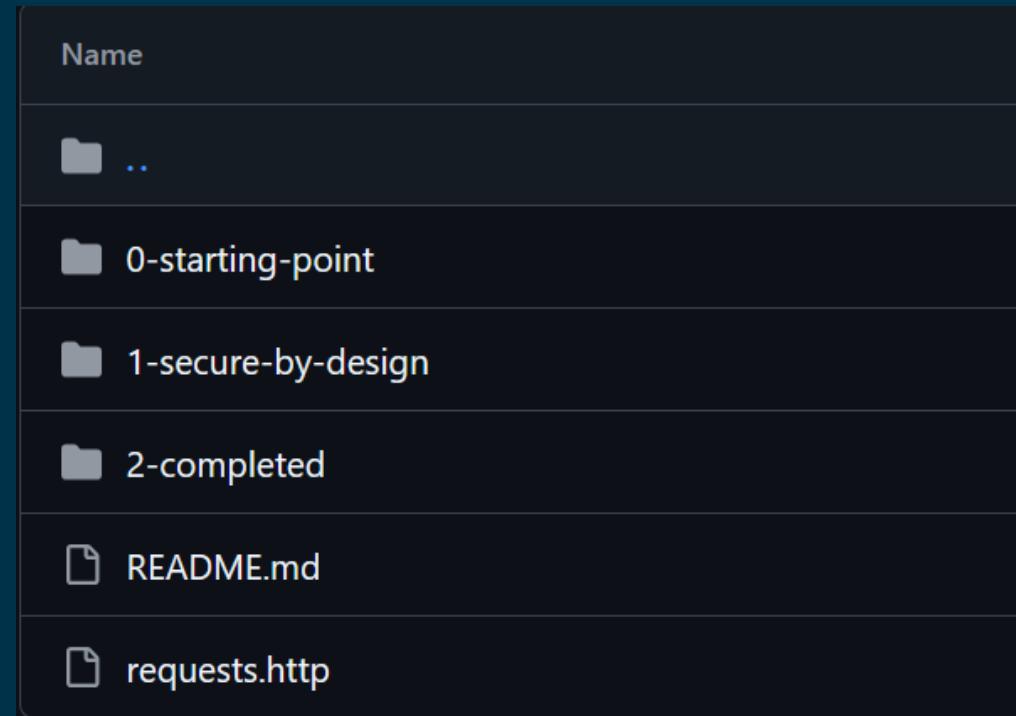
# Repository Structure

Contracts	Workshop
Resources	Workshop
SalesApi	Remove files that aren't needed
.gitignore	Workshop
OAuht2 and OpenID Connect best practices.pdf	Presentations
README.md	Remove files that aren't needed
SecureByDesign.sln	Remove files that aren't needed
Workshop introduction.pdf	Presentations
nuget.config	Minor improvements
secure-by-design-keynote.pdf	Updated slides



# Sales API

- Start with step 0 in the readme
- Start coding in folder 0
- No shame in having a peek at the spoilers



# bit.ly/op-secure-by-design

The screenshot shows a GitHub repository page for a project named "SecureByDesign". The repository has 1 branch and 0 tags. The main branch is "main". The repository contains several files and folders:

- .gitignore
- README.md
- SecureByDesign.sln
- Contracts
- Resources
- SalesApi
- SalesClient
- Tests

Most files and folders have a status of "Workshop", except for "Minor improvements" which appears twice.

A context menu is open over the repository URL "https://github.com/Omegapoint-Norge-Academy/secure-by-design". The menu is titled "Local" and includes the following options:

- Clone
- HTTPS
- SSH
- GitHub CLI

The "Clone" option is highlighted. A red box surrounds the "Clone" button and the URL field. Below the URL, there is a note: "Clone using the web URL." followed by a copy icon.

Other options in the menu include:

- Open with GitHub Desktop
- Open with Visual Studio
- Download ZIP



# Git Clone



```
$ git clone https://github.com/Omegapoint-Norge-Academy/secure-by-design.git  
$ cd secure-by-design/  
$ code .
```

# requests.http

- Install the VSCode extension: REST Client
- Send HTTP requests from VSCode
- Or use other tool of your choice to test manually

```
● ● ●  
###  
Send request  
GET https://localhost:7094/api/product  
  
###  
Send request  
# @name getReadToken  
POST https://omegapoint-norge-workshop.eu.auth0.com/oauth/token  
content-type: application/json  
  
{  
    "client_id": "<client-id-here>",  
    "client_secret": "<client-secret-here>",  
    "audience": "sales-api",  
    "grant_type": "client_credentials",  
    "scope": "products.read"  
}  
  
###  
@readToken = {{getReadToken.response.body.access_token}}  
  
###  
Send request  
GET http://localhost:5017/api/product  
Authorization: Bearer {{readToken}}
```

# Run the code

- Run the code with .NET 8

```
● ● ●  
> cd SalesAPI  
> cd 5-data-access-validation  
> dotnet run
```

# TL;DR

- Start coding in folder 0 Sales API
  - SalesAPI/0-starting-point/

Workshop: [bit.ly/op-secure-by-design](https://bit.ly/op-secure-by-design)

Client IDs and Secrets: [bit.ly/op-secure-by-design-secrets](https://bit.ly/op-secure-by-design-secrets)

