# **Exception Handling**

#### **1. UseDeveloperExceptionPage**

* **Purpose**: This middleware is used to display detailed error information during development to help developers diagnose issues quickly.
* **When to Use**: It is only enabled in the **Development** environment.
* **Behavior**:
  + It provides detailed error pages, including the exception’s stack trace, request details, and source code, if available.
  + The developer can see exactly where and why an exception occurred in the application.
* **Code Example**:
* if (app.Environment.IsDevelopment())  
  {  
   app.UseDeveloperExceptionPage();  
  }
* **Key Features**:
  + Shows the exception details (e.g., message, stack trace).
  + Displays request details, such as query parameters, headers, and more.
  + Helps developers quickly identify issues and fix them during development.
* **When NOT to Use**:
  + **In Production**: Exposing detailed error information in production can be a security risk, so it is **not enabled** in production environments.

#### **2. UseExceptionHandler**

* **Purpose**: This middleware is used to handle exceptions in a controlled way, often used in production environments to avoid exposing sensitive information to users.
* **When to Use**: It is typically enabled in the **Production** environment.
* **Behavior**:
  + It provides a centralized way of catching unhandled exceptions.
  + Developers can define a custom error-handling route (e.g., a global error page or API error response).
  + Unlike UseDeveloperExceptionPage, it does not display detailed exception information to the end user but can return a generic error message or redirect to a custom error page.
* **Code Example**:
* if (!app.Environment.IsDevelopment())  
  {  
   app.UseExceptionHandler("/Home/Error");  
  }
* **Key Features**:
  + Redirects to a designated error handler route (e.g., /Home/Error).
  + Ensures that no detailed exception data is exposed to the client in production.
  + Can be configured to return a custom error page or a generic JSON response.
* **When NOT to Use**:
  + **In Development**: It might obscure useful debugging information that is needed for development, so it is **not recommended** in development environments.

### **Comparison of UseDeveloperExceptionPage and UseExceptionHandler**

| Feature | UseDeveloperExceptionPage | UseExceptionHandler |
| --- | --- | --- |
| **Purpose** | Displays detailed error information for debugging | Handles unhandled exceptions and returns a user-friendly response |
| **When to Use** | Development environment only | Production environment |
| **Error Information Displayed** | Detailed error information (stack trace, request data) | Generic error message, hides exception details |
| **Customizable** | Limited customization, mainly for debugging | Highly customizable (redirect to a page, return JSON) |
| **Security** | Should not be used in production due to detailed error exposure | Safe for production, avoids exposing sensitive details |