

Universidad ORT

Facultad de Ingeniería.

Carrera Licenciatura en Sistemas.

Diseño de aplicaciones 1

Obligatorio 2

Matias Fontes - 248142

Renzo Matonti - 230829

Link GitHub: <https://github.com/ORT-DA1/Obligatorio1-DA1-Fontes-Matonti>

2021

Índice

Descripción general	3
Bugs Conocidos	3
Diseño	4
Reglas de negocios	4
Diagrama de Paquetes	5
Diagrama de Clases	6
Dominio	6
Excepciones	7
Controllers	7
Diseño Definido para Perfil (Profile Class)	8
Conexión UserInterface y Lógica de Negocio	9
Manejo de Excepciones	9
Pruebas	10
Unit Test	10
Casos de prueba	11
Casos de prueba Perfil	11
Casos de prueba de categorías	12
Casos de prueba de Contraseña	13
Casos de prueba de Tarjetas de crédito	14
Para mayor definición abrir archivo ubicado en: [Abrir archivo en Anexo - Casos de prueba]	15
Bibliografía	16
Anexo	17
Diagrama de paquetes	17
Diagrama de clases [PasswordManager]Domain	17
Diagrama de clases [PasswordManager]Exceptions	18
Diagrama de clases - [PasswordManager]Controllers	19
Diagrama de clases - [UserInterface]	20
Casos de prueba	20

Descripción general

Se desarrolló un gestor de contraseñas, el cual cumple con los siguientes requerimientos:

- Permite ingresar una clave a la aplicación, antes de hacerlo no se puede ver o modificar datos guardados.
- La aplicación permite registrar categorías, tarjetas de crédito y contraseñas.
- Se puede agregar combinaciones usuario~contraseña para sitios o aplicaciones específicas.
- Se puede guardar datos de tarjetas de crédito.
- Chequea los datos guardados en la aplicación no hayan aparecido en algún data breach.
- Guardar un historial de data breaches
- Permite chequear el nivel de seguridad de contraseñas guardadas, se dividen en 5 tipos (rojo, naranja, amarillo, verde claro y verde oscuro).

Bugs Conocidos

Diseño

Reglas de negocios

Al enfrentar el desafío de desarrollar una aplicación con funcionalidades de gestionar contraseñas, categorías y tarjetas de créditos fueron necesarios, consultar al cliente en este escenario los docentes por las restricciones aplicadas a la estructura de los datos necesarios para el cumplimiento de las funcionalidades planeadas.

Es así como se definieron las siguientes reglas de negocios:

De las categorías

Se debe poder ingresar categorías las cuales cuentan y son identificadas con un nombre, la cantidad de caracteres válidos del nombre se contempla entre 3 a 15 caracteres, haciendo

También para impedir Categorías de igual nombre, se define que el mismo sea Case Sensitive.

De las contraseñas

Se solicita poder almacenar la siguiente información:

Seleccionar 1 categoría de las previamente ingresadas en el sistema, este campo es obligatorio se impide la opción de agregar contraseñas al sistema si no se cuenta con al menos 1 categoría en el sistema.

Sitio, El sitio es un campo obligatorio que puede recibir un valor texto entre 3 y 25 caracteres de largo

Usuario, un campo obligatorio que toma valor texto entre 5 y 25 caracteres de largo.

Contraseña, es un campo de texto obligatorio debe tener un mínimo de 5 caracteres de largo y puede recibir un máximo de 25 caracteres.

Nota, es un campo no obligatorio que puede tomar un máximo de 250 caracteres.

Además se pide almacenar la última fecha de modificación la cual es calculada y almacenada por el sistema cada vez que se modifique algún campo de la contraseña, esta fecha se guarda en formato dd/MM/yyyy hh:mm:ss.

Se define que 2 contraseñas con el mismo par Usuario/Sitio son iguales y se impide la redundancia de datos de este tipo.

De las tarjetas de crédito

Se solicita poder almacenar tarjetas de crédito con la siguiente información:

Seleccionar 1 categoría de las previamente ingresadas en el sistema, este campo es obligatorio se impide la opción de agregar tarjetas de crédito al sistema si no se cuenta con al menos 1 categoría en el sistema.

Nombre, se necesita almacenar un nombre asociado a la tarjeta de crédito el cual puede tener un largo comprendido entre los 3 a 25 caracteres.

Tipo, un campo de texto obligatorio que puede tomar un largo límite entre 3 a 25 caracteres.

Código CCV, es un campo obligatorio que guarda un código numérico comprendido entre el 000 al 999.

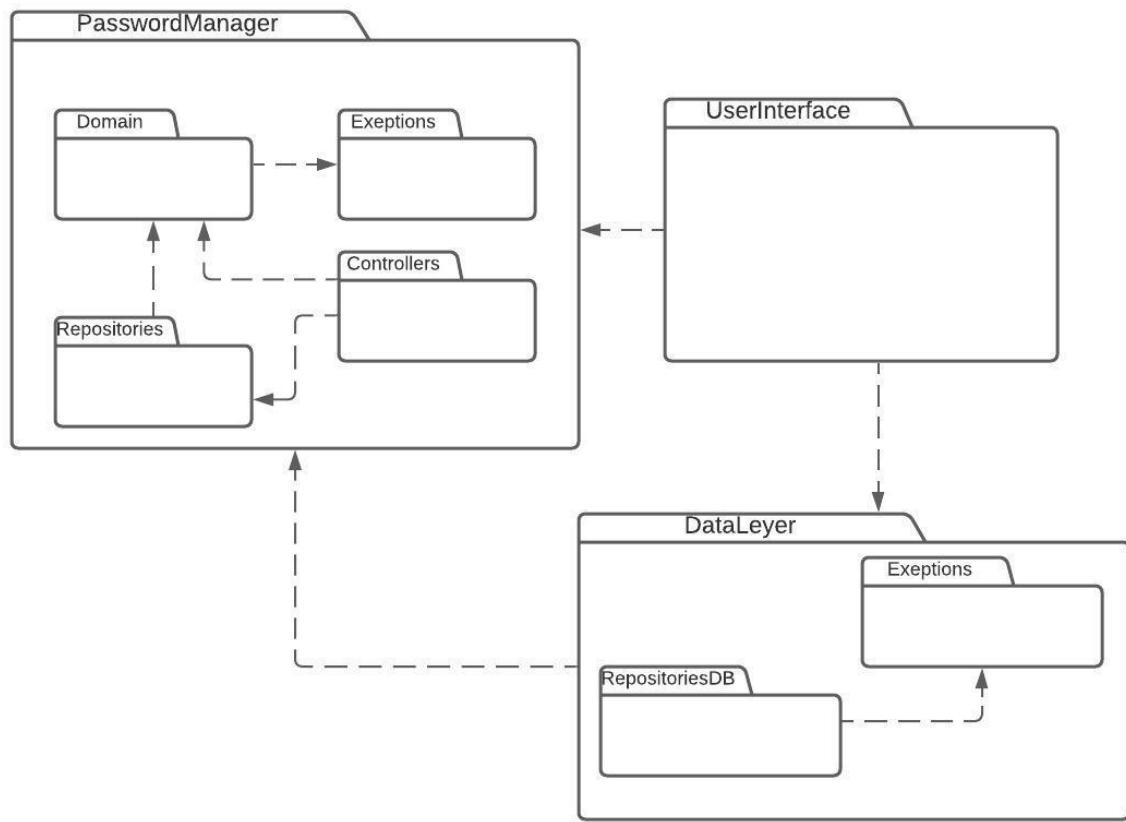
Número de tarjeta, Campo obligatorio el cual recibe 16 caracteres numéricos, los cuales pueden ser ingresados con espacios entre sí como el formato presentado en las tarjetas de crédito.

Fecha de Vencimiento, Campo obligatorio donde se selecciona la fecha de caducidad de una tarjeta de crédito.

Nota, Un campo de texto no obligatorio el cual no puede pasar los 250 caracteres de largo.

Diagrama de Paquetes

A continuación se muestra la organización de paquetes del sistema y sus dependencias.



Se desarrollaron 3 paquetes diferentes que dividen la implementación en capa lógica, capa de datos y Interfaz.

Interfaz

UserInterface el cual contiene definido los formularios y paneles por los cuales estará navegando el usuario.

Capa Lógica

La lógica de Negocio es el paquete PasswordManager el cual contiene 3 carpetas que separan la implementación de clases del dominio.

Domain

- Profile
- Password
- CreditCard
- Category
- IRepository
- ISearchable
- Data Breach
- GeneratePasswordSettings
- PasswordGenerator

Repositories

- CategoryRepository
- PasswordRepository
- CreditCardRepository
- DataBreachRepository

Exceptions, donde se definen las excepciones implementadas para la capa lógica.

Controllers, donde se ubican los controladores de los repositorios en memoria y el controlador de la clase Profile.

Capa de Datos

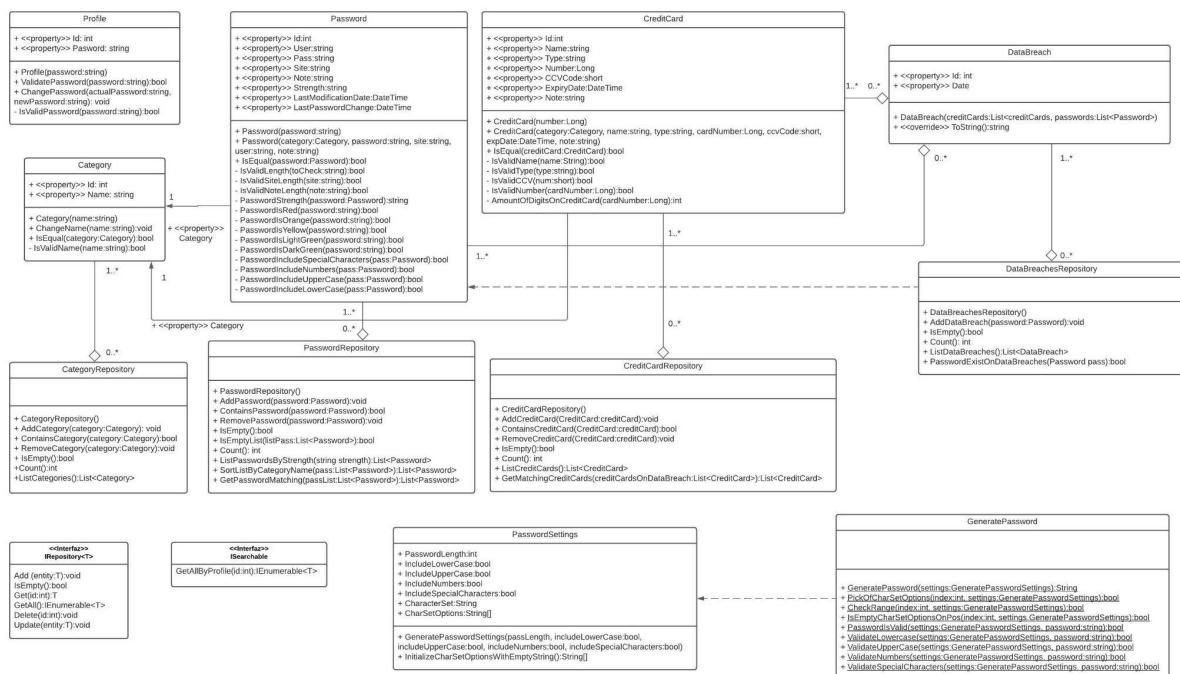
La capa de datos es el paquete PasswordManagerDataLeyer, en esta capa definimos objetos DTO que representan a las entidades de nuestro dominio (PasswordEntity, ProfileEntity, CategoryEntity, etc), junto a estos DTO[*] definimos Un contexto de base de datos (DbContext) y mediante el uso de la biblioteca Entity Framework 6[*] implementamos una migración de base de datos usando SQL Express y su IDE SQL Management Studio.

En este paquete también se definió una subcarpeta llamada RepositoriesDB, la cual contiene diferentes clases que implementan la interfaz Repository del dominio, esta interfaz define las operaciones CRUD básicas que cada repositorio debe implementar.

Diagrama de Clases

A continuación los diagramas de clases para los diferentes paquetes del sistema.

Dominio



Para mayor definición abrir archivo ubicado en: [Diagrama de clases \[PasswordManager\]Domain](#)

Excepciones

En el siguiente UML se muestra las dependencias entre el dominio y las excepciones implementadas

[\[ver Anexo Diagrama de clases PasswordManager - Exceptions\]](#)

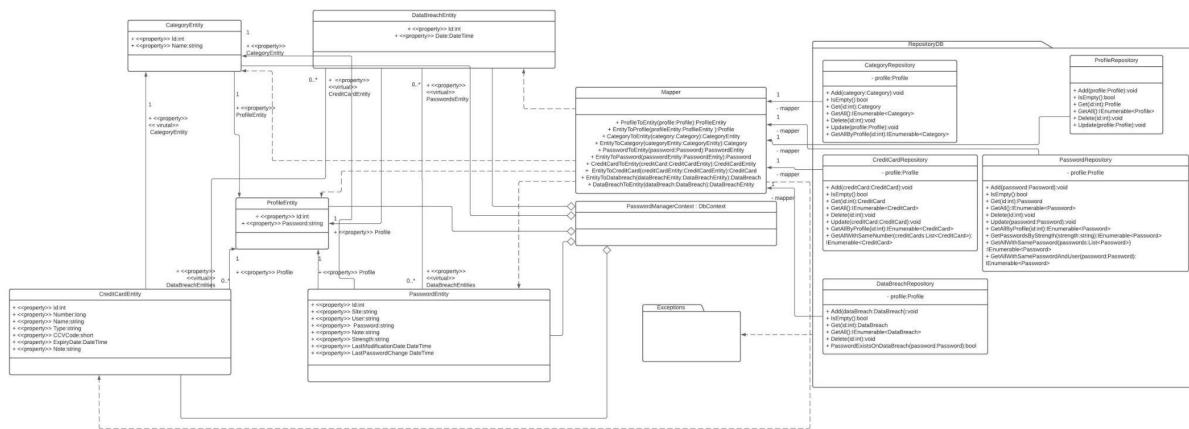
Controllers

El diagrama representa las relaciones entre los controladores y las clases de dominios.

[\[Anexo Diagrama de clases - PasswordManager.Controllers\]](#)

Se puede observar como los controladores (Controllers) cuentan con una relación de tipo asociación con las respectivas clases del dominio, esta implementación de controladores brinda una interfaz pública ajena a la implementación la cual será usada principalmente por la interfaz para comunicarse con los principales objetos y estructuras de datos del dominio, generando menor dependencia entre UI y Implementación

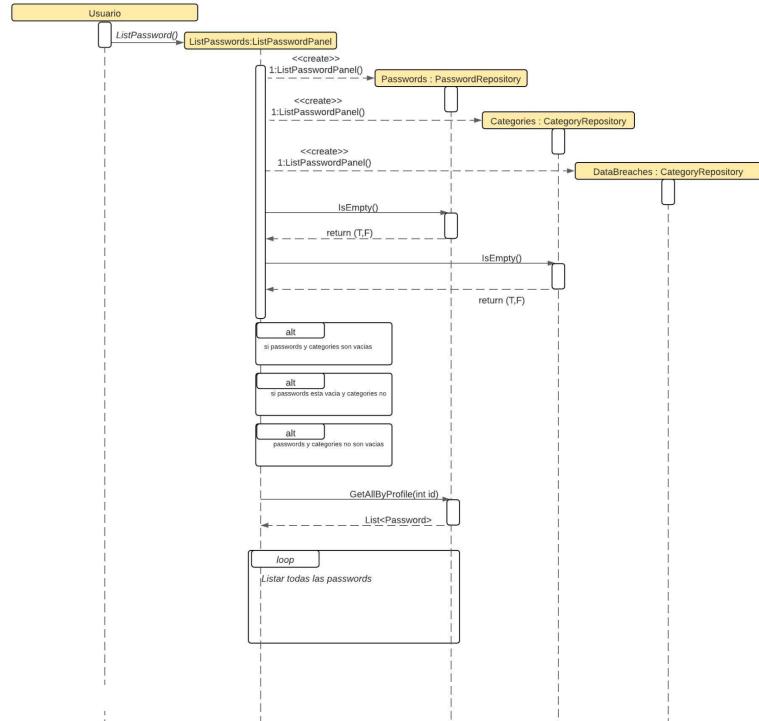
Password Manager Data Layer



Para mayor definición abrir archivo ubicado en: [\[Anexo Diagrama de clases - PasswordManager.Controllers\]](#)

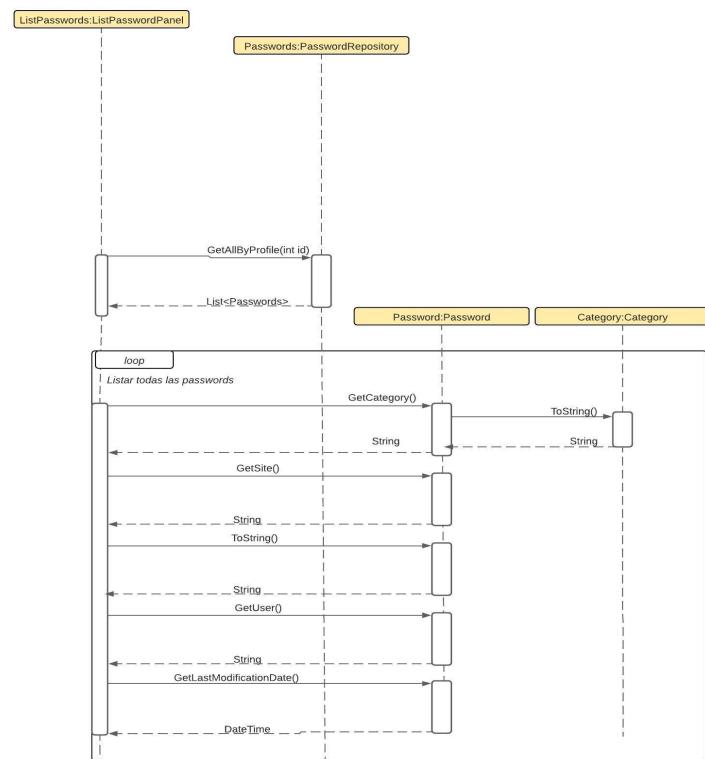
Diagrama de secuencia

List Password



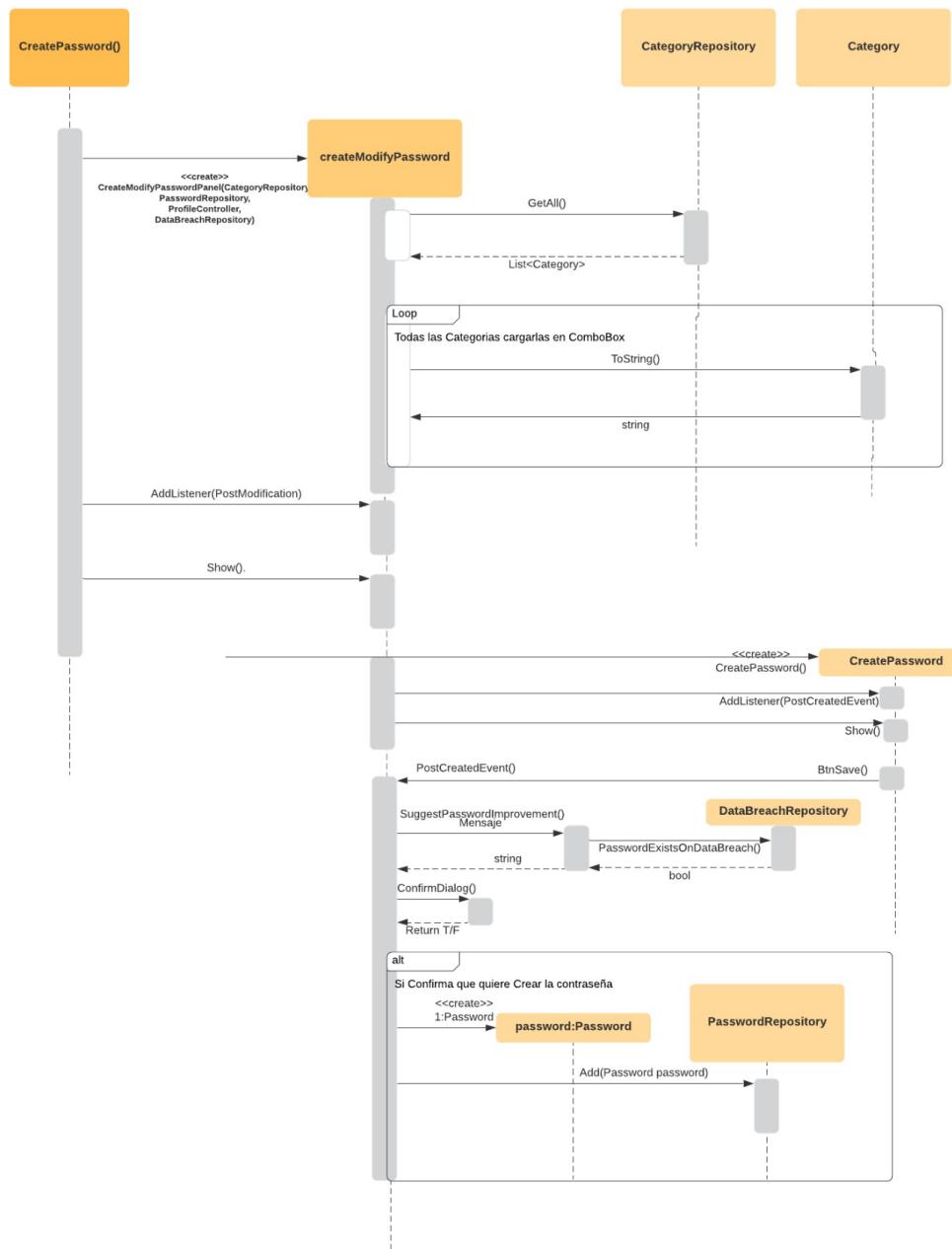
[\[Abrir imagen\]](#)

Continuación del diagrama anterior:



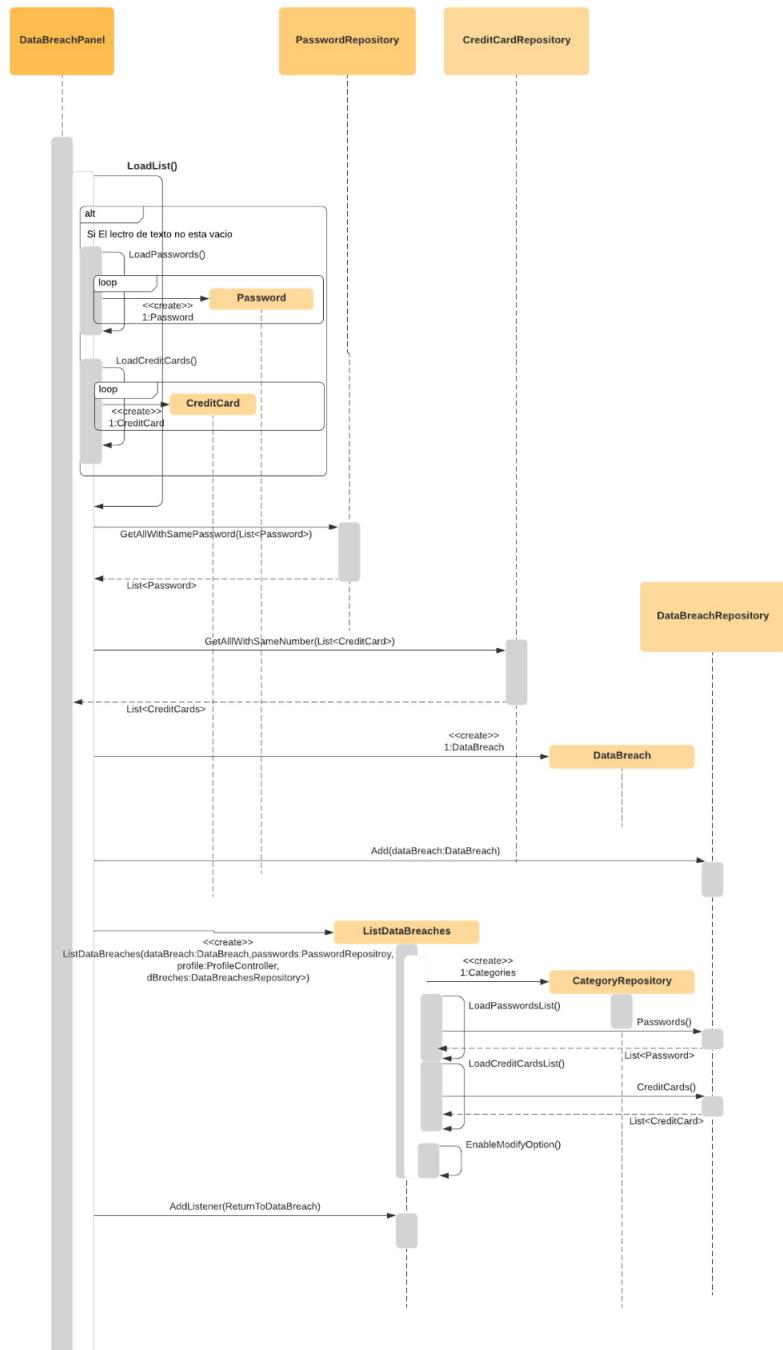
[\[Abrir imagen\]](#)

Create password



[\[Abrir imagen\]](#)

Verify data breach



[\[Abrir archivo\]](#)

Diagrama Entidad Relación

Para ver el archivo completo del diagrama entidad relación acceder en
[[Anexo Modelo entidad-relación](#)]

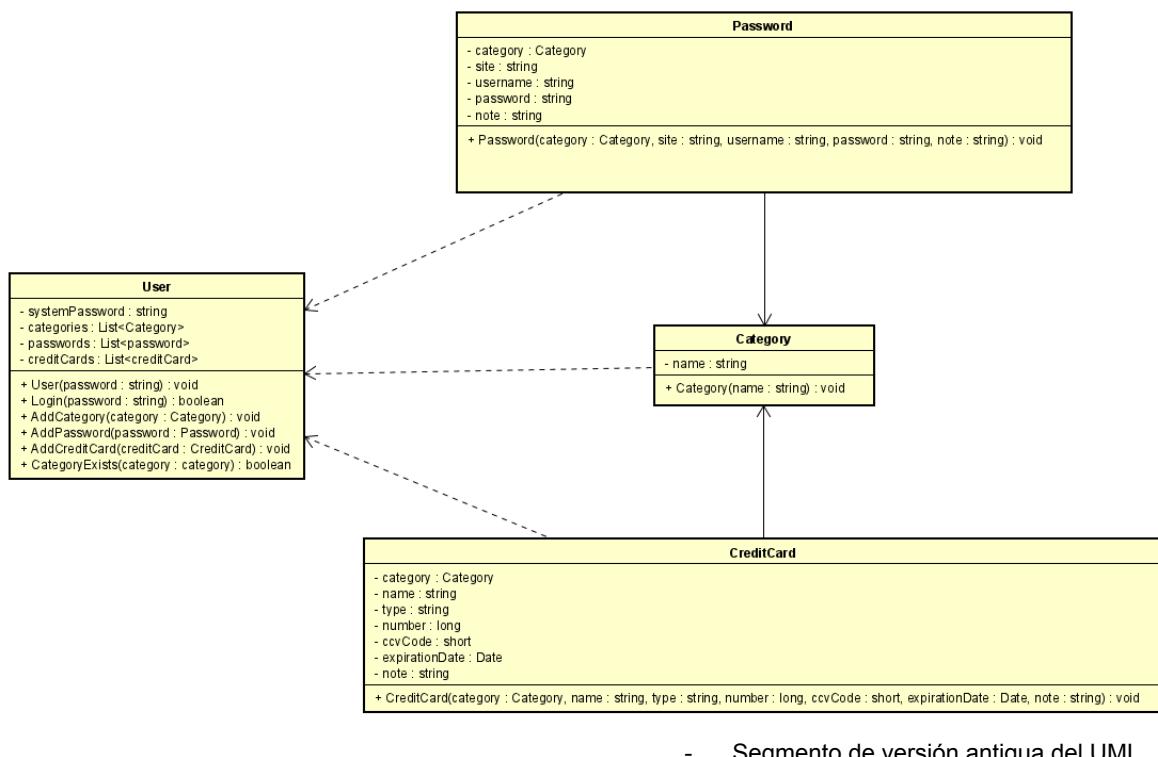
Diseño Definido para Perfil (Profile Class)

A continuación justificamos la implementación de esta clase y las ventajas de su diseño.

En una primera implementación básica de la clase profile se había definido en notación UML la siguiente estructura básica.

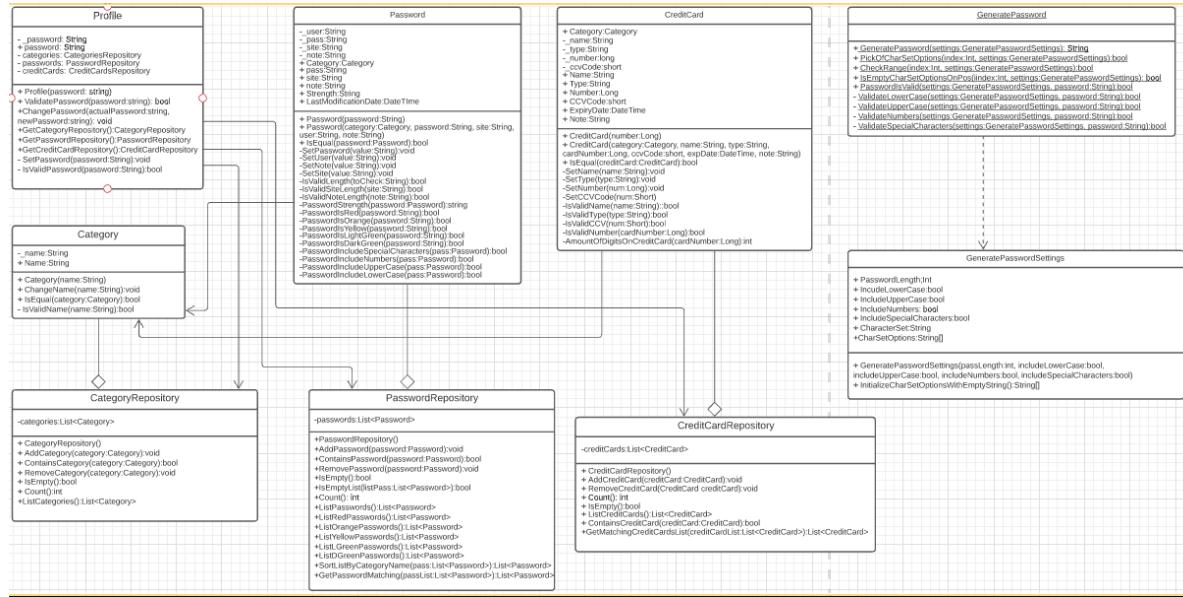
User actuando en esta versión antigua del UML como la clase profile, se puede observar como esta implementación hubiera causado una baja cohesión para muchos de los métodos definidos en la clase User, siendo que user funciona más como una clase sistema.

(La imagen a continuación no representa el diseño de la solución final)



Además de la baja cohesión de la clase, también se violaba SRP (Single Responsibility Principle)[\[1\]](#) debido a que esta clase user tenía más de una razón por la cual cambiar.

De esta forma decidimos implementar clases repositorios que se encargarán de implementar la forma en cómo se almacenarán los conjuntos de objetos y su interfaz pública, el resultado fue lo que se puede observar a continuación



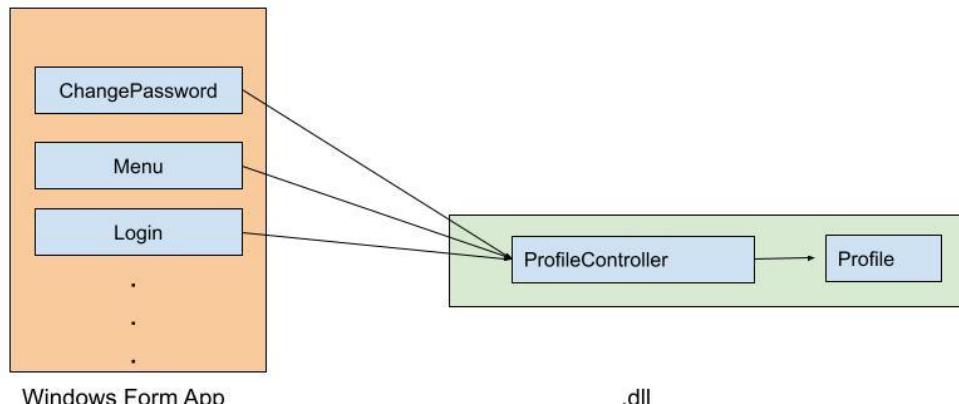
En el diagrama se puede observar como esta siguiente versión de la clase **Profile** presentaba una más alta cohesión que su anterior clase sin embargo a esta solución aún violaba el principio de SRP, ya que tenía mas de 1 razón para cambiar, como ejemplo el extender la aplicación a almacenar algún otro tipo de elemento que utilice un repositorio implica agregar otra propiedad en profile.

En continuas mejoras y rediseños de la clase **profile** llegamos a la versión final que se puede observar en el [diagrama uml del dominio](#). Esta versión tiene alta cohesión de sus funciones y es más fácil de entender y extender además de cumplir con SRP.

Conexión UserInterface y Lógica de Negocio

Explicado el uso de los repositorios a continuación se describe la conexión de la UI con la lógica de negocio a través del uso de controladores.

El siguiente diagrama representa a modo de ejemplo las dependencias de Paneles con el controlador de **Profile**.



Este Controlador es necesario ya que las conexiones a los repositorios se realizarán mediante el profile, el cual es accesible mediante su controlador.

Data breach

Implementar el nuevo formato no resultó algo complicado ya que para la primera entrega por error lo habíamos hecho y tuvimos que borrarlo. Pero si hicimos algunas modificaciones, creamos una clase en el dominio llamada DataBreach que guarda dos listas, una de contraseñas y otra de tarjetas de crédito, cuando se verifica un data breach se guarda un nuevo objeto DataBreach ya sea con las listas vacías o cargadas si existe una coincidencia. La lectura del archivo de texto se hace mediante un botón que implementamos, el mismo abre un explorador de archivos y al seleccionar lee el el texto, luego lo separa en caso de que exista algún "\t" como especifica en la letra.

Manejo de Excepciones

Como se mostró previamente se definieron un conjunto de excepciones con el fin de impedir violaciones a las reglas de negocios establecidos y con el fin de mantener el continuo funcionamiento de la aplicación brindando a esta la capacidad de responder a dichas eventualidades.

De esta forma se buscó evadir el uso masivo de segmentos try catch aislando los a eventos de modificación y creación de elementos (tales como: contraseñas, tarjetas de crédito, categorías).

Por otro lado, para evitar la ocurrencia de eventualidades no deseadas causado por posibles diferentes estados del sistema, se controla en todas las ventanas la disponibilidad de sus operaciones, habilitando las únicamente cuando se cuenta con el estado necesario para su ejecución.

Pruebas

Unit Test

Siguiendo la metodología de desarrollo TDD (Test Driven Development) se realizaron 165 pruebas unitarias para la verificación del correcto funcionamiento de las diferentes librerías de clases (dll). Esto incluye el package de PasswordManager y sus subcarpetas.

Prueba	Duración	Rasgos	Mensaje de error	Resumen del grupo
>PasswordManagerTest (189)	58 ms			Resumen del grupo
>PasswordManagerTest (189)	58 ms			PasswordManagerTest
CategoriesControllerTest (8)	16 ms			Pruebas en grupo: 189
CategoryRepositoryTest (10)	< 1 ms			Duración total: 58 ms
CategoryTest (8)	< 1 ms			
CreditCardControllerTest (10)	1 ms			
CreditCardRepositoryTest (10)	< 1 ms			
CreditCardTest (16)	< 1 ms			
DataBreachControllerTest (6)	39 ms			
DataBreachTest (4)	< 1 ms			
DataBreachesRepositoryTest (7)	< 1 ms			
PasswordGenerateTest (6)	1 ms			
PasswordRepositoryTest (18)	1 ms			
PasswordSettingsTest (17)	< 1 ms			
PasswordTest (32)	< 1 ms			
PasswordsControllerTest (17)	< 1 ms			
ProfileControllerTest (9)	< 1 ms			
ProfileTest (11)	< 1 ms			
				Salidas
				189 Correcta

Utilizando la extensión de Fine Code coverage logramos alcanzar mediante el uso de sus métricas una cobertura de código del 99,8%.

Fine Code Coverage					
	Coverage	Summary	Risk Hotspots	Rate & Review	Log Issue/Suggestion
Collapse all Expand all					
- Name					
- PasswordManager					
CategoriesController	761	1	762	1477	99.8%
Category	22	0	22	45	100%
CategoryAlreadyExistsException	24	0	24	53	100%
CategoryRepository	3	0	3	16	100%
CreditCard	33	0	33	61	100%
CreditCardAlreadyExistsException	86	0	86	145	100%
CreditCardRepository	3	0	3	16	100%
CreditCardsController	47	0	47	78	100%
DataBreach	26	0	26	55	100%
DataBreachesController	11	0	11	28	100%
DataBreachesRepository	16	0	16	40	100%
FailToValidatePasswordException	25	1	26	49	96.1%
InvalidCategoryNameException	3	0	3	12	100%
InvalidCreditCardCCVCodeException	3	0	3	12	100%
InvalidCreditCardNameException	3	0	3	16	100%
InvalidCreditCardNumberException	3	0	3	12	100%
InvalidCreditCardTypeException	3	0	3	16	100%
InvalidPasswordException	3	0	3	17	100%
InvalidPasswordNoteException	3	0	3	16	100%
InvalidPasswordSizeException	3	0	3	16	100%
InvalidPasswordUserException	3	0	3	16	100%
Password	163	0	163	268	100%
PasswordAlreadyExistsException	3	0	3	16	100%
PasswordGenerator	55	0	55	92	100%
PasswordRepository	75	0	75	113	100%
PasswordSettings	46	0	46	74	100%
PasswordsController	35	0	35	65	100%
Profile	39	0	39	75	100%
ProfileController	19	0	19	39	100%

Casos de prueba

Se realizaron casos de pruebas para el ingreso, modificación y listado de contraseñas, tarjetas de crédito, categorías, creación de un perfil, data breaches y modificación de contraseña maestra. Para esto se definió las entradas necesarias, clases validas e invalidas, luego documentamos el resultado esperado y lo probamos directamente en la interfaz gráfica.

Casos de prueba Perfil.

Definimos las entradas que son contraseña y volver a ingresar la misma contraseña, las clases válidas son que la contraseña tiene que tener entre 5 y 25 caracteres, y la contraseña tiene que coincidir en los dos campos. Las clases inválidas son cuando no cumple la longitud especificada anteriormente y cuando los campos no coinciden, teniendo en cuenta esto realizamos las pruebas.

Entrada/Variable	Clases válidas	Clases inválidas
Contraseña	Entre 5 y 25 caracteres (1)	<5 dígitos (3) >25 digitos(4)
Repetir contraseña	Igual a contraseña (2)	Contraseña diferente (5)

En la interfaz gráfica se ve de esta manera cuando ingresamos contraseñas de mayor o menor longitud de la que se acepta o no coinciden los campos.

Ingrese nueva contraseña

Ingrese nuevamente la contraseña

La contraseña debe tener entre 5 a 25 caracteres

Crear

Ingrese nueva contraseña

Ingrese nuevamente la contraseña

Las contraseñas no coinciden

Crear

Casos de prueba de categorías

Las clases de equivalencias definidas fueron:

Entrada/Variable	Clases válidas	Clases inválidas
Categoría	Entre 3 y 15 caracteres (1)	<3 dígitos (2) >15 dígitos(3) Categorías repetidas (4)

Para las cuales realizamos una tabla de pruebas y luego corrimos en la interfaz para chequear el resultado esperado, cuando creamos una categoría con un nombre que ya existe el mensaje es: “Ya existe una categoría con ese nombre”, y en caso de que se cree correctamente la categoría aparece “Categoría creada con éxito”. La ventana en la cual ingresamos el nombre de la categoría y tenemos el botón de crear no se cierra luego de crear porque muchas veces se quiere ingresar más de una categoría nueva.

Volver

Lista de categorias:

12345679012345
123456790123451
Personal
Trabajo
Trabajo.
Trabajo.@

Modificar Agregar

Categoría

Nombre: Personal

Categoría creada con éxito

Crear

Volver

Lista de categorias:

12345679012345
123456790123451
Personal
Trabajo
Trabajo.
Trabajo.@

Modificar Agregar

Categoría

Nombre: 12345678901234567

Largo del nombre de categoría incorrecto

Crear

Casos de prueba de Contraseña

Clases de equivalencia utilizadas:

Entrada/Variable	Clases válidas	Clases inválidas
Sitio	Entre 3 y 25 caracteres (1)	<3 dígitos (5) >25 digitos (6)
Usuario	Entre 5 y 25 caracteres (2)	<5 digitos (7) >25 digitos (8)
Contraseña	Entre 5 y 25 caracteres (3)	<5 digitos (9) >25 digitos (10) Espacios " " (11)
Notas	Entre 0 y 250 caracteres (4)	>250 caracteres (12)

Existe una clase inválida número (13) que es cuando el par usuario y sitio son iguales a una contraseña ya ingresada.

Cuando se crea y modifica correctamente una contraseña se cierra la ventana que nos permite hacerlo, y ya se puede ver en la lista la nueva contraseña. En caso de que ocurra un error tenemos mensajes para que el usuario sepa que está fallando en su nueva contraseña, por ejemplo “Largo de usuario incorrecto” este mensaje ocurre cuando ingresa un Usuario con longitud menor a 5 caracteres o mayor a 25 caracteres, otro mensaje es “Largo de contraseña incorrecto” que sale cuando se ingresa una contraseña con longitud incorrecta.

Contraseña

Categoría:	Personal
Sitio:	www.google.com
Usuario:	Ramon
Contraseña:	*****
Largo:	5
<input type="checkbox"/> Mayúsculas (A,B,C,...)	
<input type="checkbox"/> Minúsculas (a,b,c,...)	
<input type="checkbox"/> Dígitos (0,1,2,...)	
<input type="checkbox"/> Especiales (!,\$,{,<,...)	
Generar	
Note:	Nueva contraseña
Largo de usuario incorrecto	
Crear	

Contraseña

Categoría:	Personal
Sitio:	www.google.com
Usuario:	Javier
Contraseña:	Javier1231231231231231231
Largo:	5
<input type="checkbox"/> Mayúsculas (A,B,C,...)	
<input type="checkbox"/> Minúsculas (a,b,c,...)	
<input type="checkbox"/> Dígitos (0,1,2,...)	
<input type="checkbox"/> Especiales (!,\$,{,<,...)	
Generar	
Note:	Contraseña de gmail
Largo de contraseña incorrecto	
Crear	

Casos de prueba de Tarjetas de crédito

Entrada/Variable	Clases válidas	Clases inválidas
Nombre	Entre 3 y 25 caracteres (1)	<3 dígitos (7) >25 dígitos (8)
Tipo	Entre 3 y 25 caracteres (2)	<5 dígitos (9) >25 dígitos (10)
Numero	16 dígitos (3)	<16 dígitos (11) >16 dígitos (12) Caracteres no numéricos(13)
Código CCV	Entre 0 y 999 (4)	<0 (14) >0 (15)
Vencimiento	Fecha (5)	Formato fecha incorrecto (16)
Notas	Entre 0 y 250 caracteres (6)	>250 caracteres (17)
		Número ya existe en otra tarjeta agregada(18)

Para las tarjetas de crédito tenemos 18 clases de equivalencia y 6 entradas. Fueron realizadas pruebas para chequear que tenga una respuesta correcta la aplicación con varias combinaciones de estas entradas.

The figure consists of two side-by-side screenshots of a software window titled "CreditCard".

Left Screenshot (Successful Creation):

- Categoría:** Personal
- Nombre:** Matias
- Tipo:** Visa
- Número:** 1234111112341111
- Código CCV:** 134
- Vencimiento:** 05/2021
- Notas:** Tarjeta visa

Message at the bottom: **Tarjeta de credito creada correctamente**

Right Screenshot (Invalid CCV):

- Categoría:** Personal
- Nombre:** Matias
- Tipo:** MasterCard
- Número:** 1515252536364545
- Código CCV:** -1
- Vencimiento:** 05/2021
- Notas:** Tarjeta MasterCard de Matias

Message at the bottom: **Código CCV invalido**

Both screenshots feature a "Crear" button at the bottom center.

The image shows two side-by-side screenshots of a 'CreditCard' form interface. Both forms have identical fields: Categoría (Personal), Nombre (Renzo/Matías), Tipo (MasterCard), Número (1234/123412341234), Código CCV (134/1999), Vencimiento (05/2021), and Notas (Tarjeta MasterCard de renzo/Tarjeta MasterCard de Matías). The first form has an error message 'Largo del numero de tarjeta invalido' below the Número field. The second form has an error message 'La cadena de entrada no tiene el formato correcto.' below the Número field. Both forms have a 'Crear' button at the bottom.

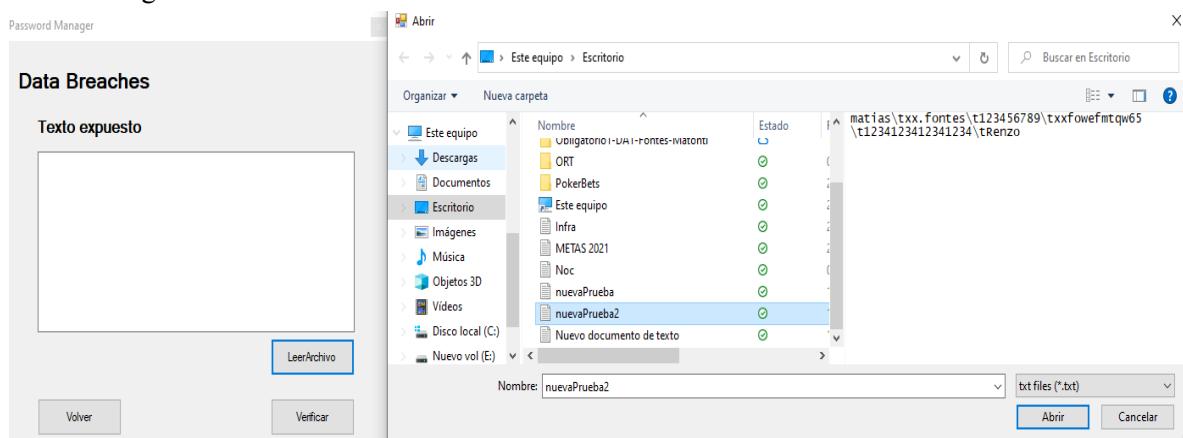
Casos de prueba de Data Breach

Las clases de equivalencia definidas fueron:

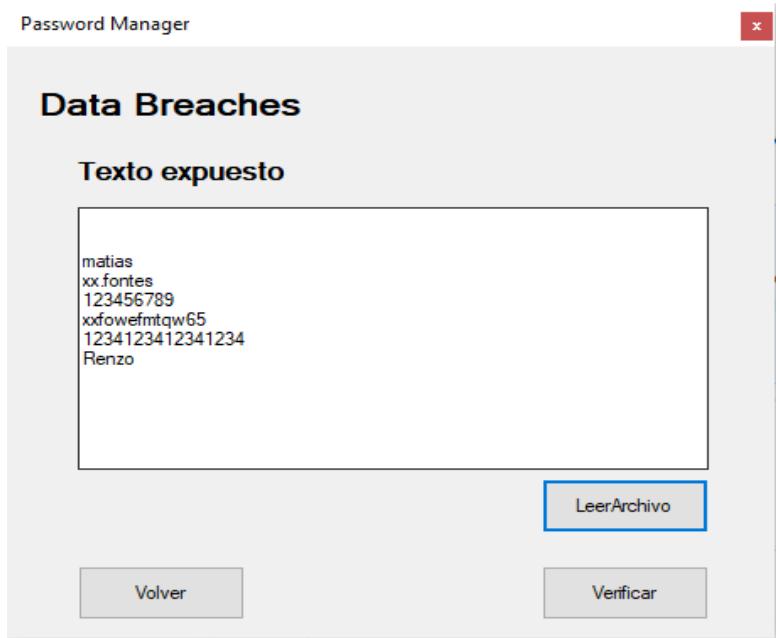
Entrada/Variable	Clases válidas	Clases inválidas
Texto	Contraseña (1) Tarjeta de crédito (2)	Texto vacío (3) Texto que no coincide con ninguna tarjeta y ninguna contraseña(4)

Con las pruebas probamos que funcione correctamente el lector del archivo de texto que ingresaba datos separados por \t, que la respuesta sea correcta en caso de textos vacíos y líneas vacías, además de cuando el texto no coincide con ninguna contraseña o tarjeta de crédito.

En esta imagen vemos como se abre el lector de archivos.



Al darle abrir nos aparece así el texto que estaba en el archivo .txt:



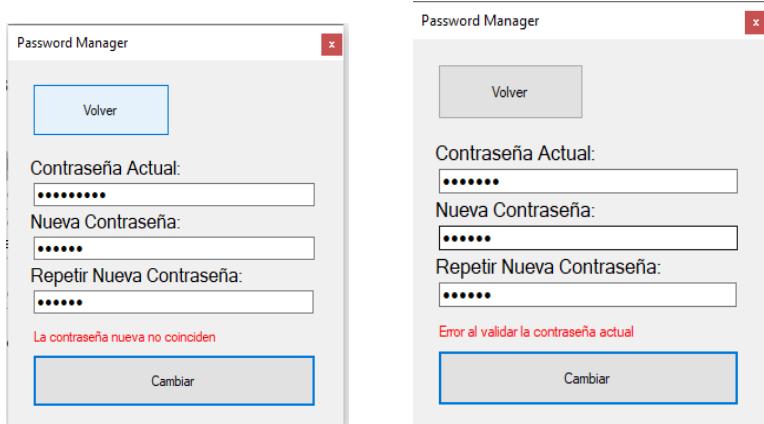
Casos de prueba para modificacion de contraseña maestra

Definimos las siguientes clases de equivalencia:

Entrada/Variable	Clases validas	Clases invalidadas
Contraseña actual	Contraseña correcta (1)	Contraseña incorrecta (4)
Nueva contraseña	Entre 5 y 25 caracteres (2)	Contraseña menos de 5 o más de 25 caracteres (5)
Repetir nueva contraseña	Contraseña coincide (3)	Contraseña no coincide(6)

Como resultado obtenemos que la contraseña se modifica o no, en la aplicación los mensajes que le da al usuario son los siguientes:

<input type="button" value="Volver"/> Contraseña Actual: <input type="text" value="*****"/> Nueva Contraseña: <input type="text" value="*****"/> Repetir Nueva Contraseña: <input type="text" value="*****"/> Contraseña Cambiada con éxito <input style="outline: none; border: 2px solid blue;" type="button" value="Cambiar"/>	<input type="button" value="Volver"/> Contraseña Actual: <input type="text" value="*****"/> Nueva Contraseña: <input type="text" value=".."/> Repetir Nueva Contraseña: <input type="text" value=".."/> Contraseña Invalida, debe tener entre 5 a 25 caracteres <input style="outline: none; border: 2px solid blue;" type="button" value="Cambiar"/>
---	---



Para ver los casos de prueba realizados: [\[Casos de prueba\]](#)

Instalación

Se requiere tener las aplicaciones para ejecutar aplicaciones de escritorio de windows, junto a una distribución de SQL Express.

El ejecutable se encuentra en la carpeta Release/UserInterface.exe.

En este caso tenemos dos archivos .bak, uno se llama EmptyPasswordManagerDB.bak que es la base de datos vacía y tenemos otro con el nombre PasswordManagerDB.bak que tiene los datos de prueba cargados. Estos archivos deben estar en la carpeta Backup que se encuentra donde fue instalada la instancia de SQL EXPRESS. Los backups los pueden encontrar en la carpeta llamada “Base de datos y script” en el repositorio de github.

Las librerías requeridas son: Entity framework 6.

Para la migración de Base de datos se debe primero modificar el siguiente archivo:

De estos archivo se debe modificar el nombre de la instancia de SQL Express en la línea de

connectionString *?"SQLExpress"

App config de PasswordManagerDataLayer.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->
    <section name="entityFramework"
      type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
      requirePermission="false"/>
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2"/>
  </startup>
  <entityFramework>
    <providers>
      <provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer"/>
    </providers>
  </entityFramework>
  <connectionStrings>
    <add name="PasswordManagerDB" connectionString="Data Source=.\\SQLExpress;Database=PasswordManagerDB;Trusted_Connection=True;" providerName="System.Data.SqlClient" />
  </connectionStrings>
</configuration>
```

[\[Abrir imagen\]](#)

App config de UserInterface

```

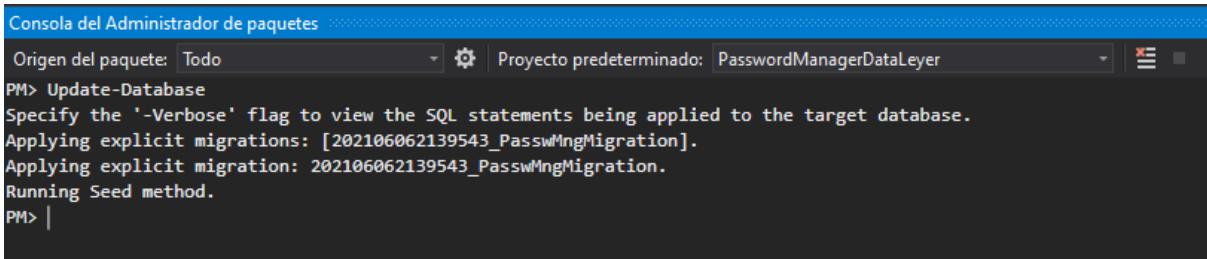
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkId=237468 -->
    <section name="entityFramework"
      type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
      requirePermission="false"/>
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2"/>
  </startup>
  <entityFramework>
    <providers>
      <provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer"/>
    </providers>
  </entityFramework>
  <connectionStrings>
    <add name="PasswordManagerDB" connectionString="Data Source=.\\SQLEXPRESS;Database=PasswordManagerDB;Trusted_Connection=True;" providerName="System.Data.SqlClient"/>
  </connectionStrings>
</configuration>

```

[\[Abrir imagen\]](#)

A continuación seleccionar como proyecto de inicio PasswordManagerDataLayer, y abrir la consola de administrador de paquetes, donde también debe estar como por predeterminado el proyecto PasswordManagerDataLayer.

Por último correr el comando de “Update-Database” para migrar la ya existente configuración a SQL Express



The screenshot shows the 'Consola del Administrador de paquetes' (Package Manager Console) window. The console output is as follows:

```

Consola del Administrador de paquetes
Origen del paquete: Todo          |   Proyecto predeterminado: PasswordManagerDataLayer
PM> Update-Database
Specify the '-Verbose' flag to view the SQL statements being applied to the target database.
Applying explicit migrations: [202106062139543_PasswMngMigration].
Applying explicit migration: 202106062139543_PasswMngMigration.
Running Seed method.
PM> |

```

Contraseña maestra para ingresar a la aplicación con datos cargados: **prueba1234**

Bibliografía

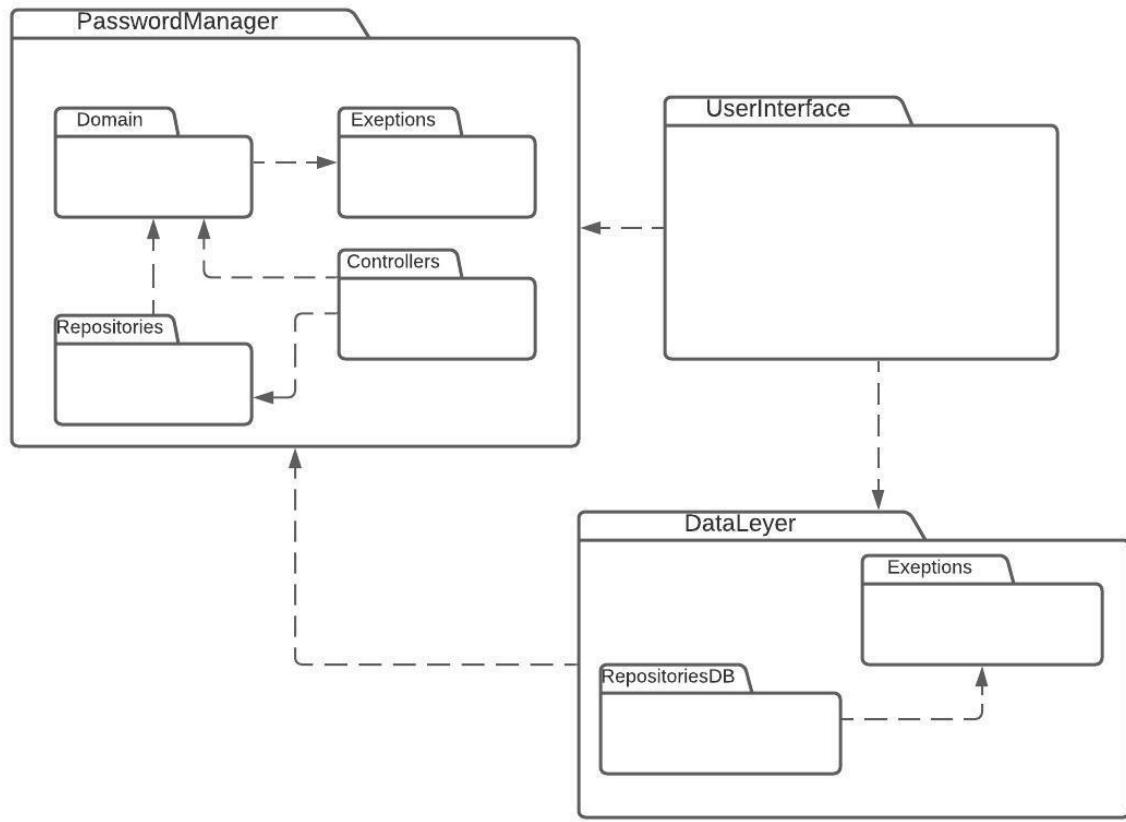
Regex, representa expresiones regulares inmutables
fuente: [Regex Microsoft Docs](#)

SRP, Single Responsibility Principle
fuente: [Wikipedia](#)

TDD, Test Driven Development
fuente: [Wikipedia](#)

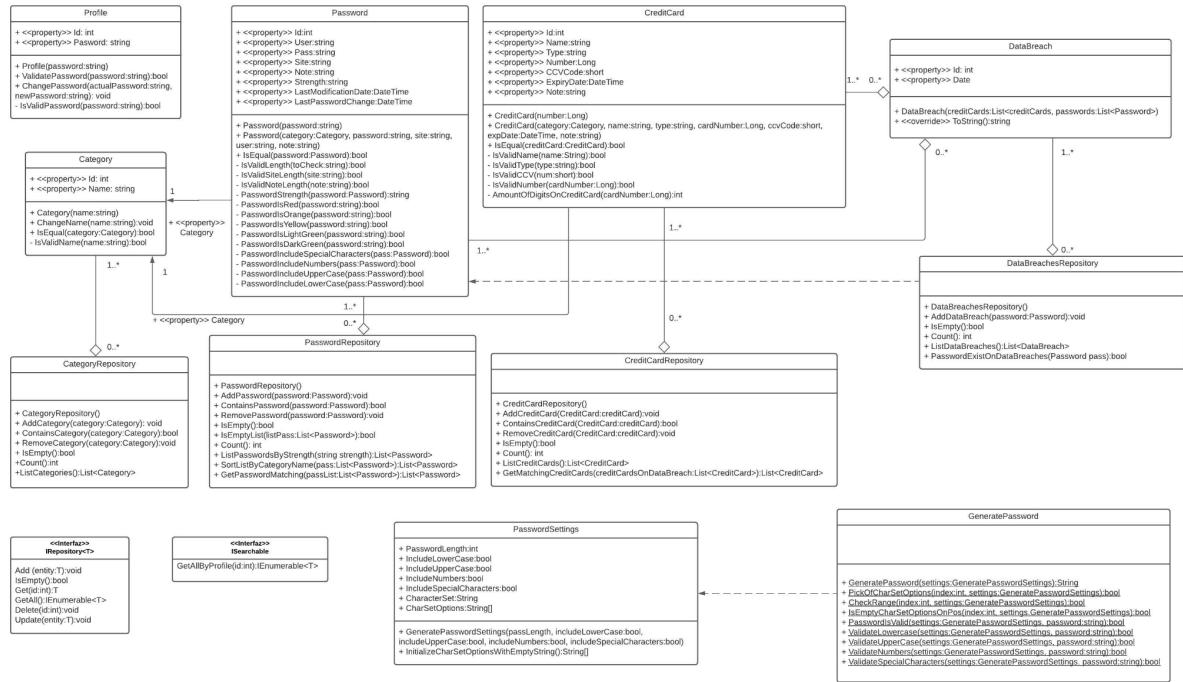
Anexo

Diagrama de paquetes



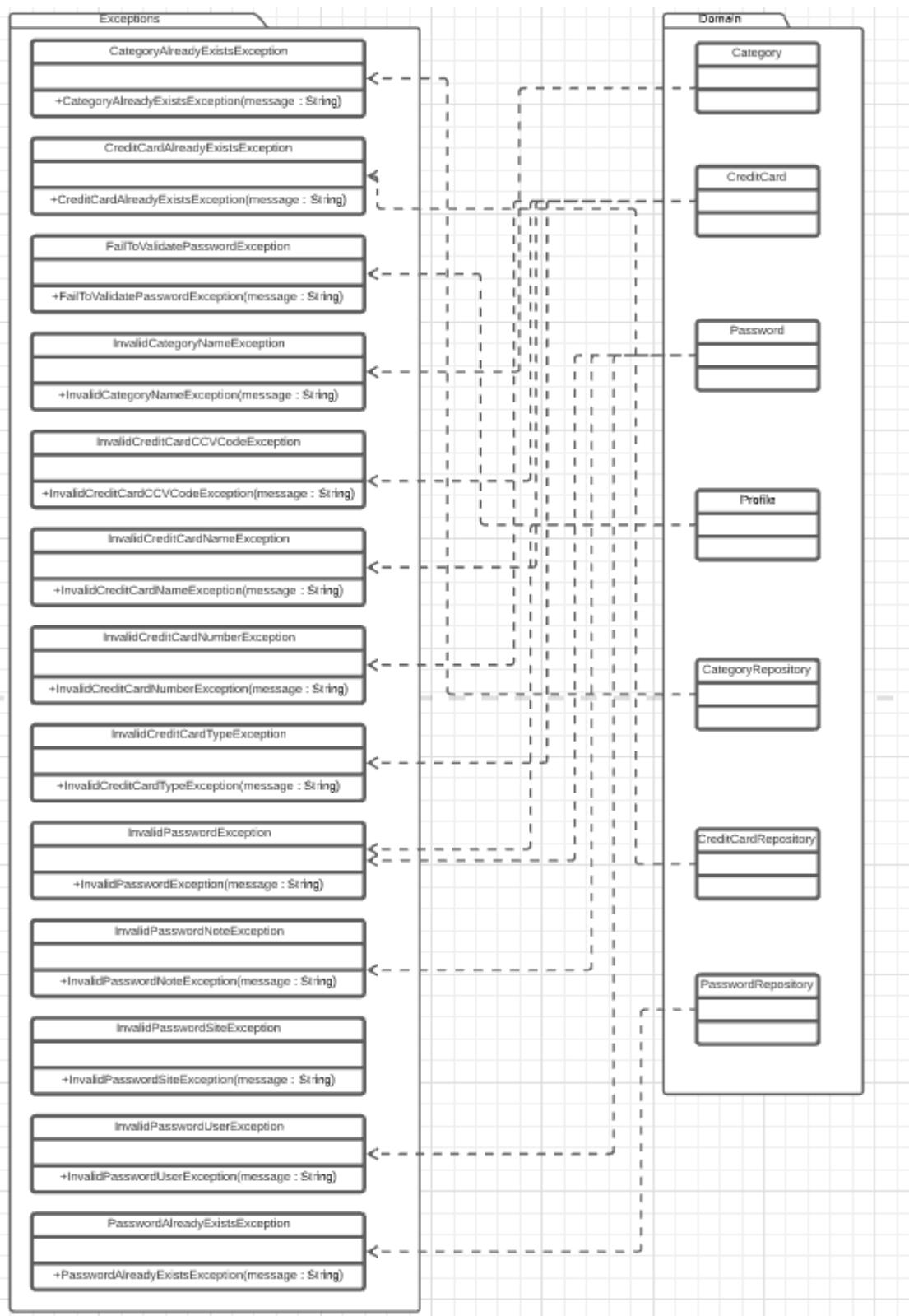
[\[Abrir archivo\]](#)

Diagrama de clases [PasswordManager]Domain



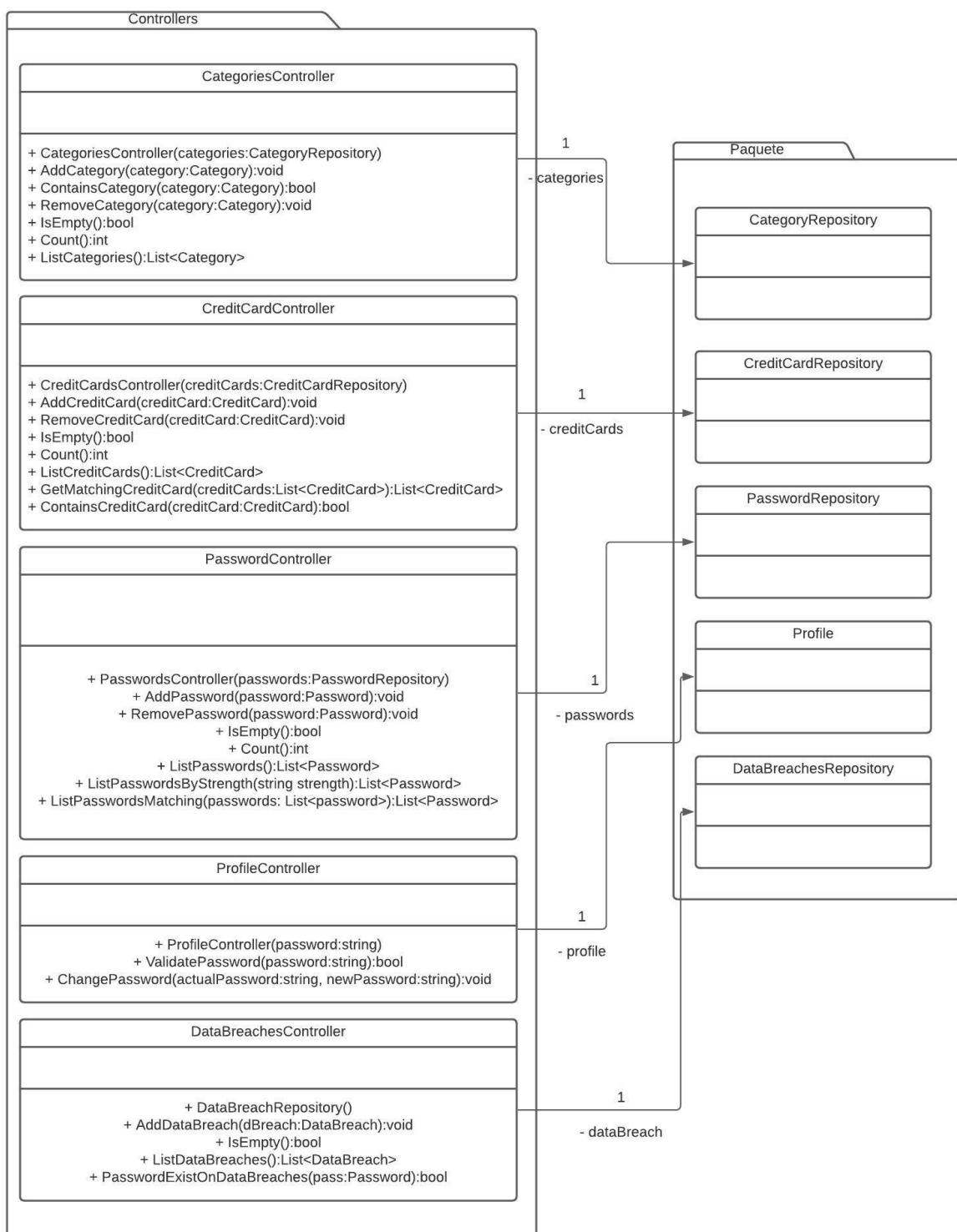
[\[Abrir archivo\]](#)

Diagrama de clases [PasswordManager]Exceptions



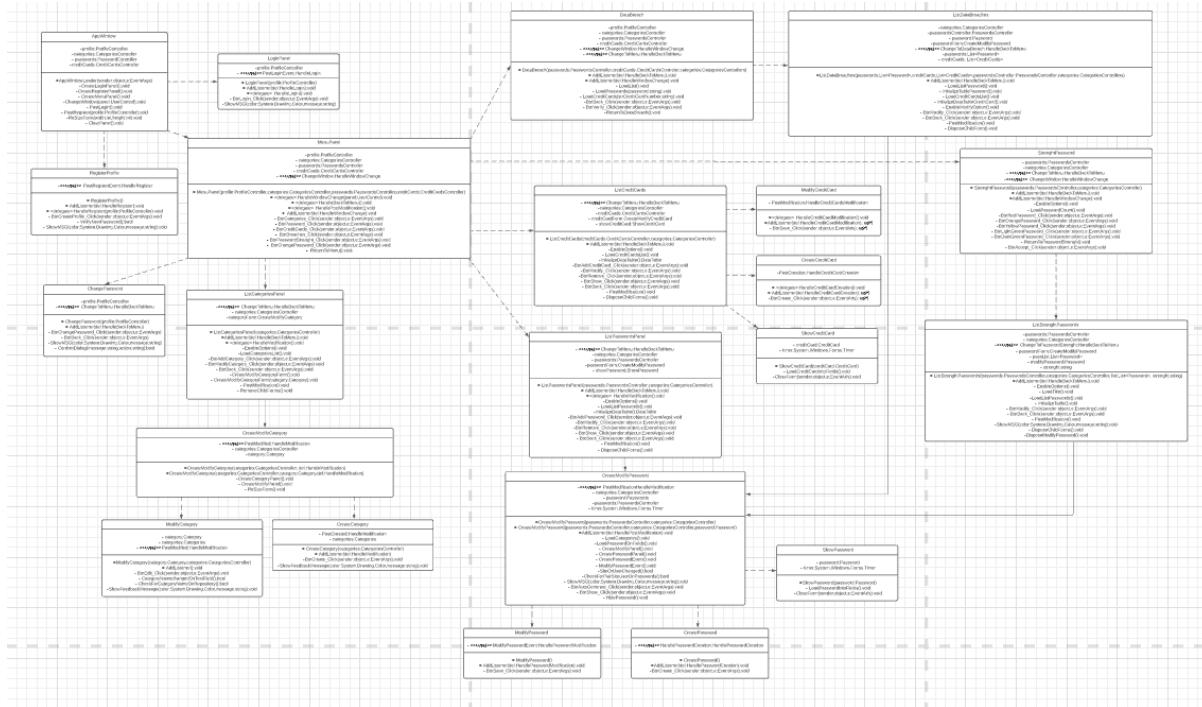
[\[Abrir archivo\]](#)

Diagrama de clases - [PasswordManager]Controllers



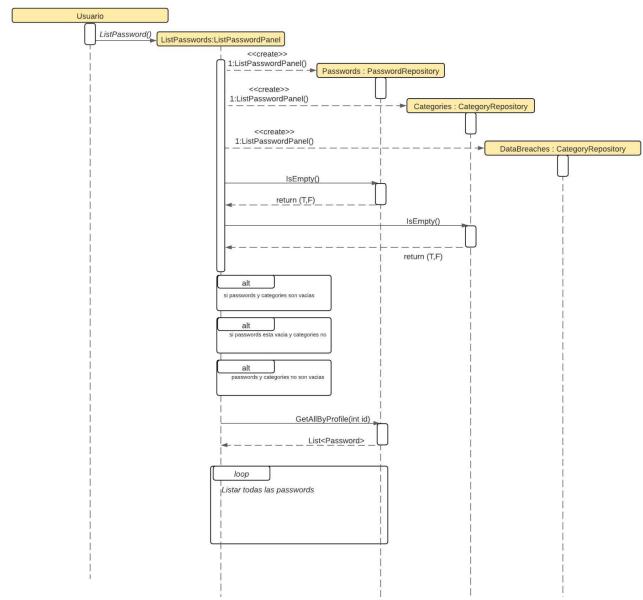
[Abrir archivo]

Diagrama de clases - [UserInterface]

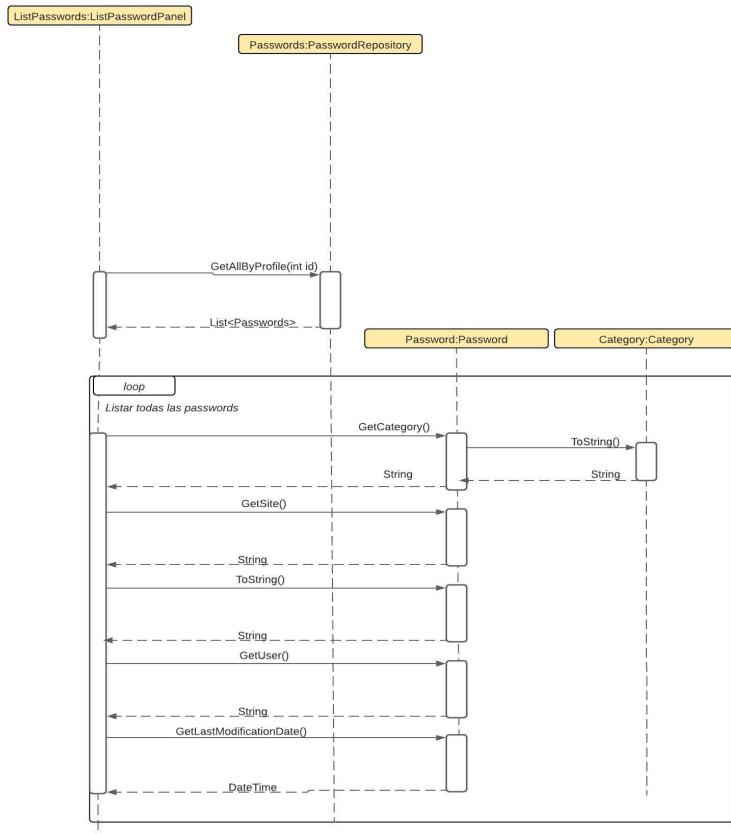


[Abrir archivo]

Diagrama de secuencia- [List password]

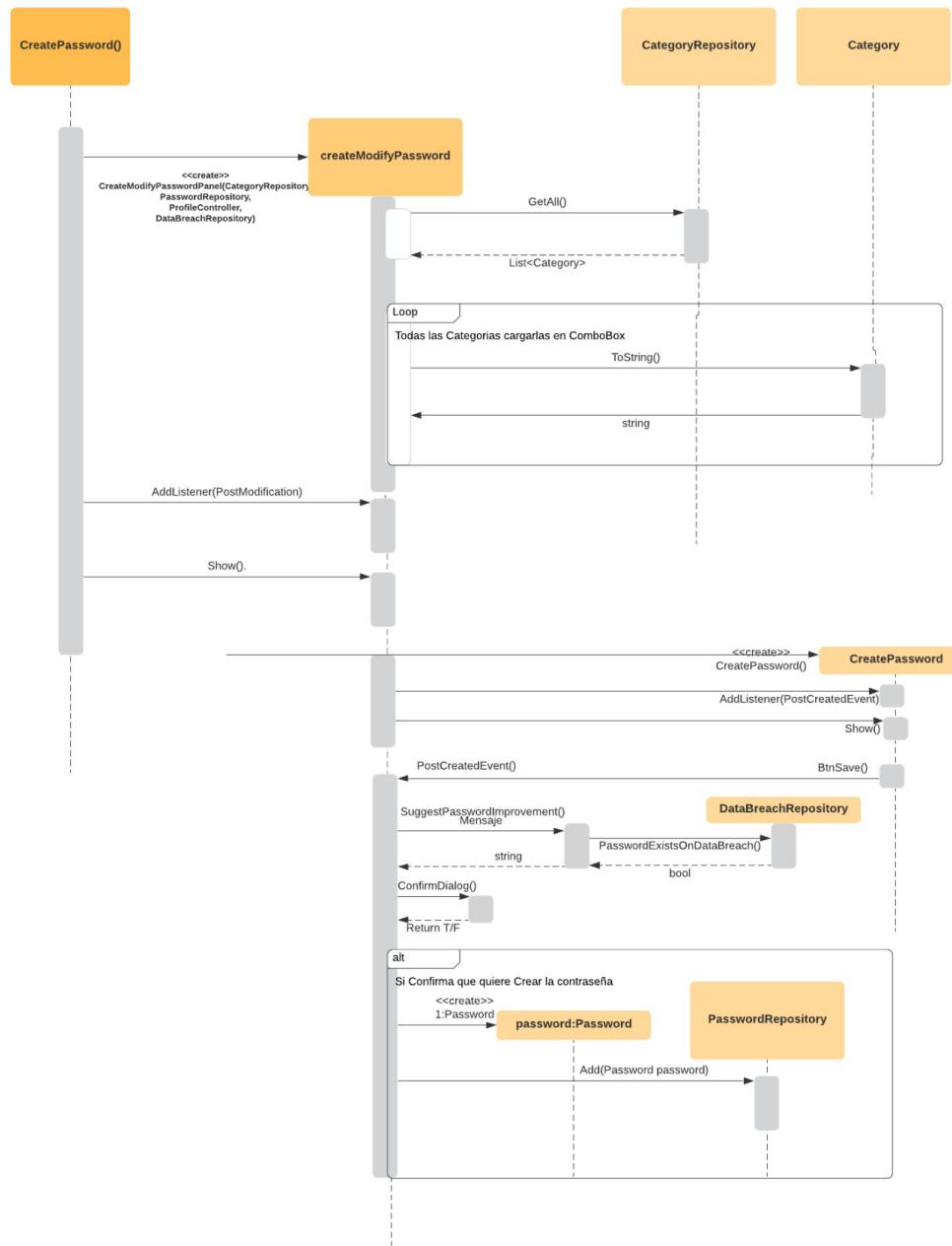


[\[Abrir archivo\]](#)



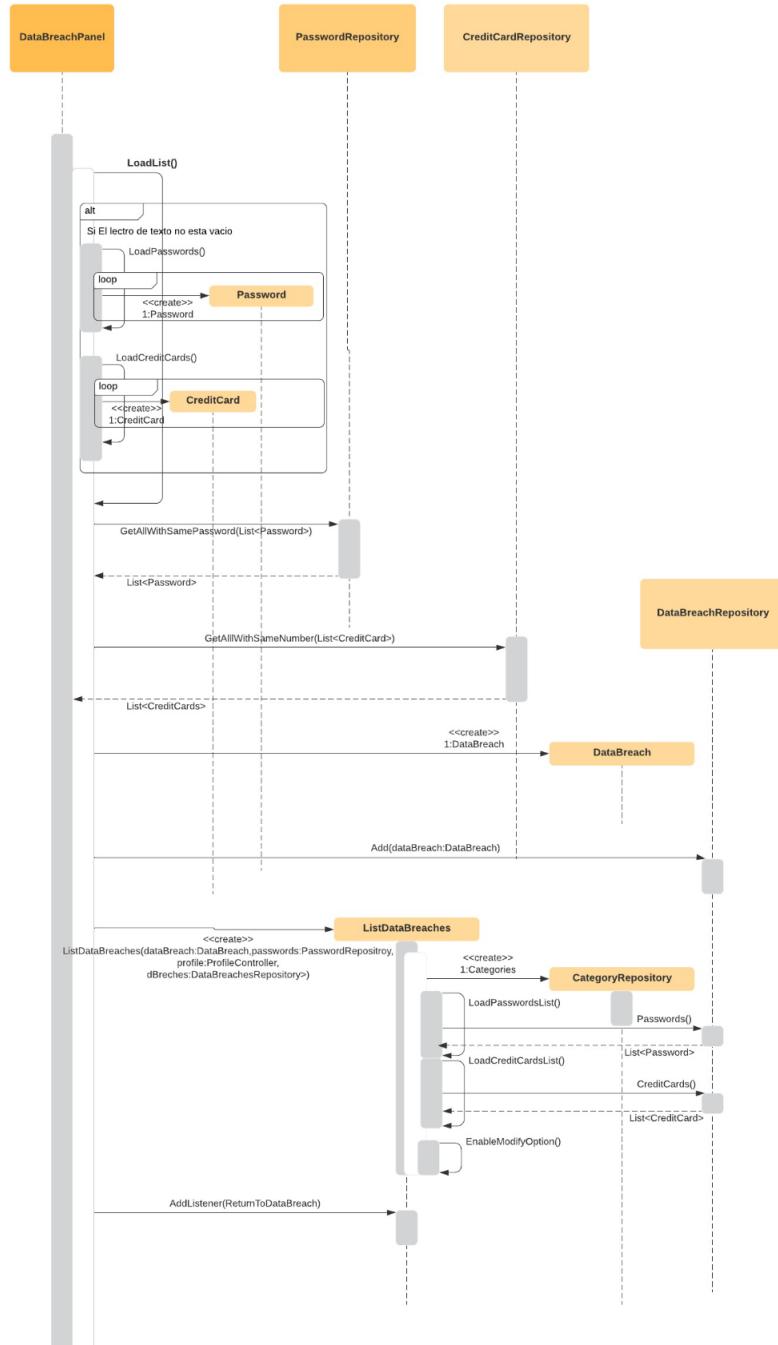
[\[Abrir archivo\]](#)

Diagrama de secuencia- [Create password]



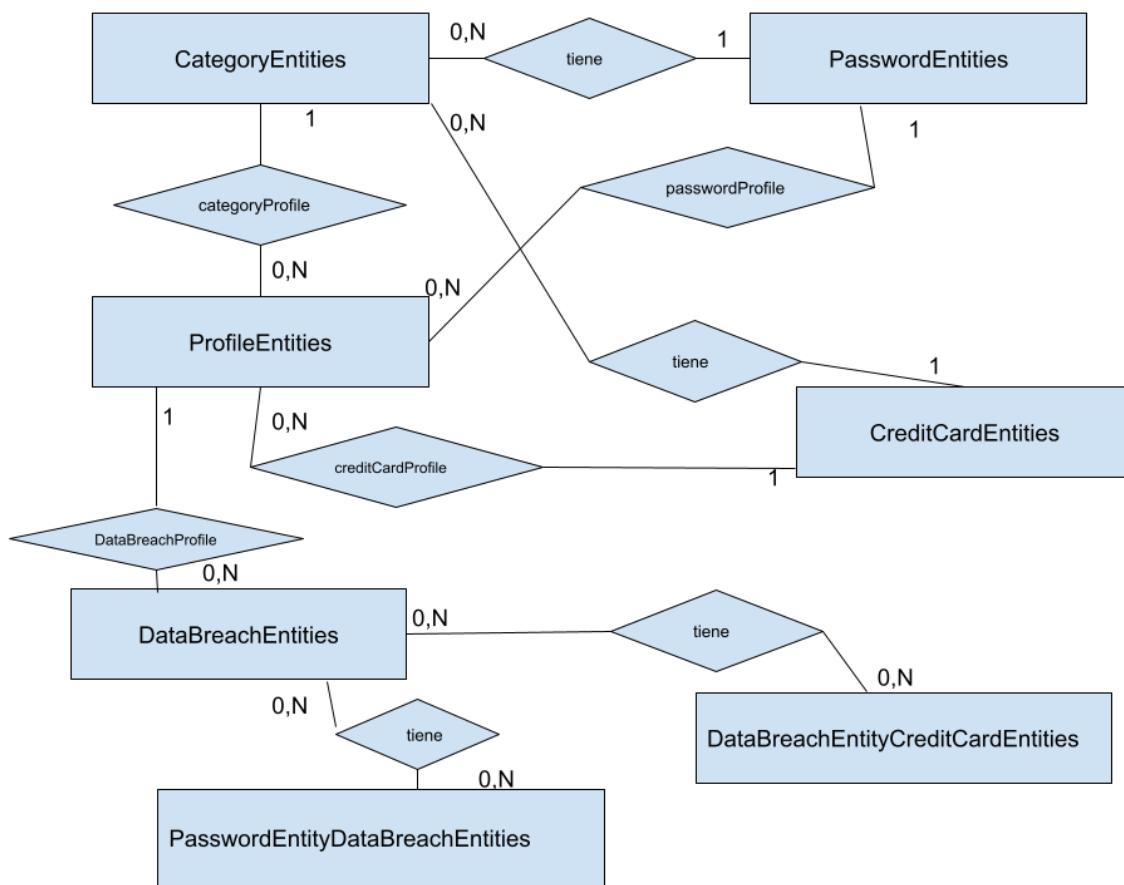
[\[Abrir archivo\]](#)

Diagrama de secuencia- [Verify data breach]



[\[Abrir archivo\]](#)

Modelo entidad-relación



Casos de prueba

Caso de prueba	Nombre	Tipo	Numero	Codigo CCV	Vencimiento	Notas	Resultado esperado	Clases de equivalencia cubiertas
CPCC01	Matias	Visa	1234111112341111		134	05/2021 Tarjeta visa de Matias	Creada correctamente	1, 2, 3, 4, 5, 6
CPCC02	Ra	Visa	1234111112341112		134	05/2020 Tarjeta visa de Ra	Largo de nombre invalido	2,3,4,5,6,7
CPCC03	RamonRamonRamonRaonRamonRam	Visa	1234111112341113		134	05/2020 Tarjeta visa de RamonNombreLargo	Largo de nombre invalido	2,3,4,5,6,8
CPCC04	Luis	Vi	1234111112341114		134	05/2021 Tarjeta Vi	Largo de tipo incorrecto	1,3,4,5,6,9
CPCC05	Luis	MasterCardMasterCardMasterCardMaster	1234111112341115		134	05/2020 Tarjeta MasterCardMasterCard	Largo de tipo incorrecto	1,3,4,5,6,10
CPCC06	Renzo	MasterCard	1234123411111234535656	1234	134	05/2021 Tarjeta de Renzo masterCard	Largo de numero incorrecto	1,2,4,5,6,11
CPCC07	Renzo	MasterCard	1234123411111234535656	134	05/2021 Tarjeta de Renzo masterCard	Largo de numero incorrecto	1,2,4,5,6,12	
CPCC08	Renzo	Visa	1234FFFF12344566		189	11/2023 Tarjeta Visa	Formato incorrecto	1,2,4,5,6,13
CPCC09	Matias	MasterCard	1515252536364545	-1		05/2021 Tarjeta de Matias MasterCard	Codigo CCV invalido	1,2,3,5,6,14
CPCC10	Matias	MasterCard	1515252536364546	1999		05/2023 Tarjeta de Matias MasterCard	Codigo CCV invalido	1, 2, 3, 5, 6, 15
CPCC11	Manuela	Visa	1515252536364546	989	wf/2021	Tarjeta de Manuela	No permite escribir caracteres no numericos	1, 2, 3, 4, 6, 16
CPCC13	Luis	MasterCard	1515252536364546	561	09/2022	Tarjeta ... (258 caracteres)	No permite escribir mas de 250 caracteres	1,2,3,4,5,17
CPCC13	Matias	MasterCard	1234111112341111		189	11/2023 Tarjeta de Matias MasterCard	Numero de tarjeta ya existe	1,2,4,5,6,18

En el enlace se puede encontrar los casos de prueba realizados.

[\[Abrir archivo de casos de prueba\]](#)