

[Review the assignment due date](#)[Open in GitHub Codespaces](#)

proyecto-formatos-02



UNIVERSIDAD PRIVADA DE TACNA

FACULTAD DE INGENIERÍA

Escuela Profesional de Ingeniería de Sistemas

Proyecto: "SOLUCIONES PDF"

Curso: Calidad y prueba de software

Docente: Ing. Patricio José Cuadros Quiroga

Integrantes:

- Mario Antonio Flores Ramos (2018000597)
- Erick Javier Salinas Condori (2020069046)
- Fiorela Milady Ticahuanca Cutipa (2020068765)

Tacna – Perú

2024

Informe de factibilidad

Versión: "2.0"

CONTROL DE VERSIONES

Versión	Hecho por	Revisada por	Aprobado por	Fecha	Motivo
2.0	Erick Salinas, Mario Flores, Fiorela Ticahuanca	Erick Salinas, Mario Flores, Fiorela Ticahuanca	Patrick Cuadros	23/11/2024	Versión 2

ÍNDICE GENERAL

Resumen

El informe presenta el mejoramiento de la aplicación PDF Solutions, diseñada para gestionar documentos PDF de manera eficiente y segura. La aplicación incluye un sistema de inicio de sesión y suscripciones, con diferentes niveles de acceso para los usuarios. Su enfoque está en facilitar tareas como combinar y dividir archivos PDF, resolviendo problemas comunes de manipulación de documentos. Con opciones de suscripción, los usuarios acceden a funciones avanzadas según sus necesidades. El objetivo es ofrecer una solución centralizada, accesible y optimizada, mejorando la experiencia de usuario y simplificando procesos manuales en el manejo de archivos PDF.

Abstracto

El informe presenta la mejora de la aplicación PDF Solutions, diseñada para gestionar de forma eficiente y segura los documentos PDF. La aplicación incluye un sistema de login y suscripciones, ofreciendo diferentes niveles de acceso para los usuarios. Se centra en simplificar tareas como la combinación y división de archivos PDF, solucionando problemas habituales en el manejo de documentos. Con las opciones de suscripción, los usuarios pueden acceder a funciones avanzadas en función de sus necesidades. El objetivo es proporcionar una solución centralizada, accesible y optimizada, mejorando la experiencia del usuario y simplificando los procesos manuales en la gestión de archivos PDF.

[1. Antecedentes o introducción](#)

[2. Título](#)

[3. Autores](#)

[4. Plantación del problema](#)

[4.1 Problemas](#)

[4.2 Justificación](#)

[4.3 Alcance](#)

[5. Objetivos](#)

[5.1 General](#)

[5.2 Especificaciones](#)

[6. Referentes teóricos](#)

7. Desarrollo de la propuesta

7.1 Tecnología de información

7.2 Metodología, técnicas usadas

7. Cronograma

Tema: Mejoramiento de la Aplicación PDF SOLUCIÓN

1. Antecedentes o introducción

El presente informe aborda el mejoramiento de la aplicación PDF Solutions, desarrollada para gestionar documentos PDF de manera eficiente y segura. Esta aplicación se ha construido con una arquitectura que integra un sistema de inicio de sesión y suscripciones, ofreciendo diferentes niveles de acceso a los usuarios según su tipo de suscripción. El proyecto se enfoca en proporcionar herramientas para el manejo de archivos PDF, permitiendo a los usuarios realizar tareas como juntar y cortar documentos de manera sencilla y rápida.

PDF Solutions busca resolver problemas comunes en la manipulación de archivos PDF, como la falta de opciones para combinar varios documentos o dividir un archivo grande en secciones más pequeñas. La aplicación está diseñada para ser intuitiva y funcional, ofreciendo una experiencia de usuario amigable y eficiente. Con opciones de suscripción, los usuarios pueden acceder a funcionalidades avanzadas según sus necesidades, asegurando un servicio personalizado y adaptable.

La mejora de esta aplicación tiene como objetivo proporcionar una solución centralizada y accesible para la gestión de archivos PDF, reduciendo los errores y simplificando procesos que habitualmente son manuales y tediosos. Además, ofrece una experiencia de usuario optimizada, facilitando la manipulación de documentos y brindando un servicio confiable y eficiente para individuos y empresas que requieren un manejo eficaz de sus archivos PDF.

2. Título

MEJORAMIENTO DE LA APLICACIÓN SOLUCIONES PDF

3. Autores

- Mario Antonio Flores Ramos (2018000597)
- Erick Javier Salinas Condori (2020069046)
- Fiorela Milady Ticahuanca Cutipa (2020068765)

4. Plantación del problema

4.1. Problema:

El manejo y la manipulación de archivos PDF presentan desafíos para muchos usuarios, especialmente cuando se requiere combinar o dividir documentos de manera eficiente. Las herramientas tradicionales o la falta de un sistema especializado dificultan la gestión de archivos, resultando en procesos tediosos y propensos a errores. Además, la ausencia de una plataforma que ofrezca funcionalidades avanzadas de manera intuitiva afecta la productividad y la experiencia del usuario.

Esta situación genera ineficiencias, pérdida de tiempo y una experiencia poco satisfactoria al trabajar con documentos PDF. Por lo tanto, es necesario contar con una aplicación que proporcione una solución centralizada y fácil de usar, que permita gestionar y manipular archivos PDF de forma segura, eficiente y adaptable a las necesidades de los usuarios.

4.2. Justificación:

La mejora de la aplicación PDF Solutions es fundamental para proporcionar a los usuarios una herramienta eficiente y segura en la gestión de archivos PDF. La implementación de funcionalidades como el login, suscripciones y opciones avanzadas para juntar y cortar PDF permite optimizar la experiencia del usuario y facilitar la manipulación de documentos de manera efectiva.

Contar con una solución centralizada y automatizada reduce la complejidad y el tiempo necesario para gestionar archivos PDF, permitiendo a los usuarios realizar estas tareas de manera rápida y precisa. Esto no solo mejora la productividad, sino que también ofrece una experiencia más satisfactoria y accesible, adaptándose a las diversas necesidades de los usuarios. Al brindar un sistema confiable y fácil de usar, se contribuye a la optimización de procesos y la mejora general de la gestión de documentos PDF.

4.3. Alcance:

Este documento se enfoca en el mejoramiento de la aplicación PDF Solutions, implementando mejoras en la vista lógica del sistema y asegurando una experiencia de usuario eficiente para la gestión de archivos PDF. Se describen los aspectos fundamentales de las funcionalidades principales, omitiendo detalles específicos de las vistas de procesos.

- **Gestión de archivos PDF :** Implementación de funciones para juntar y cortar archivos PDF, permitiendo a los usuarios manipular documentos de manera sencilla y eficiente.
- **Sistema de autenticación y suscripciones :** Desarrollo de un sistema de inicio de sesión y gestión de suscripciones, proporcionando acceso controlado a funciones avanzadas según el tipo de suscripción.
- **Experiencia de usuario optimizada :** Creación de una interfaz de usuario intuitiva y fácil de usar para mejorar la interacción con la aplicación.
- **Administración de suscripciones :** Gestión de los niveles de acceso y suscripciones de los usuarios, facilitando la administración y actualización de sus permisos.
- **Seguridad y privacidad :** Garantizar la integridad y confidencialidad de los datos del usuario, ofreciendo un sistema de autenticación seguro.
- **Vista lógica :** Gestión de usuarios, manipulación de archivos PDF, administración de suscripciones y funcionalidades de control de acceso.
- **Vista de desarrollo :** Implementación del patrón MVC con C# y ASP.NET Core estructura, organizada del código y uso de buenas prácticas de programación.

- **Vista física :** Despliegue de la aplicación en un entorno web, asegurando la compatibilidad y el rendimiento en servidores y bases de datos.

5. Objetivos

5.1. General

Optimizar la aplicación PDF Solutions para mejorar la gestión y manipulación de archivos PDF, brindando funcionalidades avanzadas de manera eficiente y fácil de usar.

5.8. Específico

- Implementar la funcionalidad de inicio de sesión y suscripciones: Garantizando un acceso seguro y personalizado para los usuarios según su nivel de suscripción.
- Desarrollar las opciones para juntar y cortar archivos PDF: Permitiendo que los usuarios realicen estas acciones de manera sencilla y eficiente.
- Optimizar la experiencia del usuario en la manipulación de archivos PDF: Asegurando un proceso intuitivo y fácil de utilizar.
- Integrar herramientas de administración para gestionar las suscripciones de los usuarios: Facilitando la modificación y actualización de sus niveles de acceso.
- Implementar un sistema de validación de usuarios para asegurar la integridad y privacidad de la información.
- Proveer una interfaz amigable y accesible que permita a los usuarios gestionar y manipular sus documentos PDF de manera efectiva.

6. Referentes teóricos

Diagramas de Casos de Uso, Diagrama de Clases, Diagrama de Componentes y Arquitectura.

Diagrama de caso de uso:



Diagrama de clases:

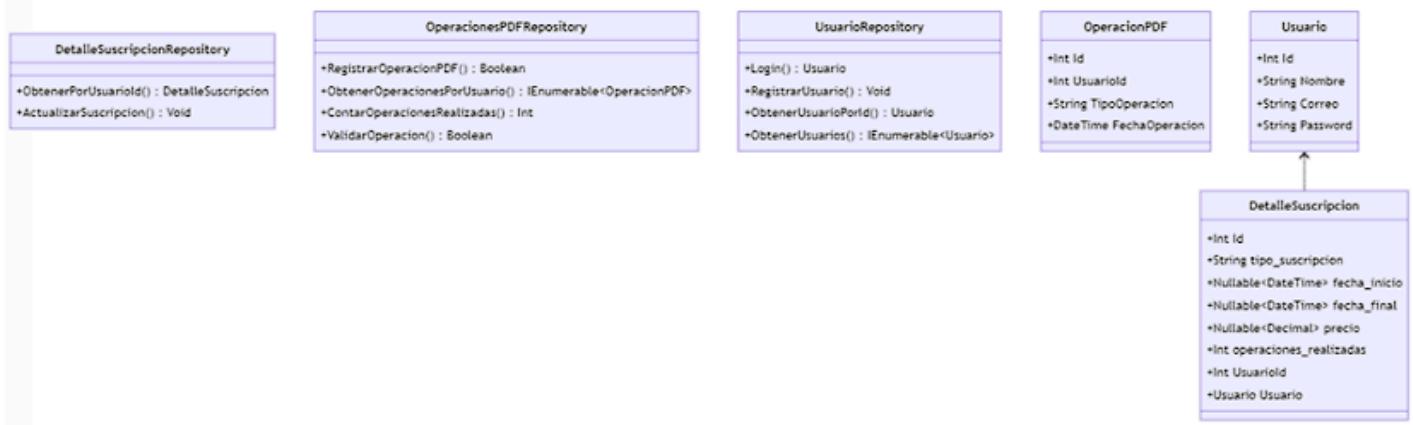
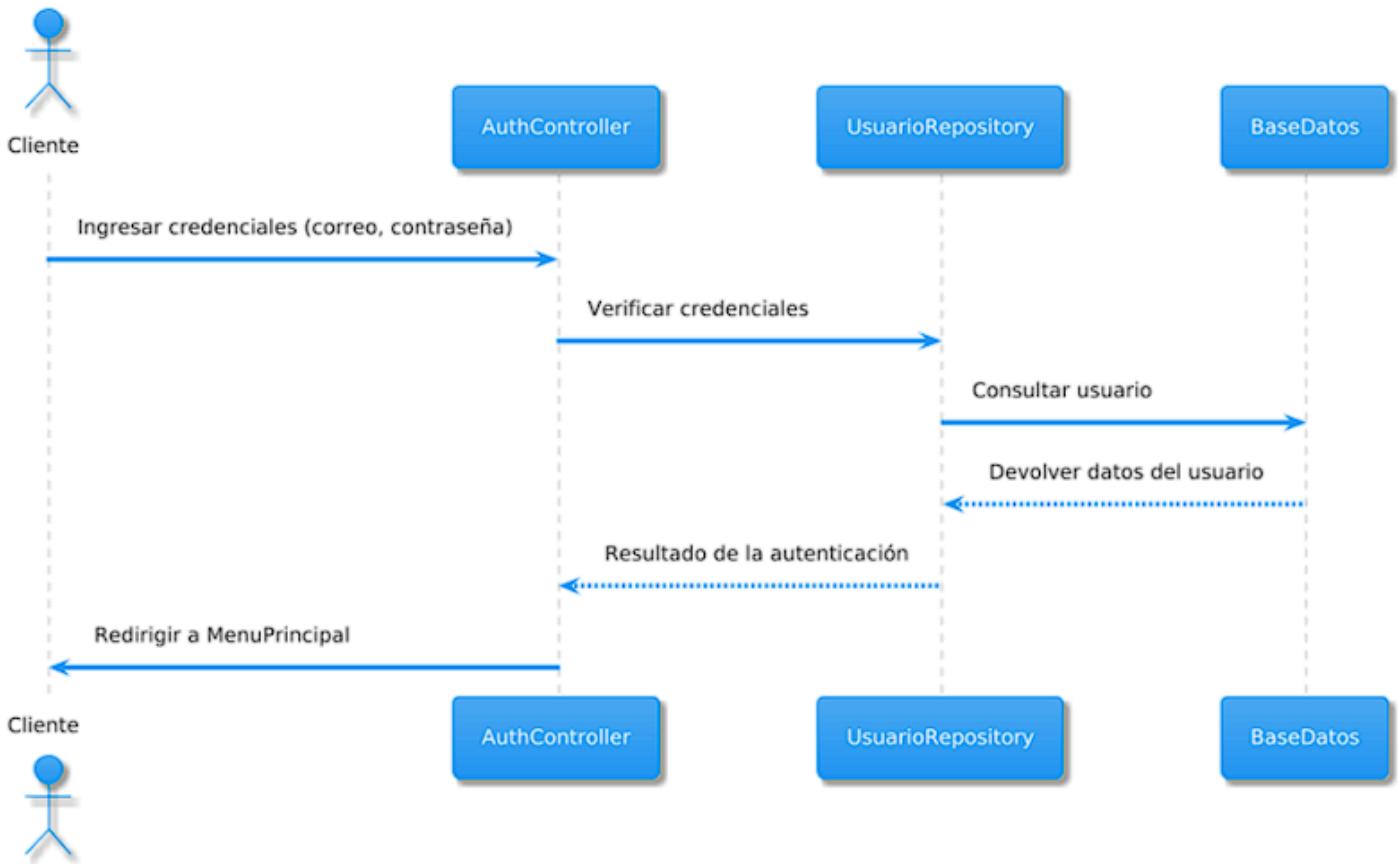
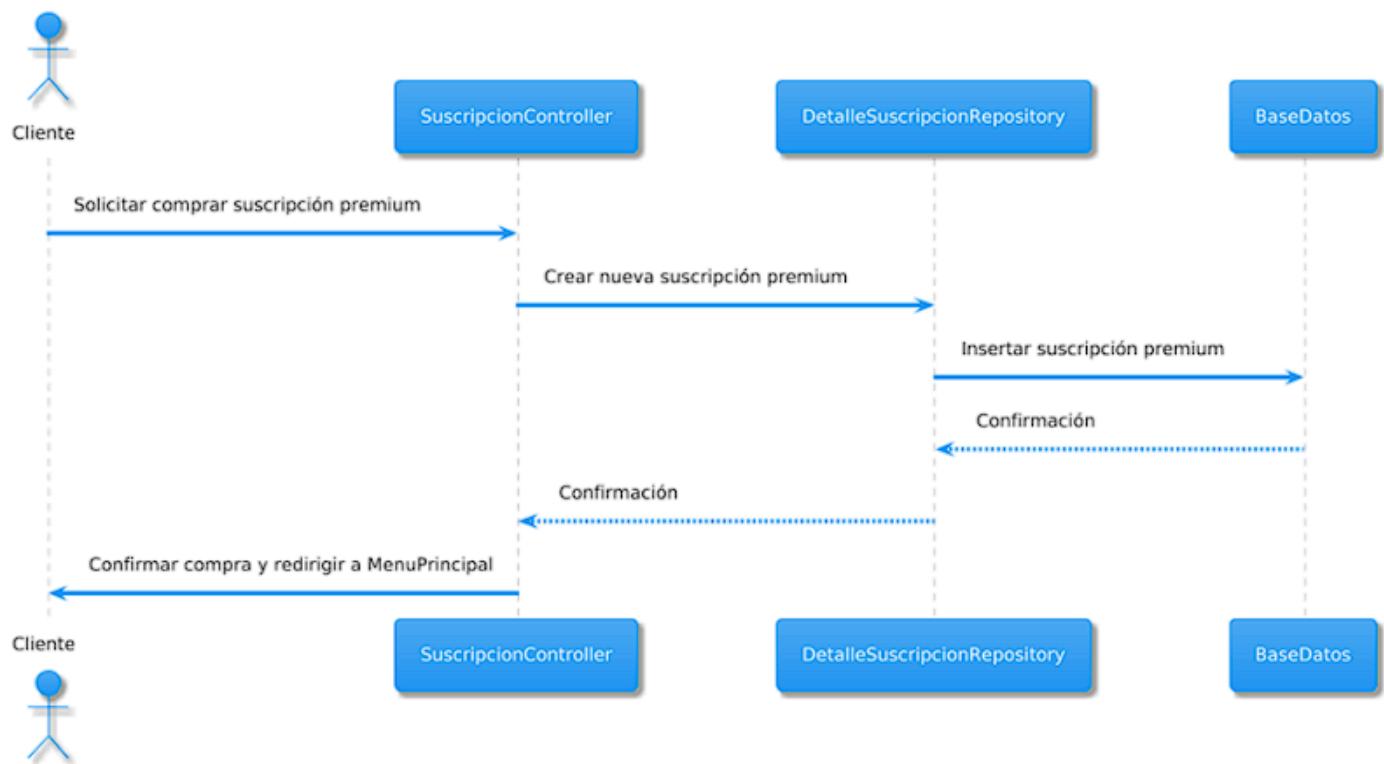


Diagrama de secuencia:

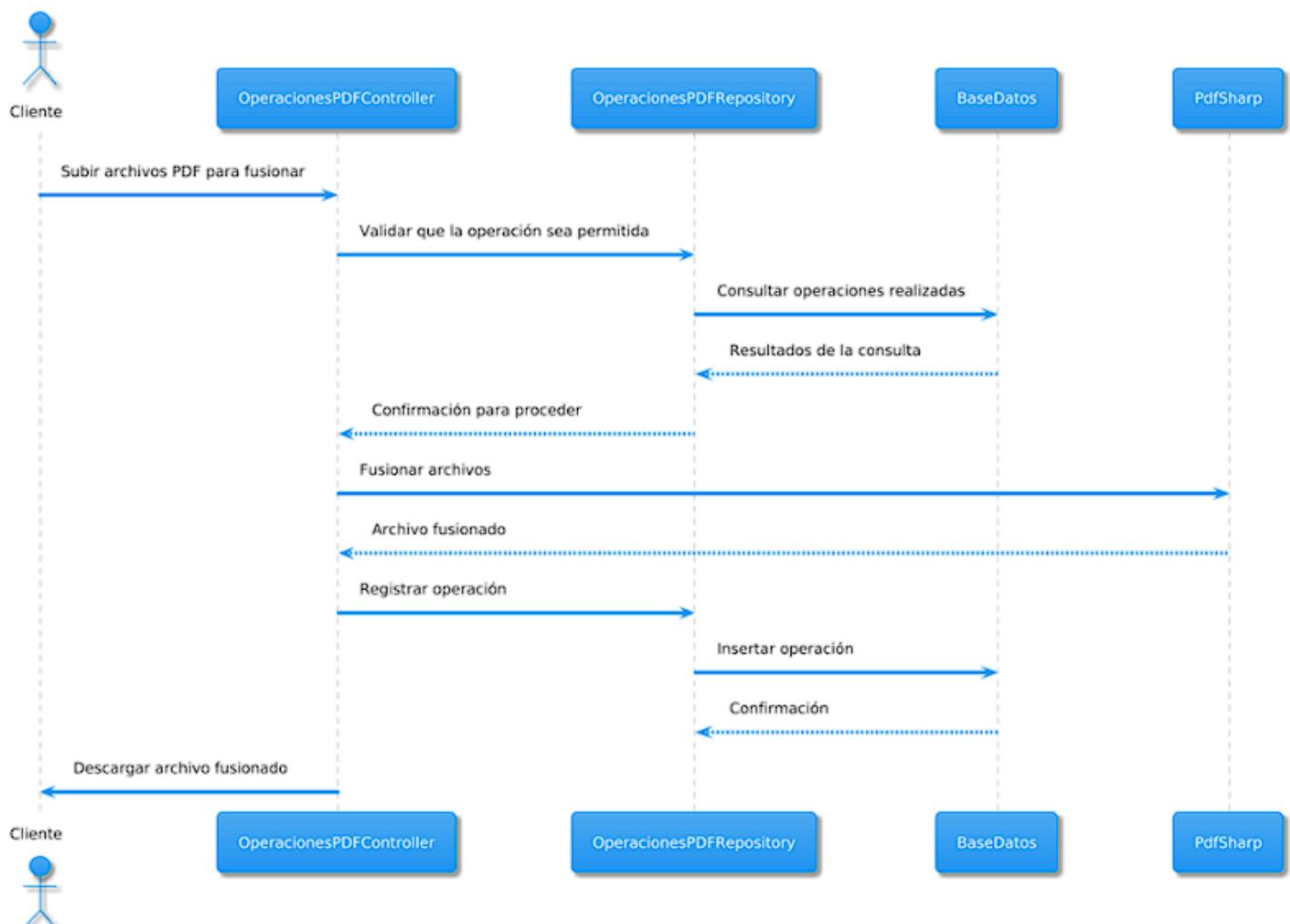
Inicio Sesión



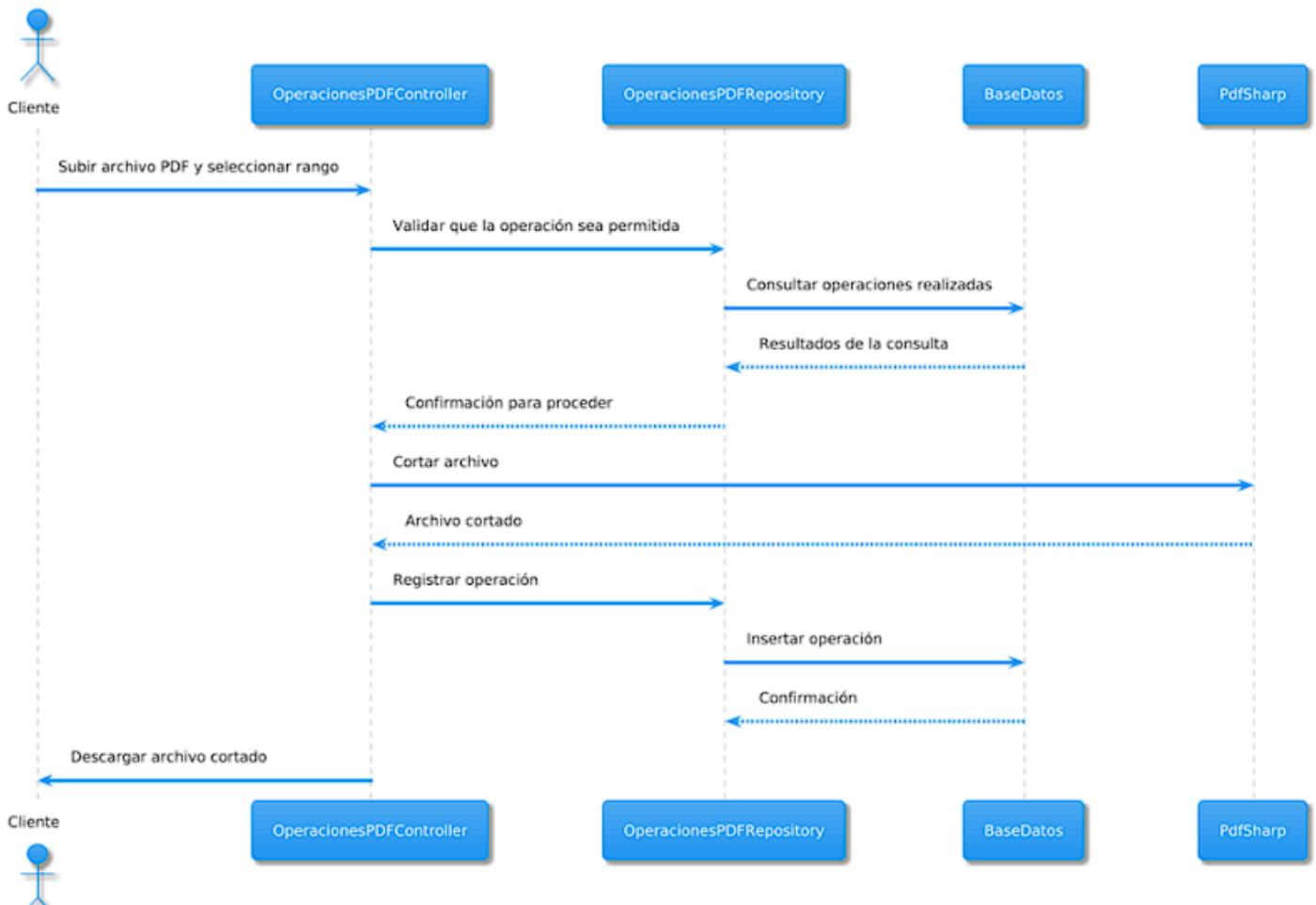
Comprar Suscripción Premium



PDF de fusión



Cortar PDF



Ver operaciones realizadas

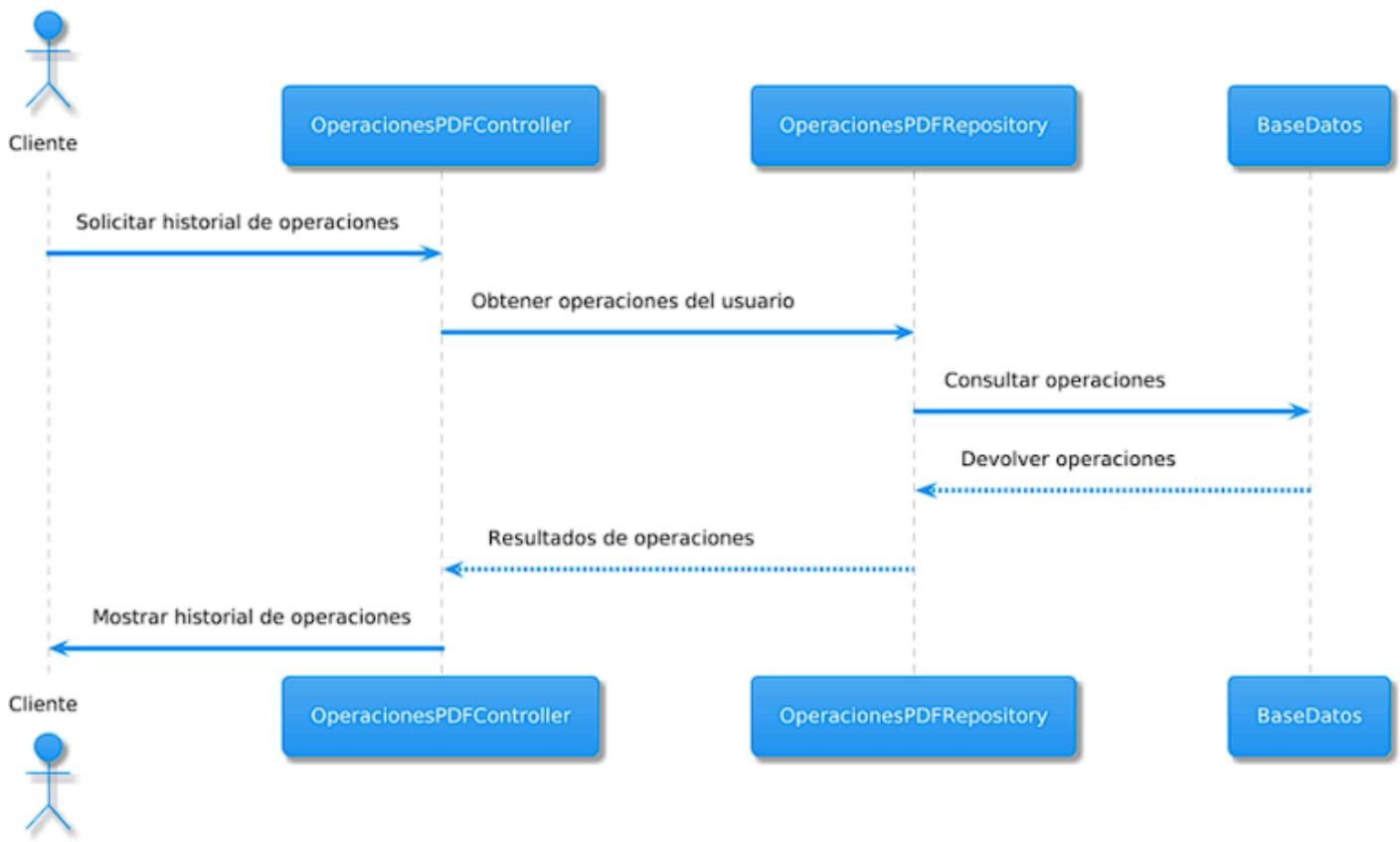


Diagrama de componentes:



Diagrama de Arquitectura:

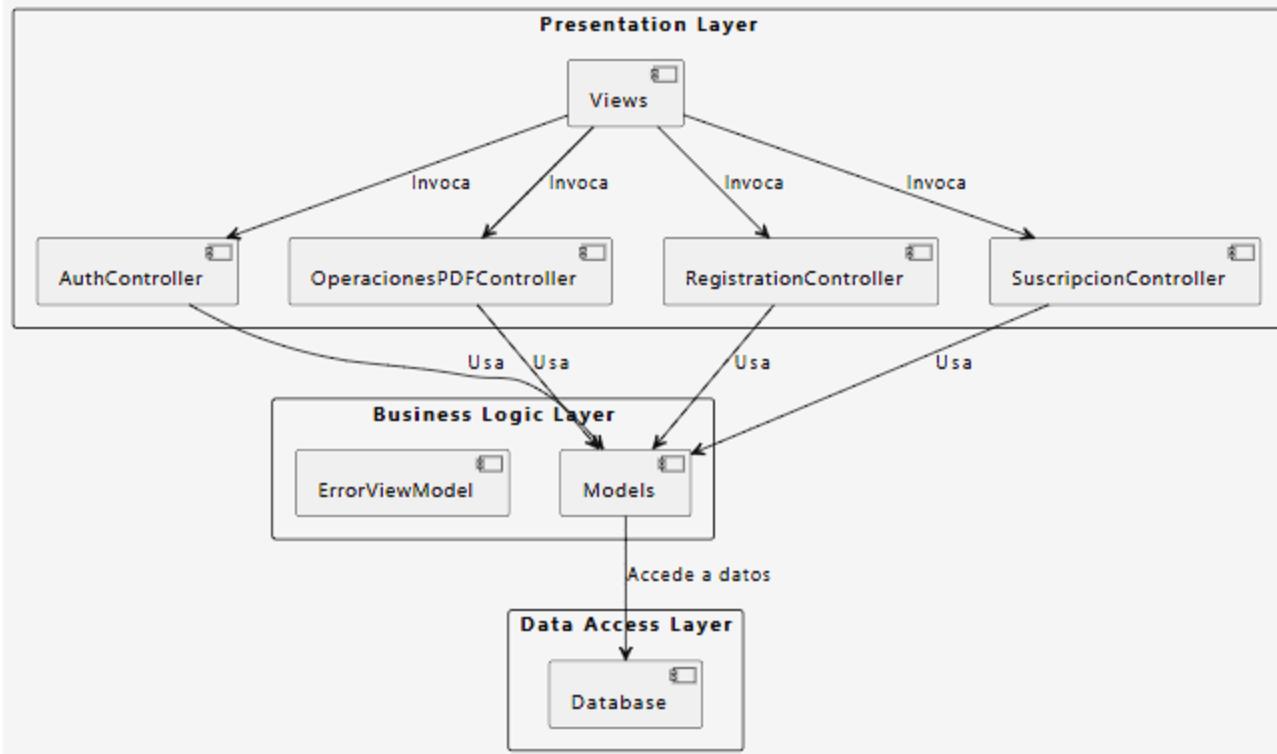


Diagrama de Despliegue:

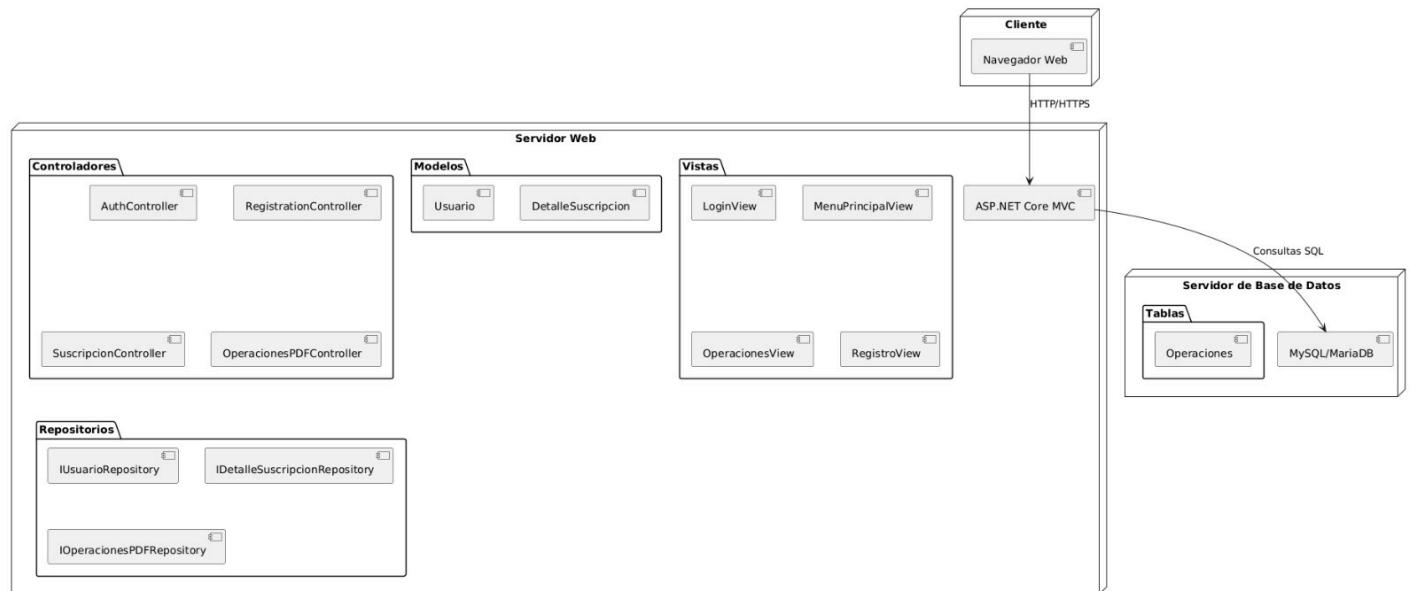


Diagrama Base de datos relacionales:

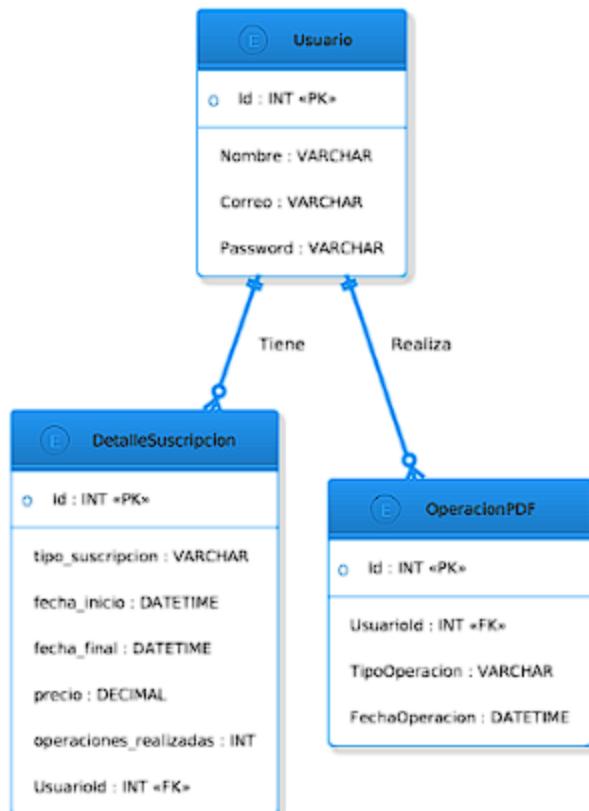
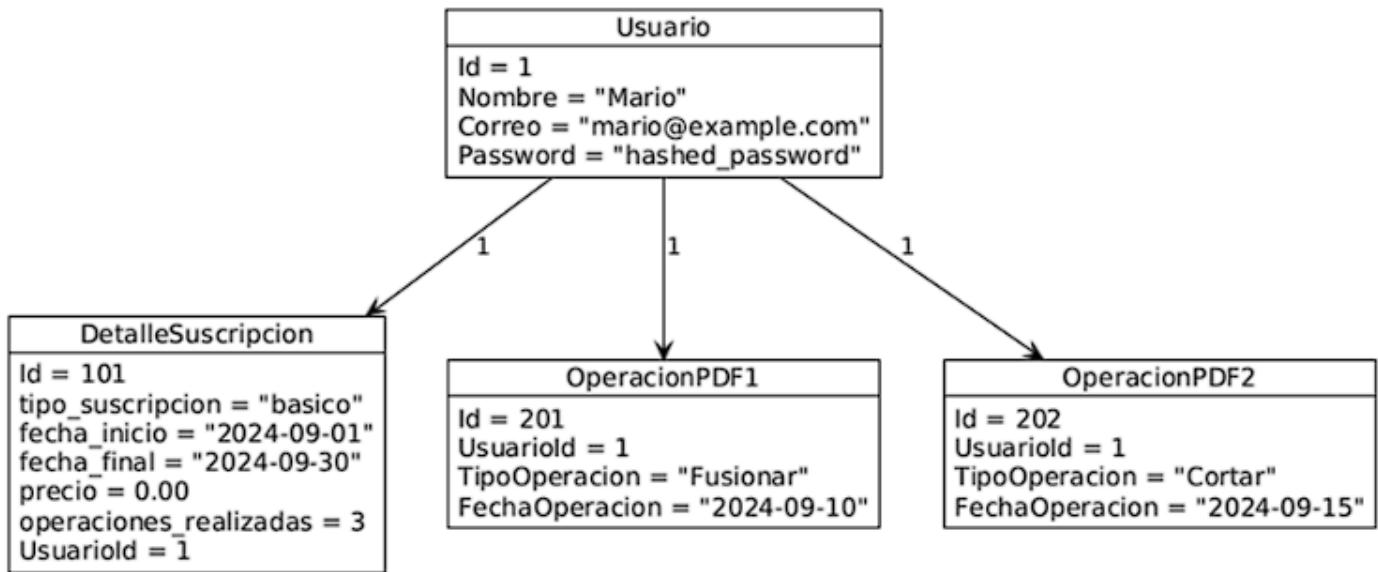


Diagrama de objetos:



7. Desarrollo de la propuesta (Aquí va el análisis de su aplicación con SonarQube y Snyk, para que les muestren todos los aspectos a mejorar de su aplicación)

Análisis de la aplicación con SonarQube: (Anteriormente el código se encuentra así)

Select issues ⌂ ⌃ ⌄ ⌅ Navigate to issue ⌂ ⌅

39 Issues 2h 14min effort ⌂ ⌃ ⌄ ⌅

mario_test / mario_test / NegocioPDF.Tests/UsuarioRepositoryTests.cs

Consistency

Consider using the constraint model, `Assert.That(actual, Is.EqualTo(expected))`, instead of the classic model, `Assert.AreEqual(expected, actual)` No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort 1 day ago

Consistency

Consider using the constraint model, `Assert.That(actual, Is.EqualTo(expected))`, instead of the classic model, `Assert.AreEqual(expected, actual)` No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort 1 day ago

Consistency

Consider using the constraint model, `Assert.That(actual, Is.EqualTo(expected))`, instead of the classic model, `Assert.AreEqual(expected, actual)` No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort 1 day ago

Consistency

Consider using the constraint model, `Assert.That(expr, Is.Not.Null)`, instead of the classic model, `Assert.IsNotNull(expr)` No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Info ROSLYN 0min effort 1 day ago

Consistency

Consider using the constraint model, `Assert.That(actual, Is.EqualTo(expected))`, instead of the classic model, `Assert.AreEqual(expected, actual)` No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort 1 day ago

Select issues ⌂ ⌃ ⌄ ⌅ Navigate to issue ⌂ ⌅

33 issues 2h 14min effort ⌂ ⌃ ⌄ ⌅

mario_test / mario_test / NegocioPDF/Models/DetalleSuscripcion.cs

Consistency

Non-nullable property 'tipo_suscripcion' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable. No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort 1 day ago

Consistency

Non-nullable property 'Usuario' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable. No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort 1 day ago

mario_test / mario_test / NegocioPDF/Models/OperacionPDF.cs

Consistency

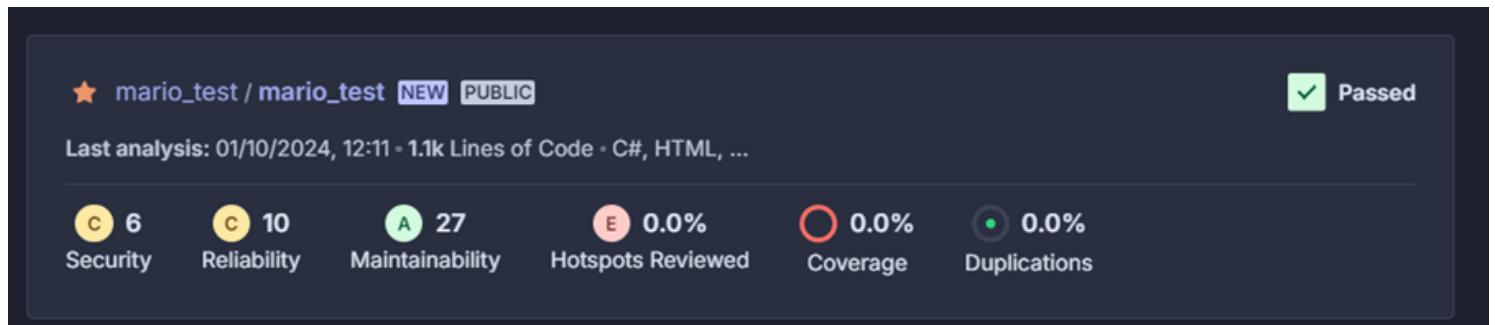
Rename class 'OperacionPDF' to match pascal case naming rules, consider using 'OperacionPdf'. convention +

Open Mario Antonio Flores Ramos Maintainability Code Smell Minor ROSLYN 5min effort 1 day ago

Consistency

Non-nullable property 'TipoOperacion' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable. No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort 1 day ago



Select issues ◀ ▶ Navigate to issue ◀ ▶ ⌚ 33 issues 2h 14min effort 🏠

mario_test / mario_test / NegocioPDF/Models/DetalleSuscripcion.cs

Consistency

Non-nullable property 'tipo_suscripcion' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.

No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort • 1 day ago

Consistency

Non-nullable property 'Usuario' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.

No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort • 1 day ago

mario_test / mario_test / NegocioPDF/Models/OperacionPDF.cs

Consistency

Rename class 'OperacionPDF' to match pascal case naming rules, consider using 'OperacionPdf'.

convention +

Open Mario Antonio Flores Ramos Maintainability Code Smell Minor 5min effort • 1 day ago

Consistency

Non-nullable property 'TipoOperacion' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.

No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major ROSLYN 0min effort • 1 day ago

mario_test / mario_test / NegocioPDF/Models/Usuario.cs

Activar Windows
Ve a Configuración para activar Wind

mario_test / mario_test / NegocioPDF/Models/Usuario.cs

Consistency

Non-nullable property 'Nombre' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

Consistency

Non-nullable property 'Correo' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

Consistency

Non-nullable property 'Password' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

mario_test / mario_test / NegocioPDF/Repositories/DetalleSuscripcionRepository.cs

Consistency

Possible null reference return.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

mario_test / mario_test / NegocioPDF/Repositories/OperacionesPDFRepository.cs

Consistency

Rename class 'OperacionesPDFRepository' to match pascal case naming rules, consider using 'OperacionesPdfRepository'.

Activar Windows convention +

Ve a Configuración para activar Wind

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Minor

5min effort • 1 day ago

Consistency

Dereference of a possibly null reference.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

Consistency

Dereference of a possibly null reference.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

mario_test / mario_test / NegocioPDF/Repositories/UsuarioRepository.cs

Consistency

Possible null reference return.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

Intentionality

'System.Exception' should not be thrown by user code.

cwe error-handling +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

20min effort • 1 day ago

Consistency

Possible null reference return.

No tags +

 Open Mario Antonio Flores Ramos

Maintainability

Code Smell

Major

ROSLYN

0min effort • 1 day ago

Responsibility

ModelState.IsValid should be checked in controller actions.

asp.net +

Open Mario Antonio Flores Ramos Security Maintainability Reliability Code Smell Major 5min effort 1 day ago

mario_test / mario_test / PROYECTOPDF/Controllers/OperacionesPDFController.cs

Consistency

Rename class 'OperacionesPDFController' to match pascal case naming rules, consider using 'OperacionesPdfController'.

convention +

Open Mario Antonio Flores Ramos Maintainability Code Smell Minor 5min effort 1 day ago

Responsibility

ModelState.IsValid should be checked in controller actions.

asp.net +

Open Mario Antonio Flores Ramos Security Maintainability Reliability Code Smell Major 5min effort 1 day ago

Adaptability

Define a constant instead of using this literal 'Error' 7 times.

design +

Open Mario Antonio Flores Ramos Maintainability Code Smell Minor 4min effort 1 day ago

Intentionality

Make 'CopyPages' a static method.

pitfall +

Open Mario Antonio Flores Ramos Maintainability Code Smell Minor 5min effort 1 day ago

Consistency

Member 'CopyPages' does not access instance data and can be marked as static

No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Info ROSLYN 0min effort 1 day ago

Responsibility

ModelState.IsValid should be checked in controller actions.

asp.net +

Open Mario Antonio Flores Ramos Security Maintainability Reliability Code Smell Major 5min effort 1 day ago

Adaptability

Define a constant instead of using this literal 'CortarPDF' 4 times.

design +

Open Mario Antonio Flores Ramos Maintainability Code Smell Minor 4min effort 1 day ago

Responsibility

Change this code to not construct the path from user-controlled data.

cwe +

Open Mario Antonio Flores Ramos Security Vulnerability Major 30min effort 1 day ago

Responsibility

ModelState.IsValid should be checked in controller actions.

asp.net +

Open Mario Antonio Flores Ramos Security Maintainability Reliability Code Smell Major 5min effort 1 day ago

Responsibility

ModelState.IsValid should be checked in controller actions.

Activar Windows asp.net +

Open Mario Antonio Flores Ramos Security Maintainability Reliability Code Smell Major 5min effort 1 day ago

Ve la Configuración para activar Wind

Adaptability

Move 'RegistrationController' into a named namespace.

No tags +

Open Mario Antonio Flores Ramos Reliability Bug Major

5min effort • 1 day ago

Consistency

Declare types in namespaces

No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell

ROSLYN 0min effort • 1 day ago

mario_test / mario_test / PROYECTOPDF/Controllers/SuscripcionController.cs

Intentionality

Remove the unused local variable 'usuarioid'.

unused +

Open Mario Antonio Flores Ramos Maintainability Code Smell Minor

5min effort • 1 day ago

mario_test / mario_test / PROYECTOPDF/Program.cs

Intentionality

Await RunAsync instead.

async-await +

Open Mario Antonio Flores Ramos Reliability Code Smell Major

5min effort • 1 day ago

mario_test / mario_test / PROYECTOPDF/Views/Registration/Registrarse.cshtml

Intentionality

A form label must be associated with a control.

accessibility +

Open Mario Antonio Flores Ramos Reliability Code Smell Minor

Activar Windows 5min effort • 1 day ago
Ve a Configuración para activar Windows

mario_test / mario_test / PROYECTOPDF/Views/Registration/Registrarse.cshtml

Intentionality

A form label must be associated with a control.

accessibility +

Open Mario Antonio Flores Ramos Reliability Code Smell Minor

5min effort • 1 day ago

Intentionality

A form label must be associated with a control.

accessibility +

Open Mario Antonio Flores Ramos Reliability Code Smell Minor

5min effort • 1 day ago

Intentionality

A form label must be associated with a control.

accessibility +

Open Mario Antonio Flores Ramos Reliability Code Smell Minor

5min effort • 1 day ago

mario_test / mario_test / PROYECTOPDF/wwwroot/css/site.css

Intentionality

Unexpected duplicate selector "html", first used at line 1

No tags +

Open Mario Antonio Flores Ramos Maintainability Code Smell Major

1min effort • 1 month ago

Activar Windows

Como se puede ver las imágenes muestran un análisis de código que identifica problemas de mantenibilidad y consistencia en el proyecto.

Análisis de la aplicación con Snyk

El problema mostrado en el código es que se está usando directamente una ruta de archivo que viene de una entrada del usuario, sin verificar si es segura. Esto podría permitir que un atacante cambie esa ruta y acceda a archivos que no debería.

The screenshot shows the Snyk Code Report interface. At the top, it displays the date and time: "October 1st 2024, 5:07:46 pm (UTC+00:00)" and the source location: "Source: /home/runner/work/proyecto-si784-2024-ii-u1-FloresR_Salinas_ticahuana/proyecto...". Below this, the report title is "Snyk Code Report" and the coverage is listed as "SCAN COVERAGE". The issue count is 11: 1 high issues, 0 medium issues, and 11 low issues. The report header for the specific issue is "H Path Traversal" with categories "SNYK-CODE", "CWE-23", and "PT". A "Data Flow" button is highlighted in red, and a "Fix Analysis" button is also present. The main content area describes the vulnerability: "Unsanitized input from an HTTP parameter flows into global::System.IO.File.Exists, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to obtain information about arbitrary files." It notes the file is "Found in: PROYECTOPDF/Controllers/OperacionesPDFController.cs (line : 109)". The "Data Flow" section shows the code snippet with annotations: line 96:39 has 0 SOURCE and 1 SINK; line 109:73 has 2 SOURCE and 2 SINK; line 109:51 has 3 SOURCE and 3 SINK. The code snippet is as follows:

```
PROYECTOPDF/Controllers/OperacionesPDFController.cs

96:39 public IActionResult CortarArchivoPDF( string rutaArchivoTemp, int startPage, int endPage)
96:39 public IActionResult CortarArchivoPDF( string rutaArchivoTemp, int startPage, int endPage)
109:73 if (string.IsNullOrEmpty(rutaArchivoTemp) || !System.IO.File.Exists(rutaArchivoTemp))
109:51 if (string.IsNullOrEmpty(rutaArchivoTemp) || !System.IO.File.Exists(rutaArchivoTemp))
```

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\OperacionesPDFController.cs (line : 39)

Data Flow

39:22 public IActionResult FusionarArchivosPDF(IFormFile archivo1, IFormFile archivo2)

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\OperacionesPDFController.cs (line : 96)

Data Flow

96:22 public IActionResult CortarArchivoPDF(string rutaArchivoTemp, int startPage, int endPage)

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

Data Flow Fix Analysis

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF/Controllers/OperacionesPDFController.cs (line : 150)

! Data Flow

```
150:22 public IActionResult ObtenerTotalPaginas( IFormFile pdffile)
```

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

Data Flow Fix Analysis

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF/Controllers/OperacionesPDFController.cs (line : 173)

! Data Flow

```
173:22 public IActionResult CargarArchivoTemporal( IFormFile pdfFile)
```

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\AuthController.cs (line : 28)

Data Flow

PROYECTOPDF\Controllers\AuthController.cs

```
28:30 public IActionResult Login( string correo, string password)
```

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\AuthController.cs (line : 65)

Data Flow

PROYECTOPDF\Controllers\AuthController.cs

```
65:30 public IActionResult Logout()
```

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\SuscripcionController.cs (line : 19)

Data Flow

PROYECTOPDF\Controllers\SuscripcionController.cs

```
19:38 public IActionResult MenuPrincipal()
```

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\SuscripcionController.cs (line : 43)

Data Flow

PROYECTOPDF\Controllers\SuscripcionController.cs

```
43:34 public IActionResult ComprarPremium()
```

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

Data Flow Fix Analysis

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\SuscripcionController.cs (line : 62)

Data Flow

62:34 public IActionResult ConfirmarCompra()

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

Data Flow Fix Analysis

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF\Controllers\RegistrationController.cs (line : 22)

Data Flow

PROYECTOPDF\Controllers\RegistrationController.cs

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF/Controllers/SuscripcionController.cs (line : 62)

Data Flow

```
62:34 public IActionResult ConfirmarCompra()
```

SOURCE SINK 0

L Anti-forgery token validation disabled

SNYK-CODE | CWE-352 | AntiforgeryTokenDisabled

This ASP.NET MVC action should use an anti-forgery validation attribute. Not using this attribute disables Cross Site Request Forgery (CSRF) protection and allows CSRF attacks.

Found in: PROYECTOPDF/Controllers/RegistrationController.cs (line : 22)

Data Flow

```
PROYECTOPDF/Controllers/RegistrationController.cs
```

```
22:26 public IActionResult Registrarse(Usuario usuario)
```

SOURCE SINK 0

Informe Snyk

The screenshot shows a browser window displaying the Snyk test report for a project. The title bar reads "upt-fang-epi.github.io/proyecto-sf784-2024-1-e2-foresR_salinas_stahwanda/reports/snyk-report.html". The main header is "snyk" and "Snky test report". Below it, it says "Scanned the following path:" followed by a single path entry: "Home\humen\work\proyecto-sf784-2024-1-e2-foresR_salinas_stahwanda\proyecto-sf784-2024-1-e2-foresR_salinas_stahwanda". It also shows "0 known vulnerabilities" and "0 vulnerable dependency paths". The bottom section displays the scanned path again and a message "No known vulnerabilities detected".

El reporte de Snyk muestra que no se detectan vulnerabilidades conocidas en las dependencias escaneadas dentro del proyecto analizado. Esto indica que las bibliotecas utilizadas actualmente no presentan riesgos de seguridad según la base de datos de vulnerabilidades de Snyk, lo que es una señal positiva en términos de seguridad para las dependencias externas del proyecto.

Análisis de la aplicación con Semgrep

tool	code	severity	confidence	function	file	line	position	message
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\AuthController.cs	27	10	Login is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\OperacionesPDFController.cs	38	2	FusionarArchivosPDF is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\OperacionesPDFController.cs	95	2	CortarArchivoPDF is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\OperacionesPDFController.cs	149	2	ObtenerTotalPaginas is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\OperacionesPDFController.cs	172	2	CargarArchivoTemporal is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\RegistrationController.cs	21	6	Registrarse is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
								checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.

								checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\OperacionesPDFController.cs	172	2	CargarArchivoTemporal is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\RegistrationController.cs	21	6	Registrarse is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csharp.dotnet.security.audit.mass-assignment.mass-assignment	WARNING	MEDIUM	unknown	PROYECTOPDF\Controllers\RegistrationController.cs	37	16	Mass assignment or Autobinding vulnerability in code allows an attacker to execute over-posting attacks, which could create a new parameter in the binding request and manipulate the underlying object in the application.
semgrep	csharp.dotnet.security.mvc-missing-antiforgery.mvc-missing-antiforgery	WARNING	LOW	unknown	PROYECTOPDF\Controllers\SuscripcionController.cs	61	30	ConfirmarCompra is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	javascript.lang.security.audit.detect-non-literal-regexp.detect-non-literal-regexp	WARNING	LOW	unknown	PROYECTOPDF\wwwroot\lib\jquery-validation-unobtrusive\jquery.validate.unobtrusive.js	349	17	RegExp() called with a 'params' function argument, this might allow an attacker to cause a Regular Expression Denial-of-Service (ReDoS) within your application as RegExp [®] blocks the main thread. For this reason, it is recommended to use hardcoded regexes instead. If your regex is run on user-controlled input, consider performing input validation or use a regex checking/sanitization library such as recheck to verify that the regex https://www.npmjs.com/package/recheck does not appear vulnerable to ReDoS.

Informe de Semgrep

tool	code	severity	confidence	function	file	line	position	message
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\AuthController.cs	27	19	Login is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\OperacionesPDFController.cs	38	2	Logout is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\OperacionesPDFController.cs	95	2	ChangePassword is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\OperacionesPDFController.cs	129	2	ObtenerTotalPaginas is a state-changing MVC method that does not validate the antiforgery token or do strict content-type checking. State-changing controller methods should either enforce antiforgery tokens or do strict content-type checking to prevent simple HTTP request types from bypassing CORS preflight controls.
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\OperacionesPDFController.cs	172	2	OrganizarPorTemporal es un método de tipo POST que no valida el token de antiforgery o la revisión del contenido. Los controladores de tipo cambio deben validar el token de antiforgery o la revisión del contenido para evitar tipos de solicitud HTTP simples que eviten las restricciones CORS de pre vuelo.
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\RegistrationController.cs	21	6	Registrar es un método de tipo POST que no valida el token de antiforgery o la revisión del contenido. Los controladores de tipo cambio deben validar el token de antiforgery o la revisión del contenido para evitar tipos de solicitud HTTP simples que eviten las restricciones CORS de pre vuelo.
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	MEDIUM	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\RegistrationController.cs	37	16	MostrarAsignacion o AsignarDedicacion en código abierto se utiliza para ejecutar ataques de sobrecarga. Esto crea una nueva persona en la solicitud de request y manipula el objeto de respuesta.
semgrep	csrf-draft-security-mvc-missing-and-forgery-mvc-missing-and-forgery	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\Controllers\SubscriptionController.cs	61	36	ConfirmacionEsperada es un método de tipo POST que no valida el token de antiforgery o la revisión del contenido. Los controladores de tipo cambio deben validar el token de antiforgery o la revisión del contenido para evitar tipos de solicitud HTTP simples que eviten las restricciones CORS de pre vuelo.
semgrep	password-long-security-audit-detected-mvc-regexp-detected-mvc-regexp	WARNING	LOW	unknown	PROYECTO_PDF\PROJECTOPDF\wwwroot\lib\jquery-validation-unobtrusive\jquery.validate.unobtruse.js	349	17	RegExp() validWith • parent function argument. This might allow an attacker to cause a Regular Expression Denial of Service (ReDoS) within your application as RegExp() blocks the main thread. For this reason, it is recommended to use backtracking assertions instead. If you're regex is too an user-controlled input, consider using getSetTimeout() to set a timeout for regular expression evaluation. https://cheatsheetseries.owasp.org/cheatsheets/Regular_Expression_Denial_of_Service_ReDoS.html https://cheatsheetseries.owasp.org/cheatsheets/Regular_Expression_Denial_of_Service_ReDoS.html

El informe de Semgrep identifica varias advertencias relacionadas con problemas de seguridad, en su mayoría vinculadas a la falta de validación de tokens anti-CSRF en métodos clave del controlador OperacionesPDFController. También se detectan problemas de configuración y advertencias sobre validaciones insuficientes, clasificadas con niveles de gravedad baja a media.

7.1. Tecnología de información

SonarQube : Es una herramienta de análisis estático de código que permite revisar automáticamente la calidad del código fuente de la aplicación. SonarQube analiza la base de código y genera informes sobre posibles errores, vulnerabilidades, deuda técnica, código duplicado y áreas donde se pueden aplicar mejoras. Además, realiza un seguimiento continuo de la calidad del código, integrándose con herramientas de CI/CD como GitHub Actions.

Snyk : Snyk es una plataforma de seguridad que se enfoca en la gestión de vulnerabilidades en las dependencias y bibliotecas de código abierto. Durante el análisis, se identifican posibles vulnerabilidades en las dependencias de terceros, lo que permite corregir problemas de seguridad antes de que afecten la producción.

ASP.NET Core y C# : El proyecto está desarrollado en ASP.NET Core utilizando el lenguaje de programación C#. Esto permite implementar una aplicación web robusta y escalable, con soporte para múltiples plataformas.

GitHub y GitHub Actions : Se utilizó GitHub como sistema de control de versiones y plataforma de colaboración. GitHub Actions se implementó para la integración continua, ejecutando automáticamente los análisis de calidad del código con SonarQube y los análisis de seguridad con Snyk en cada push al repositorio.

MariaDB/MySQL : Para la base de datos se utilizó MariaDB, que es una bifurcación de MySQL. Esta tecnología permitió gestionar el almacenamiento de datos de manera eficiente, con soporte para transacciones y consultas complejas.

7.2. Metodología, técnicas utilizadas Para el desarrollo de la aplicación, se adoptaron diversas metodologías y técnicas que permitieron optimizar el flujo de trabajo y asegurar la calidad del producto

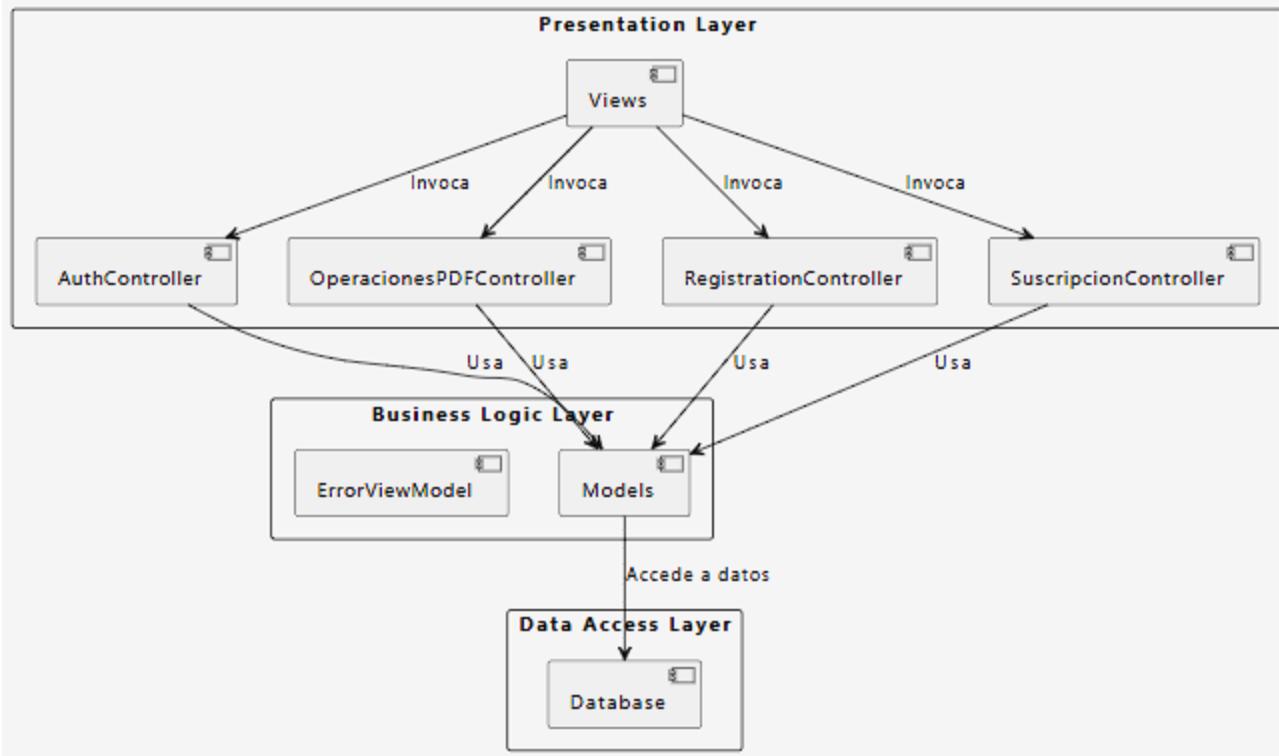
final. Entre las principales metodologías y técnicas utilizadas se destacan:

- **Desarrollo Ágil con GitHub Projects** : Se utilizó GitHub Projects para la planificación y gestión del trabajo en un entorno ágil. Las tareas se organizaron en tableros Kanban, permitiendo una visibilidad clara del progreso, asignación de tareas y establecimiento de prioridades. Cada tarea se vinculó con issues y pull request dentro del repositorio, facilitando la colaboración y el seguimiento del avance de las funcionalidades.
- **Integración Continua (CI)** : Se implementó un flujo de integración continua a través de GitHub Actions. Cada vez que se realizaba un cambio en el código (mediante un push o pull request), se ejecutaban pruebas automáticas, análisis de calidad con SonarQube y análisis de seguridad con Snyk, asegurando que el código estuviera siempre en condiciones óptimas para ser fusionado con la rama principal.
 - **Análisis Estático de Código** :
 - **SonarQube** : Para garantizar la calidad del código, se realizaron análisis estáticos con SonarQube, que permitieron detectar errores, vulnerabilidades, duplicación de código y deuda técnica. Esto ayudó a mantener un código más limpio, eficiente y mantenable.
 - **Snyk** : Se utilizó Snyk para identificar vulnerabilidades en las dependencias de terceros, ofreciendo soluciones y actualizaciones a las bibliotecas vulnerables, mejorando la seguridad general de la aplicación.

8. Cronograma (personas, tiempo, otros recursos) Basado en las observaciones que la herramienta SonarQube les informará sobre la aplicación, a fin de reducir la deuda técnica, vulnerabilidades, fallas, etc. a 0.

9. Desarrollo de Solución de Mejora

9.1 Diagrama de Arquitectura de la aplicación



9.2. Diagrama de clases de la aplicación

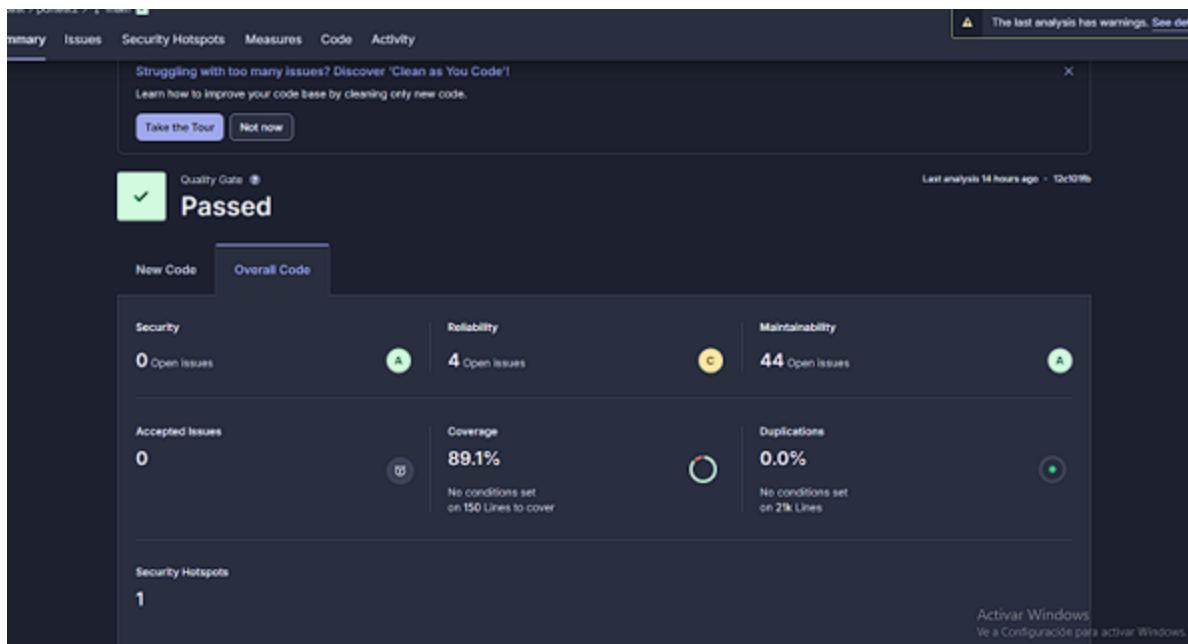


9.3. Métodos de pruebas implementadas para cubrir la aplicación.

- Informe de cobertura de pruebas

Actualmente

Pruebas Unitarias (cobertura de al menos 80% de código - los métodos más importantes)



La imagen nos muestra un informe de pruebas unitarias donde se ha logrado una cobertura del 89.1% en el código, superando el umbral recomendado del 80%. Esto refleja un buen nivel de pruebas aplicadas a los métodos más importantes del sistema, lo que garantiza la validación de su funcionalidad. El análisis también destaca que no se encontraron problemas de seguridad ni duplicaciones de código, lo que contribuye a la mantenibilidad y confiabilidad del proyecto.

This screenshot displays a list of four code review issues from a tool like SonarQube or similar. Each issue is presented in a card format with the following details:

- PROYECTO_PDF/NegocioPDF/Repositories/IUsuarioRepository.cs**: An issue titled "Move 'IUsuarioRepository' into a named namespace." is marked as a Reliability bug (severity Major). It was created by Mario Antonio Flores Ramos and assigned to him. The issue has 5min effort and was created 4 days ago. It is associated with the 'Adaptability' rule.
- PROYECTO_PDF/PROYECTOPDF/Views/Registration/Registrarse.cshtml**: Three identical issues under the "Intentionality" rule are listed. Each issue states: "A form label must be associated with a control." They are marked as Reliability bugs (severity Minor). All three were created by Mario Antonio Flores Ramos and assigned to him, with 5min effort and created 4 days ago. These issues are associated with the 'Intentionality' rule.

At the bottom of the list, it says "4 of 4 shown".

The screenshot shows a code review interface. At the top, it says "Move 'IUsuarioRepository' into a named namespace." with a warning icon. Below that, it says "Types should be defined in named namespaces csharpsquid:S3903". The "Software qualities impacted" section shows "Reliability" with a warning icon. At the bottom, there are tabs for "Where is the issue?", "Why is this an issue?", and "Activity", along with a "Open in IDE" button.

```

1  nf2018_
2      using System.Collections.Generic;
3      using NegocioPDF.Models;
4
5      public interface IUsuarioRepository
6  {
7          Usuario Login(string correo, string password);
8          void RegistrarUsuario(Usuario usuario);
9          IEnumerable<Usuario> ObtenerUsuarios();
10         Usuario ObtenerUsuarioPorId(int idUsuario);
11     }
12

```

Move 'IUsuarioRepository' into a named namespace.

The screenshot shows the Visual Studio Solution Explorer. On the left, under "EXPLORADOR", the project structure is visible, including "PROYECTO_PDF", "NegocioPDF", "Repositories", and "IUsuarioRepository.cs". The "IUsuarioRepository.cs" file is currently selected. On the right, the code editor shows the content of the "IUsuarioRepository.cs" file:

```

1  using System.Collections.Generic;
2  using NegocioPDF.Models;
3
4  namespace NegocioPDF.Repositories
5  {
6      public interface IUsuarioRepository
7  {
8      Usuario Login(string correo, string password);
9      void RegistrarUsuario(Usuario usuario);
10     IEnumerable<Usuario> ObtenerUsuarios();
11     Usuario ObtenerUsuarioPorId(int idUsuario);
12 }

```

Se detectó un problema en la organización del código, donde una interfaz no estaba ubicada en el espacio de nombres adecuado, lo que afectaba la mantenibilidad del proyecto. Para solucionarlo, se movió la interfaz a un espacio de nombres más apropiado, mejorando la estructura y modularidad del código. Esta corrección hace que el código sea más organizado y esté alineado con las buenas prácticas de desarrollo.

The screenshot shows the SonarQube interface with the following details:

- Project:** PROYECTO_PDF
- File:** main.cs
- Issue Type:** Consistency | Not conventioned
- Issue Description:** Declare types in namespaces (roslyn:CA1050_external_relyin-CA1050)
- Software Qualities Impacted:** Maintainability
- Status:** Open
- Assigned To:** Mario Antonio Flores Ramos
- Tags:** No tags
- Line Affected:** L8
- Effort:** 0 min
- Introduced:** 1 day ago
- Open in IDE:** Button

The code snippet shown is from `PROYECTO_PDF/.../NegocioPDF.Tests/DetalleSuscripcionRepositoryTests.cs`:

```

1  #nullable enable
2  using Microsoft.EntityFrameworkCore;
3  using NegocioPDF.Data;
4  using NegocioPDF.Models;
5  using NegocioPDF.Repositories;
6  namespace NegocioPDF.Tests
7  {
8      [TestFixture]
9      public class DetalleSuscripcionRepositoryTests
10     {
11         private PDFSolutionsContext _context;
12         private IDetalleSuscripcionRepository _repository;
13
14         [SetUp]
15         public void Setup()
16         {
17             var options = new DbContextOptionsBuilder<PDFSolutionsContext>()
18                 .UseInMemoryDatabase(databaseName: "TestDB_" + Guid.NewGuid().ToString())
19                 .Options;
20
21             _context = new PDFSolutionsContext(options);
22             _repository = new DetalleSuscripcionRepository(_context);
23
24             // Datos de prueba
25             var usuario = new Usuario
26             {
27                 Id = 1,
28                 Nombre = "Test User",
29                 Correo = "test@test.com",
30                 Password = "password123"
31             };
32             _context.Usuarios.Add(usuario);
33
34             var suscripcion = new DetalleSuscripcion
35             {
36                 UsuarioId = 1,
37                 ...
38             };
39         }
40     }
41 }

```

A red arrow points to the line `public class DetalleSuscripcionRepositoryTests`, indicating the issue.

The screenshot shows the code editor with the following changes made to the file `DetalleSuscripcionRepositoryTests.cs`:

```

1  #nullable enable
2  using NegocioPDF.Data;
3  using NegocioPDF.Models;
4  using NegocioPDF.Repositories;
5  namespace NegocioPDF.Tests
6  {
7      [TestFixture]
8      public class DetalleSuscripcionRepositoryTests
9      {
10         private PDFSolutionsContext _context;
11         private IDetalleSuscripcionRepository _repository;
12
13         [SetUp]
14         public void Setup()
15         {
16             var options = new DbContextOptionsBuilder<PDFSolutionsContext>()
17                 .UseInMemoryDatabase(databaseName: "TestDB_" + Guid.NewGuid().ToString())
18                 .Options;
19
20             _context = new PDFSolutionsContext(options);
21             _repository = new DetalleSuscripcionRepository(_context);
22
23             // Datos de prueba
24             var usuario = new Usuario
25             {
26                 Id = 1,
27                 Nombre = "Test User",
28                 Correo = "test@test.com",
29                 Password = "password123"
30             };
31             _context.Usuarios.Add(usuario);
32
33             var suscripcion = new DetalleSuscripcion
34             {
35                 UsuarioId = 1,
36                 ...
37             };
38         }
39     }
40 }

```

A red arrow points to the line `using NegocioPDF.Repositories;`, indicating the addition of the namespace declaration.

Se identificó un problema de consistencia en la organización de las pruebas, ya que la clase `DetalleSuscripcionRepositoryTests` no estaba correctamente declarada en un namespace adecuado, lo que afectaba la claridad y mantenibilidad del código. Para solucionar esto, se corrigió el problema ubicando la clase en el espacio de nombres correspondiente, `NegocioPDF.Tests`, y se configuraron los

datos de prueba y las dependencias con un contexto en memoria. Esta mejora asegura que las pruebas sean más modulares y sigan las mejores prácticas para mantener una estructura clara y bien organizada en el proyecto.

```

    public class UsuarioRepository : IUsuarioRepository
    {
        private readonly DbContext _context;
        private readonly IMapper _mapper;

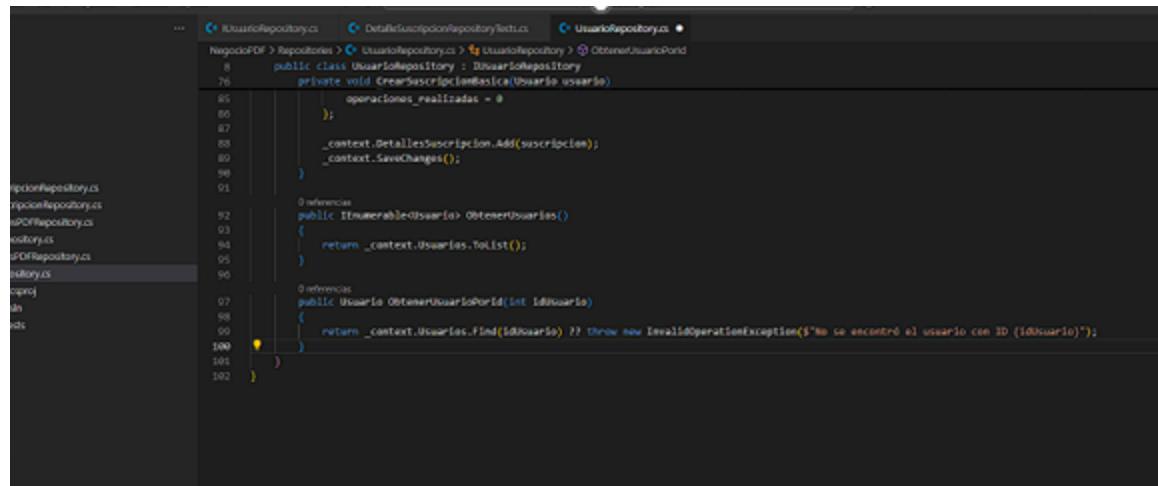
        public UsuarioRepository(DbContext context, IMapper mapper)
        {
            _context = context;
            _mapper = mapper;
        }

        public void Login(string correo, string password)
        {
            var usuario = _context.Usuarios.FirstOrDefault(u => u.Correo == correo & u.Password == password);
            if (usuario != null)
            {
                _context.SaveChanges();
            }
        }

        public IEnumerable<Usuario> ObtenerUsuarios()
        {
            return _context.Usuarios.ToList();
        }

        public Usuario? ObtenerUsuarioPorId(int idUsuario)
        {
            return _context.Usuarios.Find(idUsuario);
        }
    }

```



```

    public class UsuarioRepository : IUsuarioRepository
    {
        private readonly DbContext _context;
        private readonly IMapper _mapper;

        public UsuarioRepository(DbContext context, IMapper mapper)
        {
            _context = context;
            _mapper = mapper;
        }

        public void Login(string correo, string password)
        {
            var usuario = _context.Usuarios.FirstOrDefault(u => u.Correo == correo & u.Password == password);
            if (usuario != null)
            {
                _context.SaveChanges();
            }
        }

        public IEnumerable<Usuario> ObtenerUsuarios()
        {
            return _context.Usuarios.ToList();
        }

        public Usuario? ObtenerUsuarioPorId(int idUsuario)
        {
            return _context.Usuarios.Find(idUsuario);
        }
    }

```

El análisis estático detectó un problema de anulabilidad en el método `ObtenerUsuarioPorId` dentro del repositorio `UsuarioRepository`, ya que el tipo de retorno no manejaba adecuadamente los valores nulos. Para solucionarlo, se corrigió el método agregando una excepción `InvalidOperationException` en caso de que no se encuentre el usuario correspondiente. Esta mejora garantiza que el método maneje correctamente los casos sin coincidencias, lo que mejora la confiabilidad y claridad del código.

SonarQube interface showing a code review comment:

Consistency | Not identifiable

Rename class 'OperacionesPDFRepository' to match pascal case naming rules, consider using 'OperacionesPdfRepository'. ↗

Types should be named in PascalCase csharpsquid:S101

Software qualities impacted: **Maintainability** ↗

Open ▾ Mario Antonio Flores Ramos ▾ Code Smell Minor

Where is the issue? Why is this an issue? Activity More info Open in IDE

```

4     using System;
5     using System.Collections.Generic;
6     using System.Linq;
7     using System.Threading.Tasks;
8
9     namespace NegocioPDF.Repositories
10    {
11        public class OperacionesPDFRepository : IOperacionesPDFRepository
12        {
13            private readonly PDFSolutionsContext _context;
14
15            public OperacionesPDFRepository(PDFSolutionsContext context)
16            {
17                _context = context;
18            }
19
20            public bool RegistrarOperacionPDF(int usuarioId, string tipoOperacion)
21            {
22                try
23                {
24                    var suscripcion = _context.DetallesSuscripcion
25                        .FirstOrDefault(d => d.UsuarioId == usuarioId);
26

```

Activar Windows
Usa Configuración para más información

NegocioPDF > Repositories > **C# OperacionesPDFRepository.cs > operacionesPdfRepository > RegistrarOperacionPDF**

```

1  using Microsoft.EntityFrameworkCore;
2  using NegocioPDF.Data;
3  using NegocioPDF.Models;
4  using System;
5  using System.Collections.Generic;
6  using System.Linq;
7  using System.Threading.Tasks;
8
9  namespace NegocioPDF.Repositories
10 {
11     1 referencia
12     public class operacionesPdfRepository : IOperacionesPDFRepository
13     {
14         8 referencias
15         private readonly PDFSolutionsContext _context;
16
17         0 referencias
18         public operacionesPdfRepository(PDFSolutionsContext context)
19         {
20             _context = context;
21         }
22
23         0 referencias
24         public bool RegistrarOperacionPDF(int usuarioId, string tipoOperacion)
25         {
26             try
27             {
28                 var suscripcion = _context.DetallesSuscripcion
29                     .FirstOrDefault(d => d.UsuarioId == usuarioId);
30
31                 if (suscripcion != null)
32                 {
33                     suscripcion.TipoOperacion = tipoOperacion;
34                     _context.SaveChanges();
35                 }
36             }
37             catch (Exception ex)
38             {
39                 throw new InvalidOperationException(ex.Message);
40             }
41         }
42
43         public void EliminarOperacionPDF(int usuarioId)
44         {
45             try
46             {
47                 var suscripcion = _context.DetallesSuscripcion
48                     .FirstOrDefault(d => d.UsuarioId == usuarioId);
49
50                 if (suscripcion != null)
51                 {
52                     _context.Remove(suscripcion);
53                     _context.SaveChanges();
54                 }
55             }
56             catch (Exception ex)
57             {
58                 throw new InvalidOperationException(ex.Message);
59             }
60         }
61
62         public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion)
63         {
64             try
65             {
66                 var suscripcion = _context.DetallesSuscripcion
67                     .FirstOrDefault(d => d.UsuarioId == usuarioId);
68
69                 if (suscripcion != null)
70                 {
71                     suscripcion.TipoOperacion = tipoOperacion;
72                     _context.SaveChanges();
73                 }
74             }
75             catch (Exception ex)
76             {
77                 throw new InvalidOperationException(ex.Message);
78             }
79         }
80
81         public void DesactivarOperacionPDF(int usuarioId)
82         {
83             try
84             {
85                 var suscripcion = _context.DetallesSuscripcion
86                     .FirstOrDefault(d => d.UsuarioId == usuarioId);
87
88                 if (suscripcion != null)
89                 {
90                     suscripcion.Desactivado = true;
91                     _context.SaveChanges();
92                 }
93             }
94             catch (Exception ex)
95             {
96                 throw new InvalidOperationException(ex.Message);
97             }
98         }
99
100        public void ActivarOperacionPDF(int usuarioId)
101        {
102            try
103            {
104                var suscripcion = _context.DetallesSuscripcion
105                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
106
107                if (suscripcion != null)
108                {
109                    suscripcion.Desactivado = false;
110                    _context.SaveChanges();
111                }
112            }
113            catch (Exception ex)
114            {
115                throw new InvalidOperationException(ex.Message);
116            }
117        }
118
119        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion)
120        {
121            try
122            {
123                var suscripcion = _context.DetallesSuscripcion
124                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
125
126                if (suscripcion != null)
127                {
128                    _context.Remove(suscripcion);
129                    _context.SaveChanges();
130                }
131            }
132            catch (Exception ex)
133            {
134                throw new InvalidOperationException(ex.Message);
135            }
136        }
137
138        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion)
139        {
140            try
141            {
142                var suscripcion = _context.DetallesSuscripcion
143                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
144
145                if (suscripcion != null)
146                {
147                    suscripcion.TipoOperacion = tipoOperacion;
148                    suscripcion.Descripcion = descripcion;
149                    _context.SaveChanges();
150                }
151            }
152            catch (Exception ex)
153            {
154                throw new InvalidOperationException(ex.Message);
155            }
156        }
157
158        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion)
159        {
160            try
161            {
162                var suscripcion = _context.DetallesSuscripcion
163                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
164
165                if (suscripcion != null)
166                {
167                    suscripcion.Desactivado = true;
168                    _context.SaveChanges();
169                }
170            }
171            catch (Exception ex)
172            {
173                throw new InvalidOperationException(ex.Message);
174            }
175        }
176
177        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion)
178        {
179            try
180            {
181                var suscripcion = _context.DetallesSuscripcion
182                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
183
184                if (suscripcion != null)
185                {
186                    suscripcion.Desactivado = false;
187                    suscripcion.TipoOperacion = tipoOperacion;
188                    suscripcion.Descripcion = descripcion;
189                    _context.SaveChanges();
190                }
191            }
192            catch (Exception ex)
193            {
194                throw new InvalidOperationException(ex.Message);
195            }
196        }
197
198        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion)
199        {
200            try
201            {
202                var suscripcion = _context.DetallesSuscripcion
203                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
204
205                if (suscripcion != null)
206                {
207                    _context.Remove(suscripcion);
208                    _context.SaveChanges();
209                }
210            }
211            catch (Exception ex)
212            {
213                throw new InvalidOperationException(ex.Message);
214            }
215        }
216
217        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion)
218        {
219            try
220            {
221                var suscripcion = _context.DetallesSuscripcion
222                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
223
224                if (suscripcion != null)
225                {
226                    suscripcion.Desactivado = false;
227                    suscripcion.TipoOperacion = tipoOperacion;
228                    suscripcion.Descripcion = descripcion;
229                    _context.SaveChanges();
230                }
231            }
232            catch (Exception ex)
233            {
234                throw new InvalidOperationException(ex.Message);
235            }
236        }
237
238        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion)
239        {
240            try
241            {
242                var suscripcion = _context.DetallesSuscripcion
243                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
244
245                if (suscripcion != null)
246                {
247                    suscripcion.Desactivado = true;
248                    _context.SaveChanges();
249                }
250            }
251            catch (Exception ex)
252            {
253                throw new InvalidOperationException(ex.Message);
254            }
255        }
256
257        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion)
258        {
259            try
260            {
261                var suscripcion = _context.DetallesSuscripcion
262                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
263
264                if (suscripcion != null)
265                {
266                    suscripcion.Desactivado = false;
267                    suscripcion.TipoOperacion = tipoOperacion;
268                    suscripcion.Descripcion = descripcion;
269                    _context.SaveChanges();
270                }
271            }
272            catch (Exception ex)
273            {
274                throw new InvalidOperationException(ex.Message);
275            }
276        }
277
278        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
279        {
280            try
281            {
282                var suscripcion = _context.DetallesSuscripcion
283                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
284
285                if (suscripcion != null)
286                {
287                    _context.Remove(suscripcion);
288                    _context.SaveChanges();
289                }
290            }
291            catch (Exception ex)
292            {
293                throw new InvalidOperationException(ex.Message);
294            }
295        }
296
297        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
298        {
299            try
300            {
301                var suscripcion = _context.DetallesSuscripcion
302                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
303
304                if (suscripcion != null)
305                {
306                    suscripcion.Desactivado = false;
307                    suscripcion.TipoOperacion = tipoOperacion;
308                    suscripcion.Descripcion = descripcion;
309                    suscripcion.MotivoDesactivacion = motivo;
310                    _context.SaveChanges();
311                }
312            }
313            catch (Exception ex)
314            {
315                throw new InvalidOperationException(ex.Message);
316            }
317        }
318
319        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
320        {
321            try
322            {
323                var suscripcion = _context.DetallesSuscripcion
324                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
325
326                if (suscripcion != null)
327                {
328                    suscripcion.Desactivado = true;
329                    _context.SaveChanges();
330                }
331            }
332            catch (Exception ex)
333            {
334                throw new InvalidOperationException(ex.Message);
335            }
336        }
337
338        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
339        {
340            try
341            {
342                var suscripcion = _context.DetallesSuscripcion
343                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
344
345                if (suscripcion != null)
346                {
347                    suscripcion.Desactivado = false;
348                    suscripcion.TipoOperacion = tipoOperacion;
349                    suscripcion.Descripcion = descripcion;
350                    suscripcion.MotivoDesactivacion = motivo;
351                    _context.SaveChanges();
352                }
353            }
354            catch (Exception ex)
355            {
356                throw new InvalidOperationException(ex.Message);
357            }
358        }
359
360        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
361        {
362            try
363            {
364                var suscripcion = _context.DetallesSuscripcion
365                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
366
367                if (suscripcion != null)
368                {
369                    _context.Remove(suscripcion);
370                    _context.SaveChanges();
371                }
372            }
373            catch (Exception ex)
374            {
375                throw new InvalidOperationException(ex.Message);
376            }
377        }
378
379        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
380        {
381            try
382            {
383                var suscripcion = _context.DetallesSuscripcion
384                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
385
386                if (suscripcion != null)
387                {
388                    suscripcion.Desactivado = false;
389                    suscripcion.TipoOperacion = tipoOperacion;
390                    suscripcion.Descripcion = descripcion;
391                    suscripcion.MotivoDesactivacion = motivo;
392                    _context.SaveChanges();
393                }
394            }
395            catch (Exception ex)
396            {
397                throw new InvalidOperationException(ex.Message);
398            }
399        }
400
401        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
402        {
403            try
404            {
405                var suscripcion = _context.DetallesSuscripcion
406                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
407
408                if (suscripcion != null)
409                {
410                    suscripcion.Desactivado = true;
411                    _context.SaveChanges();
412                }
413            }
414            catch (Exception ex)
415            {
416                throw new InvalidOperationException(ex.Message);
417            }
418        }
419
420        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo)
421        {
422            try
423            {
424                var suscripcion = _context.DetallesSuscripcion
425                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
426
427                if (suscripcion != null)
428                {
429                    suscripcion.Desactivado = false;
430                    suscripcion.TipoOperacion = tipoOperacion;
431                    suscripcion.Descripcion = descripcion;
432                    suscripcion.MotivoDesactivacion = motivo;
433                    _context.SaveChanges();
434                }
435            }
436            catch (Exception ex)
437            {
438                throw new InvalidOperationException(ex.Message);
439            }
440        }
441
442        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
443        {
444            try
445            {
446                var suscripcion = _context.DetallesSuscripcion
447                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
448
449                if (suscripcion != null)
450                {
451                    _context.Remove(suscripcion);
452                    _context.SaveChanges();
453                }
454            }
455            catch (Exception ex)
456            {
457                throw new InvalidOperationException(ex.Message);
458            }
459        }
460
461        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
462        {
463            try
464            {
465                var suscripcion = _context.DetallesSuscripcion
466                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
467
468                if (suscripcion != null)
469                {
470                    suscripcion.Desactivado = false;
471                    suscripcion.TipoOperacion = tipoOperacion;
472                    suscripcion.Descripcion = descripcion;
473                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
474                    _context.SaveChanges();
475                }
476            }
477            catch (Exception ex)
478            {
479                throw new InvalidOperationException(ex.Message);
480            }
481        }
482
483        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
484        {
485            try
486            {
487                var suscripcion = _context.DetallesSuscripcion
488                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
489
490                if (suscripcion != null)
491                {
492                    suscripcion.Desactivado = true;
493                    _context.SaveChanges();
494                }
495            }
496            catch (Exception ex)
497            {
498                throw new InvalidOperationException(ex.Message);
499            }
500        }
501
502        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
503        {
504            try
505            {
506                var suscripcion = _context.DetallesSuscripcion
507                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
508
509                if (suscripcion != null)
510                {
511                    suscripcion.Desactivado = false;
512                    suscripcion.TipoOperacion = tipoOperacion;
513                    suscripcion.Descripcion = descripcion;
514                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
515                    _context.SaveChanges();
516                }
517            }
518            catch (Exception ex)
519            {
520                throw new InvalidOperationException(ex.Message);
521            }
522        }
523
524        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
525        {
526            try
527            {
528                var suscripcion = _context.DetallesSuscripcion
529                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
530
531                if (suscripcion != null)
532                {
533                    _context.Remove(suscripcion);
534                    _context.SaveChanges();
535                }
536            }
537            catch (Exception ex)
538            {
539                throw new InvalidOperationException(ex.Message);
540            }
541        }
542
543        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
544        {
545            try
546            {
547                var suscripcion = _context.DetallesSuscripcion
548                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
549
550                if (suscripcion != null)
551                {
552                    suscripcion.Desactivado = false;
553                    suscripcion.TipoOperacion = tipoOperacion;
554                    suscripcion.Descripcion = descripcion;
555                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
556                    _context.SaveChanges();
557                }
558            }
559            catch (Exception ex)
560            {
561                throw new InvalidOperationException(ex.Message);
562            }
563        }
564
565        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
566        {
567            try
568            {
569                var suscripcion = _context.DetallesSuscripcion
570                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
571
572                if (suscripcion != null)
573                {
574                    suscripcion.Desactivado = true;
575                    _context.SaveChanges();
576                }
577            }
578            catch (Exception ex)
579            {
580                throw new InvalidOperationException(ex.Message);
581            }
582        }
583
584        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
585        {
586            try
587            {
588                var suscripcion = _context.DetallesSuscripcion
589                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
590
591                if (suscripcion != null)
592                {
593                    suscripcion.Desactivado = false;
594                    suscripcion.TipoOperacion = tipoOperacion;
595                    suscripcion.Descripcion = descripcion;
596                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
597                    _context.SaveChanges();
598                }
599            }
600            catch (Exception ex)
601            {
602                throw new InvalidOperationException(ex.Message);
603            }
604        }
605
606        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
607        {
608            try
609            {
610                var suscripcion = _context.DetallesSuscripcion
611                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
612
613                if (suscripcion != null)
614                {
615                    _context.Remove(suscripcion);
616                    _context.SaveChanges();
617                }
618            }
619            catch (Exception ex)
620            {
621                throw new InvalidOperationException(ex.Message);
622            }
623        }
624
625        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
626        {
627            try
628            {
629                var suscripcion = _context.DetallesSuscripcion
630                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
631
632                if (suscripcion != null)
633                {
634                    suscripcion.Desactivado = false;
635                    suscripcion.TipoOperacion = tipoOperacion;
636                    suscripcion.Descripcion = descripcion;
637                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
638                    _context.SaveChanges();
639                }
640            }
641            catch (Exception ex)
642            {
643                throw new InvalidOperationException(ex.Message);
644            }
645        }
646
647        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
648        {
649            try
650            {
651                var suscripcion = _context.DetallesSuscripcion
652                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
653
654                if (suscripcion != null)
655                {
656                    suscripcion.Desactivado = true;
657                    _context.SaveChanges();
658                }
659            }
660            catch (Exception ex)
661            {
662                throw new InvalidOperationException(ex.Message);
663            }
664        }
665
666        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
667        {
668            try
669            {
670                var suscripcion = _context.DetallesSuscripcion
671                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
672
673                if (suscripcion != null)
674                {
675                    suscripcion.Desactivado = false;
676                    suscripcion.TipoOperacion = tipoOperacion;
677                    suscripcion.Descripcion = descripcion;
678                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
679                    _context.SaveChanges();
680                }
681            }
682            catch (Exception ex)
683            {
684                throw new InvalidOperationException(ex.Message);
685            }
686        }
687
688        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
689        {
690            try
691            {
692                var suscripcion = _context.DetallesSuscripcion
693                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
694
695                if (suscripcion != null)
696                {
697                    _context.Remove(suscripcion);
698                    _context.SaveChanges();
699                }
700            }
701            catch (Exception ex)
702            {
703                throw new InvalidOperationException(ex.Message);
704            }
705        }
706
707        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
708        {
709            try
710            {
711                var suscripcion = _context.DetallesSuscripcion
712                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
713
714                if (suscripcion != null)
715                {
716                    suscripcion.Desactivado = false;
717                    suscripcion.TipoOperacion = tipoOperacion;
718                    suscripcion.Descripcion = descripcion;
719                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
720                    _context.SaveChanges();
721                }
722            }
723            catch (Exception ex)
724            {
725                throw new InvalidOperationException(ex.Message);
726            }
727        }
728
729        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
730        {
731            try
732            {
733                var suscripcion = _context.DetallesSuscripcion
734                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
735
736                if (suscripcion != null)
737                {
738                    suscripcion.Desactivado = true;
739                    _context.SaveChanges();
740                }
741            }
742            catch (Exception ex)
743            {
744                throw new InvalidOperationException(ex.Message);
745            }
746        }
747
748        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
749        {
750            try
751            {
752                var suscripcion = _context.DetallesSuscripcion
753                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
754
755                if (suscripcion != null)
756                {
757                    suscripcion.Desactivado = false;
758                    suscripcion.TipoOperacion = tipoOperacion;
759                    suscripcion.Descripcion = descripcion;
760                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
761                    _context.SaveChanges();
762                }
763            }
764            catch (Exception ex)
765            {
766                throw new InvalidOperationException(ex.Message);
767            }
768        }
769
770        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
771        {
772            try
773            {
774                var suscripcion = _context.DetallesSuscripcion
775                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
776
777                if (suscripcion != null)
778                {
779                    _context.Remove(suscripcion);
780                    _context.SaveChanges();
781                }
782            }
783            catch (Exception ex)
784            {
785                throw new InvalidOperationException(ex.Message);
786            }
787        }
788
789        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
790        {
791            try
792            {
793                var suscripcion = _context.DetallesSuscripcion
794                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
795
796                if (suscripcion != null)
797                {
798                    suscripcion.Desactivado = false;
799                    suscripcion.TipoOperacion = tipoOperacion;
800                    suscripcion.Descripcion = descripcion;
801                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
802                    _context.SaveChanges();
803                }
804            }
805            catch (Exception ex)
806            {
807                throw new InvalidOperationException(ex.Message);
808            }
809        }
810
811        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
812        {
813            try
814            {
815                var suscripcion = _context.DetallesSuscripcion
816                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
817
818                if (suscripcion != null)
819                {
820                    suscripcion.Desactivado = true;
821                    _context.SaveChanges();
822                }
823            }
824            catch (Exception ex)
825            {
826                throw new InvalidOperationException(ex.Message);
827            }
828        }
829
830        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
831        {
832            try
833            {
834                var suscripcion = _context.DetallesSuscripcion
835                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
836
837                if (suscripcion != null)
838                {
839                    suscripcion.Desactivado = false;
840                    suscripcion.TipoOperacion = tipoOperacion;
841                    suscripcion.Descripcion = descripcion;
842                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
843                    _context.SaveChanges();
844                }
845            }
846            catch (Exception ex)
847            {
848                throw new InvalidOperationException(ex.Message);
849            }
850        }
851
852        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
853        {
854            try
855            {
856                var suscripcion = _context.DetallesSuscripcion
857                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
858
859                if (suscripcion != null)
860                {
861                    _context.Remove(suscripcion);
862                    _context.SaveChanges();
863                }
864            }
865            catch (Exception ex)
866            {
867                throw new InvalidOperationException(ex.Message);
868            }
869        }
870
871        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
872        {
873            try
874            {
875                var suscripcion = _context.DetallesSuscripcion
876                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
877
878                if (suscripcion != null)
879                {
880                    suscripcion.Desactivado = false;
881                    suscripcion.TipoOperacion = tipoOperacion;
882                    suscripcion.Descripcion = descripcion;
883                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
884                    _context.SaveChanges();
885                }
886            }
887            catch (Exception ex)
888            {
889                throw new InvalidOperationException(ex.Message);
890            }
891        }
892
893        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
894        {
895            try
896            {
897                var suscripcion = _context.DetallesSuscripcion
898                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
899
900                if (suscripcion != null)
901                {
902                    suscripcion.Desactivado = true;
903                    _context.SaveChanges();
904                }
905            }
906            catch (Exception ex)
907            {
908                throw new InvalidOperationException(ex.Message);
909            }
910        }
911
912        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
913        {
914            try
915            {
916                var suscripcion = _context.DetallesSuscripcion
917                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
918
919                if (suscripcion != null)
920                {
921                    suscripcion.Desactivado = false;
922                    suscripcion.TipoOperacion = tipoOperacion;
923                    suscripcion.Descripcion = descripcion;
924                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
925                    _context.SaveChanges();
926                }
927            }
928            catch (Exception ex)
929            {
930                throw new InvalidOperationException(ex.Message);
931            }
932        }
933
934        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
935        {
936            try
937            {
938                var suscripcion = _context.DetallesSuscripcion
939                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
940
941                if (suscripcion != null)
942                {
943                    _context.Remove(suscripcion);
944                    _context.SaveChanges();
945                }
946            }
947            catch (Exception ex)
948            {
949                throw new InvalidOperationException(ex.Message);
950            }
951        }
952
953        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
954        {
955            try
956            {
957                var suscripcion = _context.DetallesSuscripcion
958                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
959
960                if (suscripcion != null)
961                {
962                    suscripcion.Desactivado = false;
963                    suscripcion.TipoOperacion = tipoOperacion;
964                    suscripcion.Descripcion = descripcion;
965                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
966                    _context.SaveChanges();
967                }
968            }
969            catch (Exception ex)
970            {
971                throw new InvalidOperationException(ex.Message);
972            }
973        }
974
975        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
976        {
977            try
978            {
979                var suscripcion = _context.DetallesSuscripcion
980                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
981
982                if (suscripcion != null)
983                {
984                    suscripcion.Desactivado = true;
985                    _context.SaveChanges();
986                }
987            }
988            catch (Exception ex)
989            {
990                throw new InvalidOperationException(ex.Message);
991            }
992        }
993
994        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
995        {
996            try
997            {
998                var suscripcion = _context.DetallesSuscripcion
999                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1000
1001                if (suscripcion != null)
1002                {
1003                    suscripcion.Desactivado = false;
1004                    suscripcion.TipoOperacion = tipoOperacion;
1005                    suscripcion.Descripcion = descripcion;
1006                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
1007                    _context.SaveChanges();
1008                }
1009            }
1010            catch (Exception ex)
1011            {
1012                throw new InvalidOperationException(ex.Message);
1013            }
1014        }
1015
1016        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1017        {
1018            try
1019            {
1020                var suscripcion = _context.DetallesSuscripcion
1021                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1022
1023                if (suscripcion != null)
1024                {
1025                    _context.Remove(suscripcion);
1026                    _context.SaveChanges();
1027                }
1028            }
1029            catch (Exception ex)
1030            {
1031                throw new InvalidOperationException(ex.Message);
1032            }
1033        }
1034
1035        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1036        {
1037            try
1038            {
1039                var suscripcion = _context.DetallesSuscripcion
1040                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1041
1042                if (suscripcion != null)
1043                {
1044                    suscripcion.Desactivado = false;
1045                    suscripcion.TipoOperacion = tipoOperacion;
1046                    suscripcion.Descripcion = descripcion;
1047                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
1048                    _context.SaveChanges();
1049                }
1050            }
1051            catch (Exception ex)
1052            {
1053                throw new InvalidOperationException(ex.Message);
1054            }
1055        }
1056
1057        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1058        {
1059            try
1060            {
1061                var suscripcion = _context.DetallesSuscripcion
1062                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1063
1064                if (suscripcion != null)
1065                {
1066                    suscripcion.Desactivado = true;
1067                    _context.SaveChanges();
1068                }
1069            }
1070            catch (Exception ex)
1071            {
1072                throw new InvalidOperationException(ex.Message);
1073            }
1074        }
1075
1076        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1077        {
1078            try
1079            {
1080                var suscripcion = _context.DetallesSuscripcion
1081                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1082
1083                if (suscripcion != null)
1084                {
1085                    suscripcion.Desactivado = false;
1086                    suscripcion.TipoOperacion = tipoOperacion;
1087                    suscripcion.Descripcion = descripcion;
1088                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
1089                    _context.SaveChanges();
1090                }
1091            }
1092            catch (Exception ex)
1093            {
1094                throw new InvalidOperationException(ex.Message);
1095            }
1096        }
1097
1098        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1099        {
1100            try
1101            {
1102                var suscripcion = _context.DetallesSuscripcion
1103                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1104
1105                if (suscripcion != null)
1106                {
1107                    _context.Remove(suscripcion);
1108                    _context.SaveChanges();
1109                }
1110            }
1111            catch (Exception ex)
1112            {
1113                throw new InvalidOperationException(ex.Message);
1114            }
1115        }
1116
1117        public void ActualizarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1118        {
1119            try
1120            {
1121                var suscripcion = _context.DetallesSuscripcion
1122                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1123
1124                if (suscripcion != null)
1125                {
1126                    suscripcion.Desactivado = false;
1127                    suscripcion.TipoOperacion = tipoOperacion;
1128                    suscripcion.Descripcion = descripcion;
1129                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
1130                    _context.SaveChanges();
1131                }
1132            }
1133            catch (Exception ex)
1134            {
1135                throw new InvalidOperationException(ex.Message);
1136            }
1137        }
1138
1139        public void DesactivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1140        {
1141            try
1142            {
1143                var suscripcion = _context.DetallesSuscripcion
1144                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1145
1146                if (suscripcion != null)
1147                {
1148                    suscripcion.Desactivado = true;
1149                    _context.SaveChanges();
1150                }
1151            }
1152            catch (Exception ex)
1153            {
1154                throw new InvalidOperationException(ex.Message);
1155            }
1156        }
1157
1158        public void ActivarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1159        {
1160            try
1161            {
1162                var suscripcion = _context.DetallesSuscripcion
1163                    .FirstOrDefault(d => d.UsuarioId == usuarioId);
1164
1165                if (suscripcion != null)
1166                {
1167                    suscripcion.Desactivado = false;
1168                    suscripcion.TipoOperacion = tipoOperacion;
1169                    suscripcion.Descripcion = descripcion;
1170                    suscripcion.MotivoDesactivacion = motivoDesactivacion;
1171                    _context.SaveChanges();
1172                }
1173            }
1174            catch (Exception ex)
1175            {
1176                throw new InvalidOperationException(ex.Message);
1177            }
1178        }
1179
1180        public void EliminarOperacionPDF(int usuarioId, string tipoOperacion, string descripcion, string motivo, string motivoDesactivacion)
1181        {
1182            try
11
```

The screenshot shows the SonarQube interface with the following details:

- Project:** PROYECTO_PDF
- Analysis Status:** The last analysis has warnings.
- Issues:** 32 / 36 issues
- Where:** The current view is on the "Where" tab.
- Code Snippet:**

```

22     .Included(d => d.Users)
23     .Where(d => d.UsuarioId == usuarioId)
24     .OrderByDescending(d => d.Id)
25     .FirstOrDefault();
26 }

27
28 public void ActualizarSuscripcion(DetalleSuscripcion suscripcion)
29 {
30     var suscripcionExistente = _context.DetallesSuscripcion
31     .FirstOrDefault(d => d.UsuarioId == suscripcion.UsuarioId);
32
33     if (suscripcionExistente != null)
34     {
35         // Actualizar los propiedades específicas de la nómina
36         suscripcionExistente.TipoSuscripcion = suscripcion.TipoSuscripcion;
37         suscripcionExistente.Fecha_Inicio = suscripcion.Fecha_Inicio;
38         suscripcionExistente.Fecha_Final = suscripcion.Fecha_Final;
39         suscripcionExistente.Precio = suscripcion.Precio;
40         suscripcionExistente.Operaciones_Realizadas = suscripcion.Operaciones_Realizadas;
41         // Guardar los cambios en la base de datos
42         _context.SaveChanges();
43     }
44     else
45     {
46         throw new InvalidOperationException("No se encontró una suscripción para actualizar.");
47     }
48 }
49 }
50 }
```
- Warning:** 'System.Exception' should not be thrown by user code.

The screenshot shows the Visual Studio code editor with the following details:

- Project:** PROYECTO_PDF
- File:** DetalleSuscripcionRepository.cs
- Method:** ActualizarSuscripcion
- Code:**

```

10     public class DetalleSuscripcionRepository : IDetalleSuscripcionRepository
11     {
12         public void ActualizarSuscripcion(DetalleSuscripcion suscripcion)
13         {
14             var suscripcionExistente = _context.DetallesSuscripcion
15             .FirstOrDefault(d => d.UsuarioId == suscripcion.UsuarioId);
16             suscripcionExistente.TipoSuscripcion = suscripcion.TipoSuscripcion;
17             suscripcionExistente.Fecha_Inicio = suscripcion.Fecha_Inicio;
18             suscripcionExistente.Fecha_Final = suscripcion.Fecha_Final;
19             suscripcionExistente.Precio = suscripcion.Precio;
20             suscripcionExistente.Operaciones_Realizadas = suscripcion.Operaciones_Realizadas;
21             // Guardar los cambios en la base de datos
22             _context.SaveChanges();
23         }
24     }
25 }
```

El análisis estático destaca un problema en el método `ActualizarSuscripcion`, donde se utiliza `System.Exception`, lo cual no es recomendable porque dificulta la identificación precisa de los errores. La corrección realizada consistió en reemplazar `System.Exception` con `InvalidOperationException`, una excepción más específica. Esto mejora la claridad y mantenibilidad del código, proporcionando un mensaje más claro cuando no se encuentra una suscripción para actualizar. Esta práctica sigue las mejores recomendaciones para el manejo de errores en aplicaciones robustas.

The screenshot shows a SonarQube analysis results page. A specific code smell has been identified:

- Consistency | Not conventional**
- Declare types in namespaces** (roslyn:CA1050 external Roslyn:CA1050)
- Software qualities impacted:** Maintainability
- Tags:** No tags
- Line affected:** L8
- Effort:** 0 min
- Introduced:** 1 day ago

The code snippet shown is from `PROYECTO_PDF/NegocioPDF.Tests/UsuarioRepositoryTests.cs`:

```

1 nf2018... using NUnit.Framework;
2 using Microsoft.EntityFrameworkCore;
3 using NegocioPDF.Data;
4 using NegocioPDF.Models;
5 using NegocioPDF.Repositories;
6
7 [TestFixture]
8 public class UsuarioRepositoryTests
9 {
10     private PDFSolutionsContext _context;
11
12     [Test]
13     public void TestMethod()
14     {
15         var options = new DbContextOptionsBuilder<PDFSolutionsContext>()
16             .UseInMemoryDatabase(databaseName: "TestDB_" + Guid.NewGuid().ToString())
17             .Options;
18
19         _context = new PDFSolutionsContext(options);
20         _repository = new UsuarioRepository(_context);
21
22         // Datos de prueba
23         var usuario = new Usuario
24         {
25             ...
26         };
    
```

A callout box highlights the line `public class UsuarioRepositoryTests` with the message **Declare types in namespaces**.

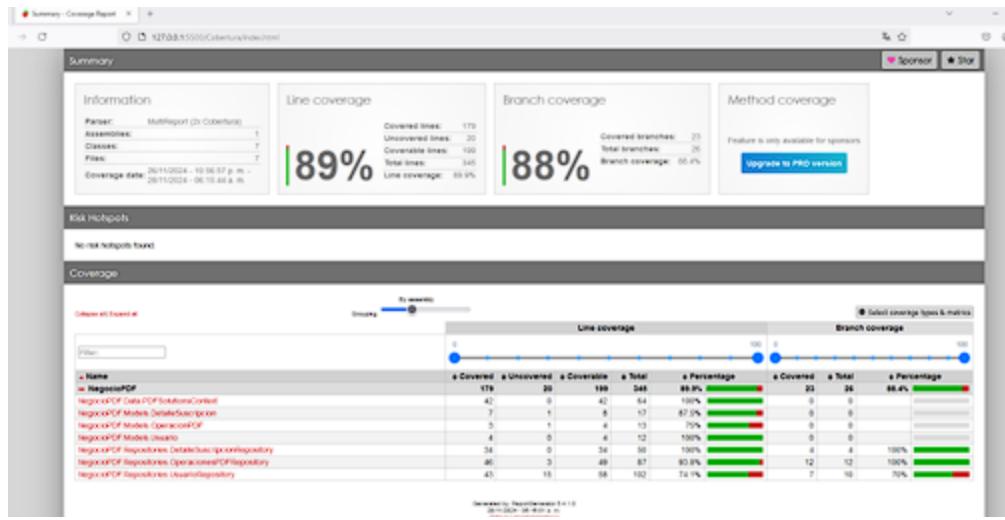
The screenshot shows the `UsuarioRepositoryTests.cs` file in Visual Studio Code. The line `public class UsuarioRepositoryTests` is highlighted with a red rectangle, indicating the change made during the refactoring.

```

1 using NUnit.Framework;
2 using Microsoft.EntityFrameworkCore;
3 using NegocioPDF.Data;
4 using NegocioPDF.Models;
5 using NegocioPDF.Repositories;
6
7 namespace NegocioPDF.Tests
8 {
9     [TestFixture]
10    public class UsuarioRepositoryTests
11    {
12        private PDFSolutionsContext _context;
13
14        [Test]
15        public void TestMethod()
16        {
17            var options = new DbContextOptionsBuilder<PDFSolutionsContext>()
18                .UseInMemoryDatabase(databaseName: "TestDB_" + Guid.NewGuid().ToString())
19                .Options;
20
21            _context = new PDFSolutionsContext(options);
22            _repository = new UsuarioRepository(_context);
23
24            // Datos de prueba
25            var usuario = new Usuario
26            {
27                ...
28            };
    
```

El análisis estático señala un problema de consistencia debido a la ausencia de un espacio de nombres definido, lo que afecta la mantenibilidad del código. La corrección realizada consistió en incluir el archivo dentro del namespace `NegocioPDF.Tests`, alineando la clase con las mejores prácticas de organización. Esto mejora la claridad del proyecto al agrupar correctamente las clases de prueba según su módulo correspondiente.

Pruebas de integración utilizando Mocks o Fake Classes



El informe destaca una cobertura del 89% en líneas de código y del 88% en ramas, lo que indica un alto nivel de pruebas realizadas sobre el código base. Esto incluye detalles de los módulos cubiertos, con porcentajes individuales para líneas y ramas, mostrando áreas bien probadas y algunas con margen de mejora.

- Reporte de Pruebas guiadas por el comportamiento (BDD Given When Then)

Pruebas de aceptación basadas en Desarrollo Guiado por el Comportamiento una por cada caso de uso o historia de usuario.

The screenshot shows a BDD test report for the 'Login de Usuario' feature. It includes a sidebar for 'Living Documentation' and 'Analytics', and a main panel for the 'Feature: Login de Usuario'. The feature description states: 'Como usuario del sistema Quiero poder iniciar sesión Para acceder a las funcionalidades del sistema'. The report lists three scenarios: 'Login exitoso', 'Login fallido con credenciales inválidas', and 'Login fallido con correo vacío'. Each scenario is broken down into 'Given', 'When', and 'Then' steps, each with a green checkmark indicating success.

NegocioPDF.Tests
generated Nov 28, 2024, 08:31 AM GMT-5

Living Documentation Analytics

Filter by Keyword Filter by Scenario Result Test results

- + -
- NegocioPDF.Tests
 - Features
 - > Login de Usuario
 - > Operaciones con PDFs
 - > Registro de Usuario

Feature: Operaciones con PDFs

Como usuario del sistema
Quiero poder realizar operaciones con PDFs
Según mi tipo de suscripción

Scenario: Usuario básico realiza operación dentro del límite 7ms

- Given un usuario con id 1 y suscripción "básico"
- When intento realizar una operación "conversion"
- Then la operación deberá realizarse correctamente

Scenario: Usuario básico excede límite de operaciones 97ms

- Given un usuario con id 1 y suscripción "básico"
- When intento realizar una operación "conversion"
- And intento realizar una operación "conversion"
- Then la operación deberá ser rechazada

NegocioPDF.Tests
generated Nov 28, 2024, 08:31 AM GMT-5

Living Documentation Analytics

Filter by Keyword Filter by Scenario Result Test results

- + -
- NegocioPDF.Tests
 - Features
 - > Login de Usuario
 - > Operaciones con PDFs
 - > Registro de Usuario

Feature: Registro de Usuario

Quiero poder registrarme
Para convertirme en usuario del sistema

Scenario: Registro exitoso 10ms

- Given un nuevo usuario con nombre "Nuevo Usuario", correo "nuevo@test.com" y contraseña "nuestrap123"
- When intento registrar el usuario
- Then el usuario deberá registrarse correctamente

Scenario: Registro fallido con correo existente 3ms

- Given un nuevo usuario con nombre "Usuario Duplicado", correo "test@test.com" y contraseña "pass123"
- When intento registrar el usuario
- Then deberá ver un mensaje de error de registro

Scenario: Registro fallido con datos incompletos 6ms

- Given un nuevo usuario con nombre "", correo "" y contraseña ""
- When intento registrar el usuario
- Then deberá ver un mensaje de error de registro

El informe muestra pruebas guiadas por el comportamiento (BDD) utilizando el formato Dado-Cuando-Entonces, específicamente para el caso de uso de inicio de sesión. Cada prueba representa un escenario distinto basado en una historia de usuario: un inicio de sesión exitoso, un intento fallido por credenciales inválidas y otro fallido por un correo vacío. Esto asegura que las funcionalidades clave se validan según las expectativas del usuario y los criterios de aceptación.

- Informe de pruebas mutantes

Pruebas mutantes para ver todas las pruebas posibles.

Informe - Pruebas con mutaciones

```

Stryker.NET

Version: 4.4.1

[06:21:28 INF] Analysis starting.
[06:21:28 INF] Analyzing 1 test project(s).
[06:21:34 INF] Found project D:\testando\proyecto-s1294-2024-11-02\Florest_salinas_ticahuanca\PROJECT0_PDF\VegocloudPDF.Tests\Tes...
[06:21:34 INF] Analysis complete.
[06:21:34 INF] Total mutants: 240. Found mutants: 113. Skipped mutants: 127. Coverage: 75.23% (V/V)
[06:21:35 INF] Building project VegocloudPDF.Tests.csproj using dotnet build VegocloudPDF.Tests.csproj (directory D:\testando\proyecto-s1294-2024-11-02\Florest_salinas_ticahuanca\PROJECT0_PDF\VegocloudPDF.Tests.)
[06:21:35 INF] Number of tests found: 76 for project D:\testando\proyecto-s1294-2024-11-02\Florest_salinas_ticahuanca\PROJECT0_PDF\VegocloudPDF\VegocloudPDF.csproj. Initial test run started.
[06:26:47 INF] 240 mutants created.
[06:26:47 INF] Capture mutant coverage using "CoverageBasedTest" mode.
Hint: By pressing "F5" open report on "c:\" the report will open automatically and update the report in real time.
[06:26:47 INF] 113 mutants are covered by at least one test. 127 mutants are not covered by any test.
[06:26:48 INF] 25 mutants got status "Killed". Reason: Not covered by any test.
[06:26:48 INF] 38 mutants got status "Ignored". Reason: Covered by block already covered filter.
[06:26:48 INF] 75 mutants got status "Ignored". Reason: Removed by exclude from code coverage filter.
[06:26:48 INF] 113 total mutants are skipped for the above mentioned reasons.
[06:26:48 INF] 189 total mutants will be tested.

58.72% | Testing mutant 64 / 189 | E:48 | S:36 | T:0 | -0h 2m | 06:26:48

Activar Windows
Visita Configuración para activar Windows. 06:26:48

```

```

Stryker.NET

Version: 4.4.1

[06:25:28 INF] Analysis starting.
[06:25:28 INF] Analyzing 1 test project(s).
[06:25:34 INF] Found project D:\testando\proyecto-s1294-2024-11-02\Florest_salinas_ticahuanca\PROJECT0_PDF\VegocloudPDF.Tests\Tes...
[06:25:34 INF] Analysis complete.
[06:25:34 INF] Total mutants: 240. Found mutants: 113. Skipped mutants: 127. Coverage: 75.23% (V/V)
[06:25:35 INF] Building project VegocloudPDF.Tests.csproj using dotnet build VegocloudPDF.Tests.csproj (directory D:\testando\proyecto-s1294-2024-11-02\Florest_salinas_ticahuanca\PROJECT0_PDF\VegocloudPDF.Tests.)
[06:25:35 INF] Number of tests found: 76 for project D:\testando\proyecto-s1294-2024-11-02\Florest_salinas_ticahuanca\PROJECT0_PDF\VegocloudPDF\VegocloudPDF.csproj. Initial test run started.
[06:26:47 INF] 240 mutants created.
[06:26:47 INF] Capture mutant coverage using "CoverageBasedTest" mode.
Hint: By pressing "F5" open report on "c:\" the report will open automatically and update the report in real time.
[06:26:47 INF] 113 mutants are covered by at least one test. 127 mutants are not covered by any test.
[06:26:48 INF] 25 mutants got status "Killed". Reason: Not covered by any test.
[06:26:48 INF] 38 mutants got status "Ignored". Reason: Covered by block already covered filter.
[06:26:48 INF] 75 mutants got status "Ignored". Reason: Removed by exclude from code coverage filter.
[06:26:48 INF] 113 total mutants are skipped for the above mentioned reasons.
[06:26:48 INF] 189 total mutants will be tested.

100.00% | Testing mutant 189 / 189 | E:42 | S:27 | T:0 | -0h 0m | 06:26:48

Killed: 27
Survived: 113
Timeout: 0

Your HTML report has been generated at:
D:\testando\report-s1294-2024-11-02\Florest_salinas_ticahuanca\PROJECT0_PDF\VegocloudPDF.Tests\StrykerOutput\2024-11-28-00-23-28\reports\mutation-report.html
You can open it in your browser of choice.
[06:29:20 INF] Time elapsed 00:00:57.655251
[06:29:26 INF] The final mutation score is 65.19 %.

Activar Windows
Visita Configuración para activar Windows. 06:29:26

```

El informe de Stryker.NET Report muestra los resultados de la prueba de mutantes:

File / Directory	Mutation Score Of total	Mutation Score Of covered	Killed	Survived	Timeout	No coverage	Skipped	Runtime errors	Compile errors	Detected	Undetected	Total
All files	65.19	75.23	82	27	0	25	105	0	3	82	82	242
Data	34.38	34.38	11	21	0	0	4	0	0	11	21	36
Migrations	0.00	0.00	0	0	0	54	77	0	0	0	14	91
Models	N/A	N/A	0	0	0	0	0	0	0	0	8	8
Repositories	85.68	92.28	71	6	0	11	26	0	3	71	17	115

El análisis indica qué partes del código están bien cubiertas por pruebas y cuáles necesitan mayor atención, ayudando a fortalecer el conjunto de pruebas.

Informe de Stryker.NET - Análisis de Pruebas Mutantes

Barra Superior Colorida

- Verde (82): Mutantes eliminados (¡Bueno!)
- Rojo (27): Mutantes sobrevivieron (¡Malo!)
- Naranja (25)**: Sin cobertura

Columnas principales

1. Puntuación de mutación

- Del total: 61,19% - Porcentaje general de mutantes eliminados
- De cubiertos: 75.23% - Porcentaje de mutantes eliminados en código cubierto por pruebas

2. Estado de los Mutantes

- Asesinado (82): Mutantes que tus pruebas detectaron y fallaron (¡Bueno!)
- Survived (27): Mutantes que tus pruebas no detectan (¡Malo!)
- Sin cobertura (25): Código sin pruebas unitarias
- Ignorados (105): Mutantes que se ignoraron

Análisis por Carpetas

- Datos/PDFSolutionsContext.cs: 34,38% de efectividad
- Migraciones: 0% (Normal, no se suelen probar)
- Modelos: N/A
- Repositorios: 80,68% (Bastante bueno)

Informe de pruebas de interfaz de usuario

Pruebas (al menos 3 completas) incluyen el video generado de forma automatizada por el framework

[vídeo_1.webm](#)

Se ingresa el correo y contraseña, iniciamos sesión y se muestra la interfaz de bienvenida.

[vídeo_2.webm](#)

Se aprecia como se ingresan los datos de nombre de usuario, correo y contraseña, le damos a registrarse y se nos muestra un mensaje de confirmación y se nos mostrará un botón para regresar al login.

[vídeo_3.webm](#)

Si se ingresa un correo contraseña o incorrectos se nos mostrará un mensaje de error

vídeo_4.webm

Iniciamos sesión, se nos mostrará la interfaz principal de bienvenida con su tipo de suscripción cantidad de operaciones realizadas y podremos tener la suscripción premium

Bibliografías:

- Semgrep | Página de inicio. (sf). Semgrep.
- Perro de datos. (2016, 14 de julio). Monitoreo de código DataDog | Perro de datos. Perro de datos.
- Snyk. (sf). Seguridad para desarrolladores | Snyk.

Enlace del vídeo: