

Universidad ORT Uruguay

Facultad de Ingeniería

Diseño de aplicaciones 1

Obligatorio 1

Juan Pablo Poittevin(169766)
Joselen Cecilia(233552)

Entregado como requisito de la materia Diseño de
aplicaciones 1

29 de octubre de 2020

Declaraciones de autoría

Nosotros, Juan Pablo Poittein y Josele Cecilia, declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos Diseño de Aplicaciones 1 ;
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hemos acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Resúmen

Para la materia Diseño de Aplicaciones 1, en el primer obligatorio se plantea la elaboración de un sistema que permita a los usuarios registrar sus transacciones y obtener reportes. El objetivo de este proyecto es aplicar los conocimientos adquiridos en clase, Clean code y TDD.

Índice general

1. Descripción General	4
2. Descripción y justificación de diseño	6
2.1. Diagrama de paquetes	6
2.2. Diagrama de clases	6
2.3. Suposiciones de diseño	8
2.4. Descripción de diseño	9
2.4.1. Diseño actual	9
2.4.2. Mejoras de diseño	10
2.4.3. Análisis de responsabilidades.	10
3. Pruebas	12
3.1. Cobertura de pruebas unitarias	12
3.2. Casos de prueba	13
4. Anexo	16

1. Descripción General

Los expertos en finanzas personales recomiendan el registro de gastos, y aseguran que hacerlo ayuda a identificar patrones de consumo, reconocer categorías en las que se gasta más, establecer objetivos de ahorro; y en general, lograr una mejor planificación financiera.

Este trabajo se enmarca en un proyecto para implementar una aplicación que permita a los usuarios registrar sus transacciones y obtener reportes. En esta primera etapa, el objetivo es la construcción de una prueba de concepto que modele las principales funcionalidades en forma sencilla, y con una interfaz de usuario básica que permita simular el uso de la aplicación.

Las características funcionales que interesan para esta primera versión son:

- 1 - La aplicación debe mantener un registro de categorías de gastos.
- 2 - Se debe permitir agregar transacciones individuales con una categoría y monto.
- 3 - Se debe poder establecer un presupuesto para cada mes, en el que cada categoría tiene un monto estimado de gasto.
- 4 - Se debe mostrar un reporte por mes de todos los gastos realizados.
- 5 - Se debe ver un reporte por mes, en el que se muestra para cada categoría el monto estimado y el real.

Al ingresar a la aplicación el usuario se encuentra con una lista de botones cada uno referido a las distintas funciones del programa, registrar categoría, registrar gasto, registrar o editar presupuesto, reporte de gastos, reporte de presupuestos, editar categoría y editar gastos.

La lógica de negocio del programa fue desarrollada utilizando TDD (desarrollo guiado por pruebas) y siguiendo las buenas prácticas propuestas por Clean Code, el cual habla de como escribir código limpio.

Para el versionado trabajamos con Git, que es una herramienta utilizada para el control de versiones del software. Esto nos permite tener versiones antiguas estables en caso de necesitarlas.

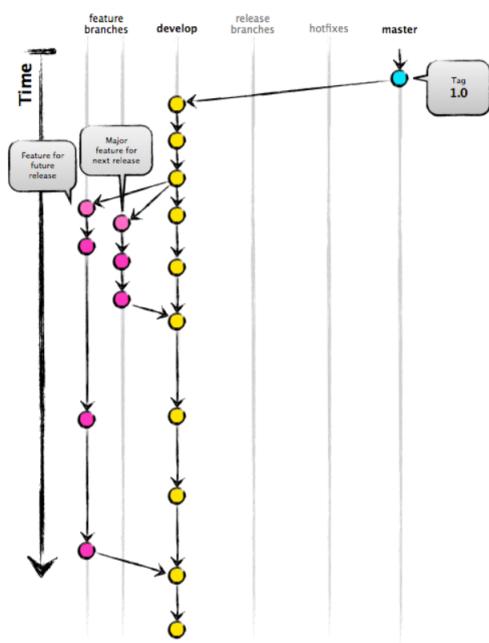
Nos brinda la posibilidad de tener múltiples respaldos del proyecto, ya que cada persona que trabaje en el proyecto tendrá una copia del repositorio de forma local, así como también existirá un repositorio central en la nube.

Para especificar la estructura interna del repositorio nos basamos en la arquitectura GitFlow, donde existe una rama master, de la cual luego se desprende una rama develop.

De la rama develop se crean las ramas "FEATURE BRANCHES", cada una de estas se corresponderá con una clase o una característica del proyecto.

Una vez teniendo la nueva característica o clase estable, tenemos que hacer merge con la rama develop. Además creamos features branches para el refactoring donde hicimos todos los cambios necesarios para mejorar la calidad del proyecto. Al final cuando se termine el proyecto se hará una release branch desde la última versión de develop hacia la rama master, haciendo merge con esta última.

En la siguiente imagen se detalla la arquitectura especificada anteriormente



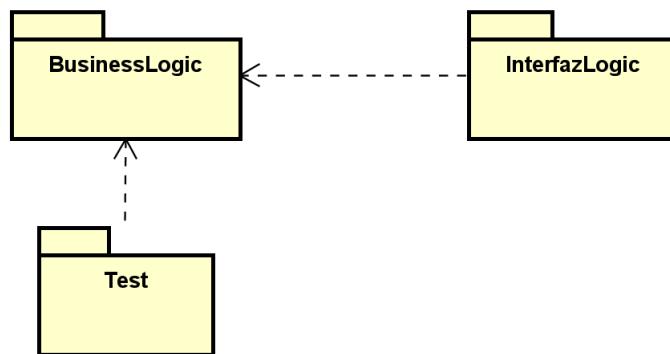
2. Descripción y justificación de diseño

2.1. Diagrama de paquetes

El diagrama de paquetes representa las dependencias entre los paquetes que componen el modelo. Es decir, muestra cómo un sistema está dividido en agrupaciones lógicas y las dependencias entre esas agrupaciones.

Dado que normalmente un paquete está pensado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema. El diagrama de paquetes tiene como objetivo dar una visión más clara del sistema, permiten dividir un modelo para agrupar y encapsular sus elementos de lógica, interfaz y test, detallando las relaciones entre ellos.

Los paquetes se utilizan para mostrar la arquitectura del sistema a nivel macro, estos se representan en el diagrama como carpetas.



2.2. Diagrama de clases

Las relaciones existentes entre las distintas clases son dependencia, asociación, agregación y composición.

Dependencia: Hay dependencia entre dos clases cuando trabajan juntas por un periodo de tiempo.

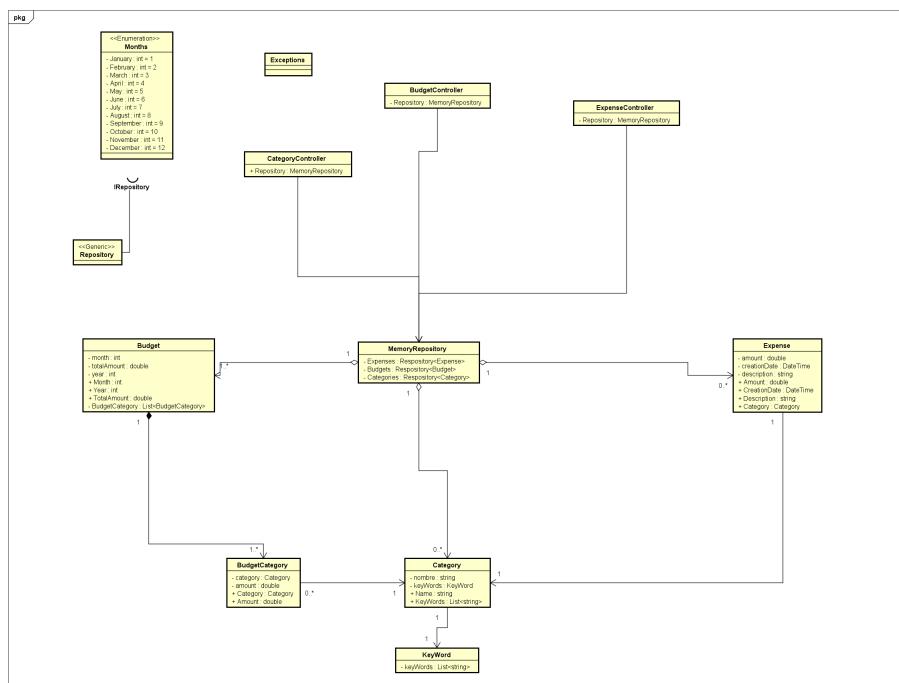
do corto de tiempo. Cuando una clase “A” crea una instancia, recibe una instancia o devuelve una instancia de una clase “B”, esa clase “A” tiene una dependencia con la clase B y se representa con una flecha de triangulo punteada.

Asociación: Cuando los objetos de una clase trabajan con objetos de otra por un periodo prolongado de tiempo. Cuando una clase “A” se construye a partir de una clase “B”, entonces la clase “A” tiene una asociación con la clase “B”. Se representa con una flecha simple.

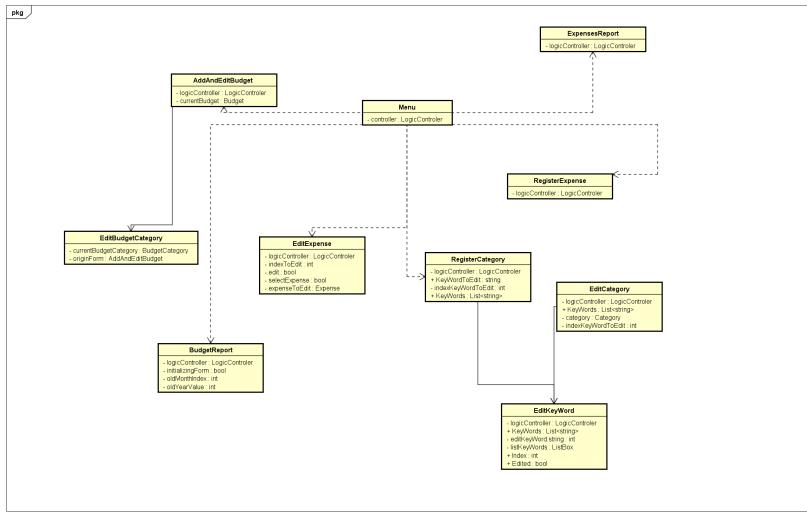
Agregación: Hay una relación compuesto-componente, en la cual el compuesto es dueño del componente, pero comparte la referencia con otra clase. Se representa con una flecha simple con un rombo blanco en la otra punta.

Composición: Hay una relación compuesto-componente, pero el compuesto no comparte la referencia a sus componentes. Se representa con una flecha simple con un rombo negro en la otra punta.

-Diagrama del paquete BusinessLogic



-Diagrama del paquete InterfazLogic



2.3. Suposiciones de diseño

Para realizar el diseño del obligatorio nos basamos en la letra del mismo y en las preguntas que se fueron realizando en el foro, además hicimos algunos supuestos:

- Se puede crear un expense con los mismos description, amount, creationDate y category ya existentes en otro expense. Esto surgió de la necesidad de que un cliente pude ir dos veces en el mismo día al mismo lugar y querer registrar el mismo gasto.
- Al ingresar una nueva category se verifica que no se ingrese dos veces la misma "key word", en dicha category, ya que no tendría sentido tener dos veces la misma palabra clave en la misma category.
- Se crea una clase budgetCategory, la misma nos facilita el control del presupuesto para cada una de las categorías.
- No se permite que el nombre de una categoría sea solo números, ya que a nuestro parecer no tendría mucho sentido.
- Decidimos utilizar en la lógica un enum month para representar la variable month de las distintas clases ya que es en estas donde se trabaja con dicha variable, y en las ventanas un string ya que se obtiene así de los campos de estas.
- En las ventanas decidimos separar el registrar gasto del eliminar y editar gastos ya que creemos que facilita el uso y entendimiento de la funcionalidad al usuario.
- También decidimos separar el registrar categoría del editar categoría ya que creemos que facilita el uso y entendimiento de la funcionalidad al usuario.
- En cambio registrar presupuesto y editar presupuesto los dejamos en la misma ventana ya que el editar, simplemente agrega un botón de editar presupuesto por

categoría, donde el usuario puede cambiar el monto de este, por lo que creímos innecesario tener que separarlas.

2.4. Descripción de diseño

2.4.1. Diseño actual

Para la implementación de este sistema creamos tres proyectos; un Class Library .NET Framework al cual llamamos BusinessLogic, un MSTest .NET Framework al cual llamamos Test y por ultimo un WindowsForms .NET Framework al cual llamamos InterfazLogic

BusinessLogic:

Dentro de BusinessLogic creamos cuatro clases correspondientes a los principales elementos del sistema Expense, Budget, Category, BudgetCategory.

Creamos también una Interfaz IRepository la cual es implementada por la clase Repository de tipo generic esta se encarga de las principales funciones de manejo de los elementos guardados en el sistema.

Además tenemos una clase MemoryRepository que usa Repository para almacenar las categorías, gastos y presupuesto.

Para manejar los errores del sistema creamos una clase Exceptions que contiene las excepciones creadas por nosotros, y una clase EnumMonths que contiene el enum Month el cual lista los meses del año.

Para comunicar la interfaz de usuario con la lógica de negocio creamos además distintos handlers para las distintas funcionalidades, uno que se encarga del registrar y editar categoría (CategoryController), otro que se encarga de registrar, dar reporte y editar gasto(ExpenseController) y por ultimo uno que se encarga de registrar, dar reporte y editar presupuesto(BudgetController).

La ventaja de mantener separada la lógica de la interfaz es que si en un futuro se decide usar esta lógica para otro tipo de aplicación se puede adaptar fácilmente. Además el hecho de tener el repository facilita el poder adaptar el obligatorio para implementar el uso de una base de datos.

Test:

Dentro de Test tenemos las clase Tests correspondientes a cada clase del paquete BusinessLogic que tiene logica (enumMonth, Exceptions e IRepository no tienen clase test), en las cuales fuimos creando los test para generar la logica del sistema.

InterfazLogic:

Dentro de InterfazLogic tenemos una clase UserControl por cada funcionalidad del sistema y una clase Formulario que es el Menú principal, el cual contiene los botones a todas las funcionalidades, además tenemos dos formularios mas uno para editar

las key words y otro para editara el budget category

2.4.2. Mejoras de diseño

2.4.3. Análisis de responsabilidades.

-Category, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Crear los objetos de tipo Category.

Manejar los datos de la categoria, nombre y palabras claves.

Pasar a string la categoria.

Ver si dos categorias son iguales.

-Expense, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Crear los objetos de tipo Expense.

Manejar los datos de los gastos, monto del gasto, fecha en la que se creo el gasto, descripcion del gasto y categoria de este.

Ver si dos gastos son iguales.

-Budget, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Crear los objetos de tipo Budget.

Manejar los datos del presupuesto, monto total, mes y año de cuando se crea, y una lista de BudgetCategories.

Pasar a string el presupuesto.

Ver si dos presupuestos son iguales.

-BudgetCategory, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Crear los objetos de tipo BudgetCategory.

Manejar los datos del presupuesto, monto y categoria.

Pasar a string el budgetCategory.

Ver si dos budgetCategory son iguales.

-KeyWords, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Manejar la lista de string donde se almacenan las palabras claves.

-BudgetController, esta clase se encuentra dentro del paquete BusinessLogic y se

encarga de:

Es la fachada, el handler, entre Budget y la interfaz.

-CategoryController, esta clase se encuentra dentro del paquete BusinessLogic

Es la fachada, el handler, entre category y la interfaz. -ExpenseController, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Es la fachada, el handler, entre Expense y la interfaz. - IRepository, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Mostrar la firma de los metodos basico para manejar un Repository, esta clase es una interfaz -Repository, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Implementar los metodos de IRepository.

-MemoryRepository, esta clase se encuentra dentro del paquete BusinessLogic y se encarga de:

Crear los objetos donde se almacenan los datos de category, expense y budget.
Manejar los datos de category, Expense y budget.

3. Pruebas

3.1. Cobertura de pruebas unitarias

Durante las pruebas unitarias, como fue mencionado previamente, se utilizo la metodología de TDD.

Lo que nos llevo a poder alcanzar un porcentaje de cobertura del 100 por ciento de la lógica de negocio.

Un porcentaje de prueba que creemos fue superior al que hubiéramos tenido de no utilizar la metodología de TDD.

Desarrollar el código de esta forma nos permitió ver claramente los casos de cada función, donde fallaba, que casos limites validar, y que variantes tener en cuenta.

Hierarchy	Not Covered (Blocks)	Not Covered (% Blocks)	Covered (Blocks)	Covered (% Block)
businesslogic.dll	0	0,00 %	857	100,00 %
BusinessLogic	0	0,00 %	857	100,00 %
Budget	0	0,00 %	147	100,00 %
Budget.<>c	0	0,00 %	6	100,00 %
Budget.<>c_Displa...	0	0,00 %	2	100,00 %
BudgetCategory	0	0,00 %	24	100,00 %
BudgetController	0	0,00 %	141	100,00 %
Category	0	0,00 %	58	100,00 %
CategoryController	0	0,00 %	78	100,00 %
Expense	0	0,00 %	49	100,00 %
ExpenseController	0	0,00 %	106	100,00 %
KeyWord	0	0,00 %	77	100,00 %
KeyWord.<>c	0	0,00 %	2	100,00 %
MemoryRepository	0	0,00 %	142	100,00 %
MemoryRepository....	0	0,00 %	2	100,00 %
MemoryRepository....	0	0,00 %	2	100,00 %
MemoryRepository....	0	0,00 %	2	100,00 %
Repository<T>	0	0,00 %	19	100,00 %

3.2. Casos de prueba

En el menú presiono el botón Register Expense

Están registradas en el sistema las categorias

Category 1

Name: Entretenimiento

Key words: cine, casino

Category 2

Name: Comida

Key words: cena

Anexo 1

[Anexo 2]

Se ingresa expense con todos los campos completos

Anexo 3

Se selecciona el boton Register

Anexo 4

Se ingresa una descripción de mas de 20 caracteres

Anexo 5

Se ingresa una descripción de menos de 3 caracteres

Anexo 6

Se ingresa un amount de mas de dos decimales, el programa informa de error y vuelve el valor al minimo amount que es 1,00

Anexo 7

Se ingresa creation date con año fuera del rango.

Anexo 8

Anexo 9

Se ingresa descripción de gasto la cual contiene una key word de la categoría

entretenimiento, se presiona el botón SEARCH.

Anexo 10

Se ingresa descripción de gasto la cual contiene no contiene ninguna key word de las categorías registradas.

Anexo 11 No se selecciona ninguna categoría de la lista de categorías.

Anexo 12

Si no hay categorías registradas en el sistema al monto de crear el gasto.

Anexo 13

En el menú presiono el botón EDIT EXPENSE

Si no hay gastos registrados en el sistema

Anexo 15

Se asume registrado el expense:

Description: salida

Amount:1,10

Date:21/10/2020

Category:Entretenimiento

Anexo 16

Se modifican todos los datos correctamente

Anexo 17

Se selecciona el botón EDIT sin tener seleccionado ningún expense

Anexo 18

Se selecciona el botón DELETE sin tener seleccionado ningún expense

Anexo 19

Se ingresa una nueva description con menos de 3 caracteres

[Anexo 20]

Se ingresa una nueva description con más de 20 caracteres

Anexo 21

Se ingresa un amount con más de dos decimales se le informa al usuario del error y

se vuelve el valor al minimo 1

Anexo 22

Se ingresa un date con año fuera de rango, menor a 2018.

Anexo 23

Se ingresa un date con año fuera de rango, mayor a 2030

Anexo 24

Se presiona el boton EDIT CATEGORY y no se selecciona ninguna categoria

Anexo 25

Se presiona el boton EXPENSE REPORT

Anexo 26

Anexo 27

No se seleccion mes que reportar

Anexo 28

Se selecciona mes para hacer el reporte

Anexo 29

No hay gastos registrados

4. Anexo

The screenshot shows a Windows application window titled "Register expense". The window has a menu bar with "Menu" and standard window controls (minimize, maximize, close). On the left is a vertical menu bar with several options: "Register Category", "Register Expense", "Register edit Budget", "Expense Report", "Budget Report", "Edit expenses", and "Edit Category". The "Register Expense" option is highlighted with a blue border. The main area contains input fields for "Description" (empty), "Amount" (set to "1,00" with a spin button), "Date" (set to "jueves , 29 de octubre de 2020" with a calendar icon), and "Category" (a large text input field with a "Search" button to its right). At the bottom are "Accept" and "Cancel" buttons.

Anexo 1

Menu

- Register Category
- Register Expense
- Register Budget
- Expense Report
- Budget Report
- Edit expenses
- Edit Category

Register category

Name	Entretenimiento
Key word	<input type="text"/> <input type="button" value="Add key word"/>
List of key word	cine casino
	<input type="button" value="Edit key word"/> <input type="button" value="Delete key"/>
	<input type="button" value="Register"/>

Anexo 2

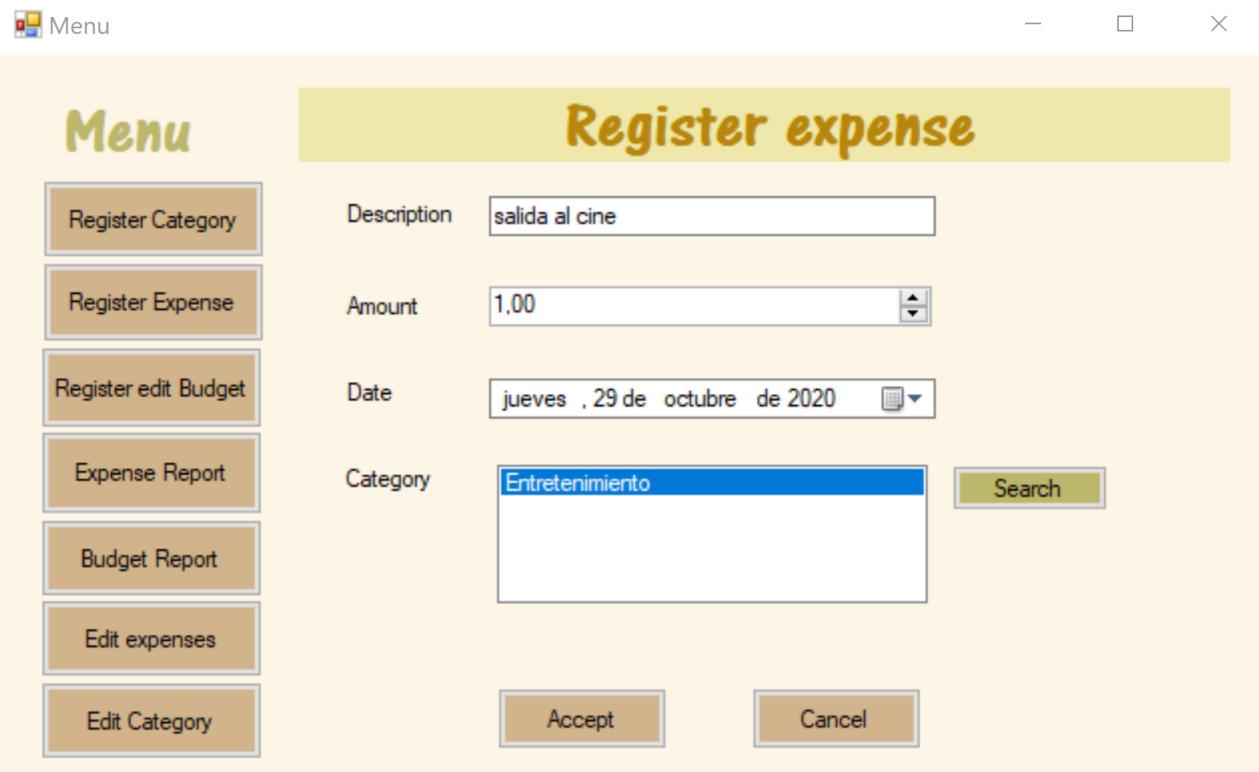
Menu

- Register Category
- Register Expense
- Register Budget
- Expense Report
- Budget Report
- Edit expenses
- Edit Category

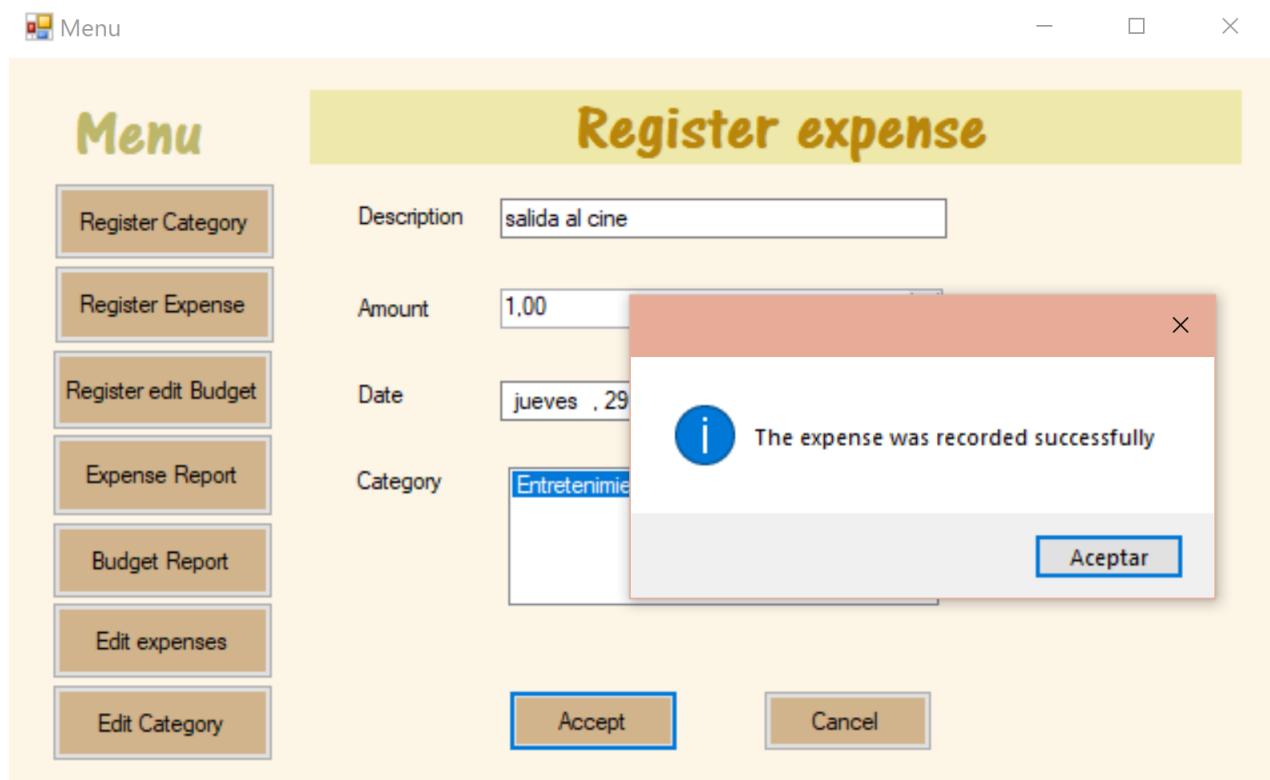
Register category

Name	Comida
Key word	<input type="text"/> <input type="button" value="Add key word"/>
List of key word	cena
	<input type="button" value="Edit key word"/> <input type="button" value="Delete key"/>
	<input type="button" value="Register"/>

Anexo 3



Anexo 4



Anexo 5

Menu

Register expense

Register Category	Description	cuando fuimos al cine
	The description must be between 3 and 20 characters long.	
Register Expense	Amount	1,00
Register edit Budget	Date	jueves , 29 de octubre de 2020
Expense Report	Category	Entretenimiento
Budget Report		<input type="button" value="Search"/>
Edit expenses		
Edit Category		

Anexo 6

Menu

Register expense

Register Category	Description	
	The description must be between 3 and 20 characters long.	
Register Expense	Amount	1,00
Register edit Budget	Date	jueves , 29 de octubre de 2020
Expense Report	Category	Entretenimiento
Budget Report		<input type="button" value="Search"/>
Edit expenses		
Edit Category		

Anexo 7

Menu

Register expense

Register Category	Description	Entretenimineto
Register Expense	Amount	1,00 The amount cannot have more than two decimal places
Register edit Budget	Date	jueves , 29 de octubre de 2020
Expense Report	Category	Entretenimiento
Budget Report		<input type="button" value="Search"/>
Edit expenses		
Edit Category		<input type="button" value="Accept"/> <input type="button" value="Cancel"/>

Anexo 8

Menu

Register expense

Register Category	Description	Entretenimineto
Register Expense	Amount	1,00
Register edit Budget	Date	domingo , 29 de octubre de 2017
Expense Report	Category	Entretenimiento The year must be between 2018 and 2030.
Budget Report		<input type="button" value="Search"/>
Edit expenses		
Edit Category		<input type="button" value="Accept"/> <input type="button" value="Cancel"/>

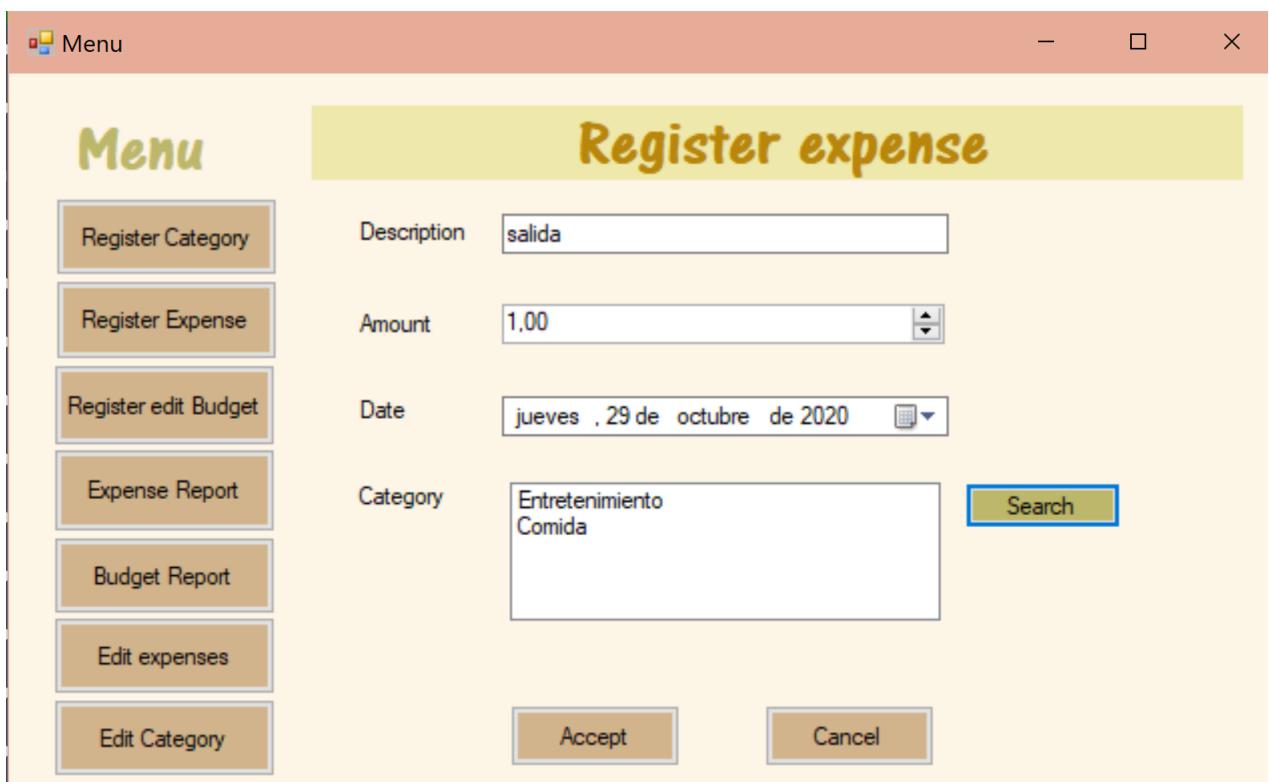
Anexo 9

The screenshot shows a Windows application window titled "Register expense". On the left, there is a vertical menu bar with the following options: Register Category, Register Expense, Register edit Budget, Expense Report, Budget Report, Edit expenses, and Edit Category. The "Register Expense" option is highlighted. The main area contains fields for Description, Amount, Date, and Category. The "Description" field has the value "Entretenimineto". The "Amount" field has the value "1,00". The "Date" field shows "miércoles, 29 de octubre de 2031" with a calendar icon. The "Category" field has the value "Entretenimiento" and includes a note: "The year must be between 2018 and 2030." A "Search" button is located next to the category dropdown. At the bottom right are "Accept" and "Cancel" buttons. A status bar at the bottom of the window displays the message "No se han ingresado expensas con todos los campos completados".

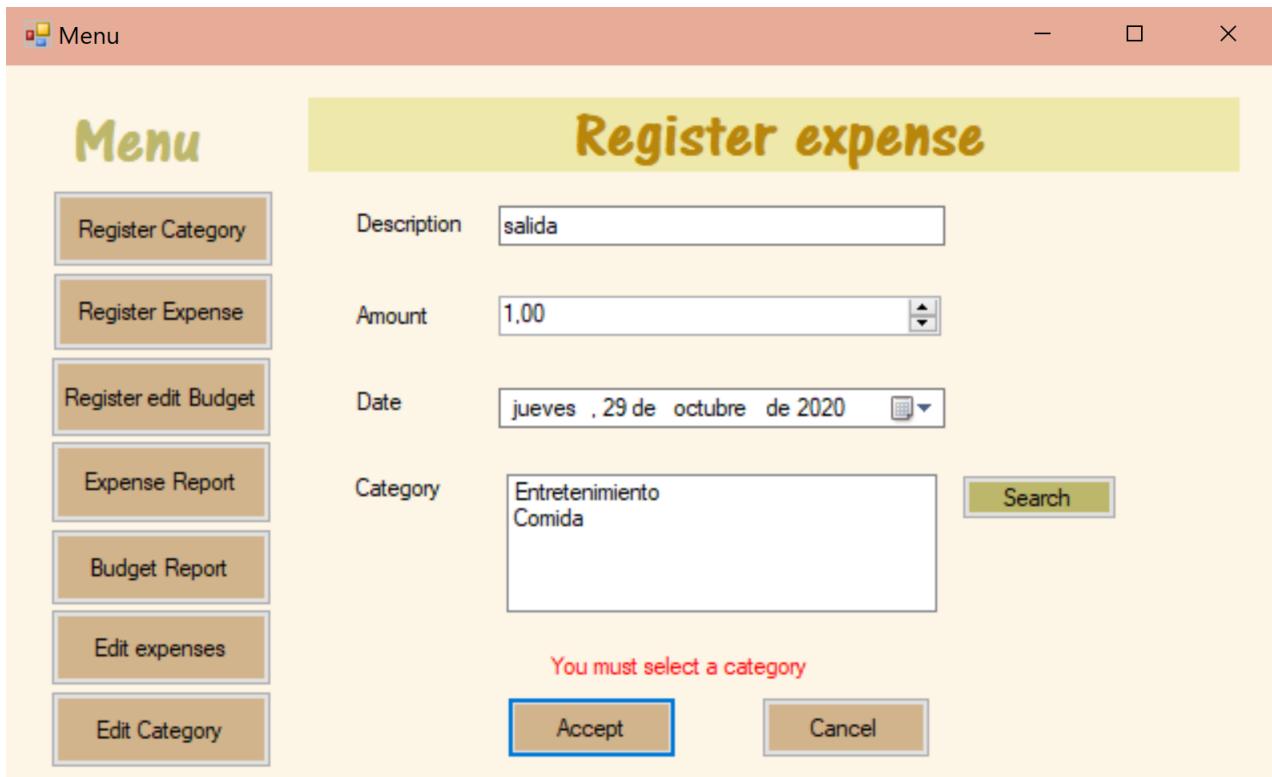
Anexo 10

The screenshot shows a Windows application window titled "Register expense". The layout is identical to Anexo 9, with a vertical menu on the left and a main form on the right. The "Register Expense" option is highlighted in the menu. The main form fields are: Description (value: "salida al cine"), Amount (value: "1,00"), Date (value: "jueves , 29 de octubre de 2020"), and Category (value: "Entretenimiento"). The "Search" button is highlighted with a blue border. At the bottom right are "Accept" and "Cancel" buttons. The status bar at the bottom of the window is empty.

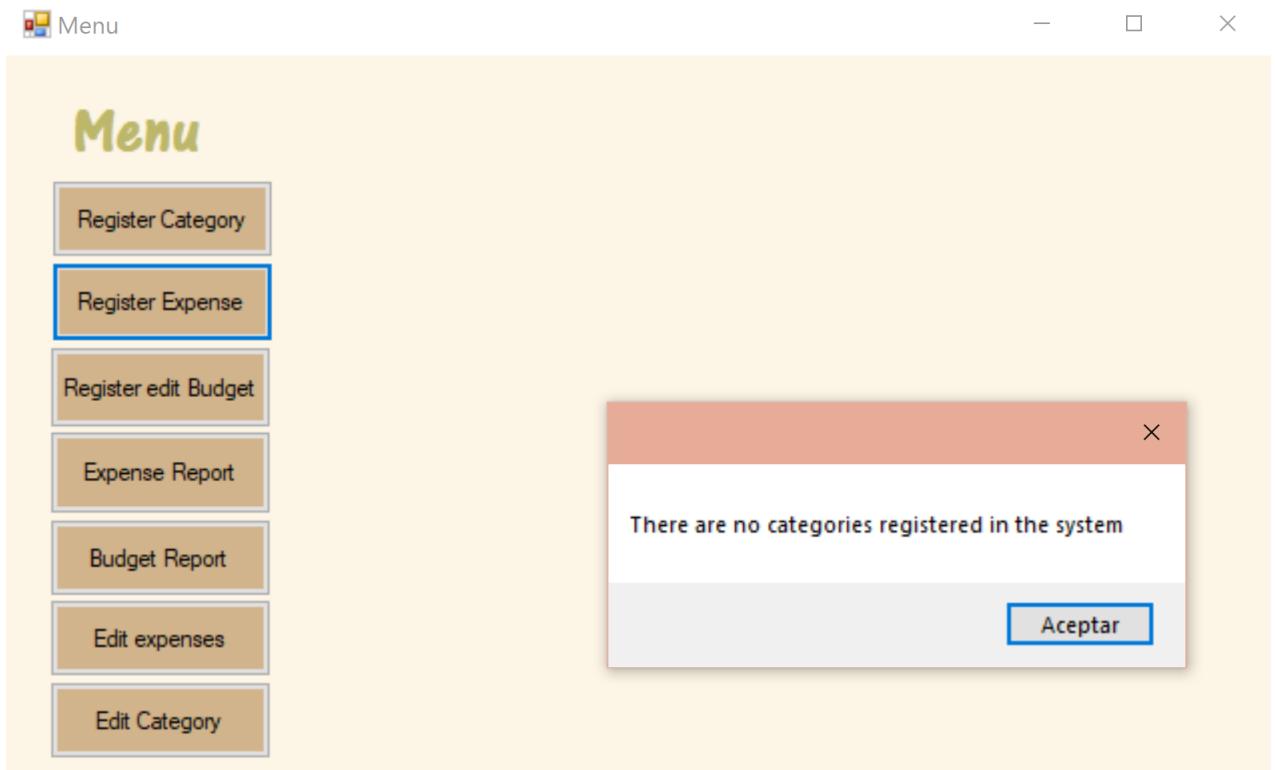
Anexo 11



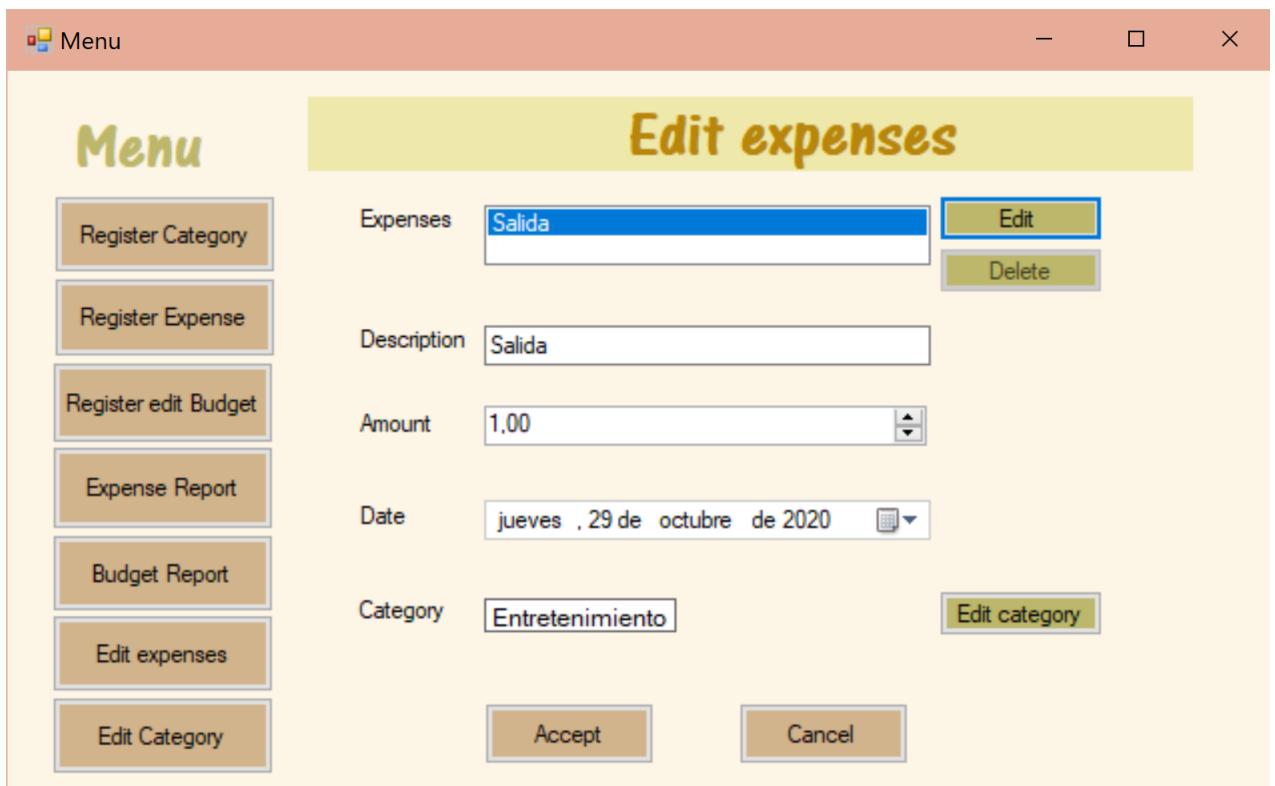
Anexo 12



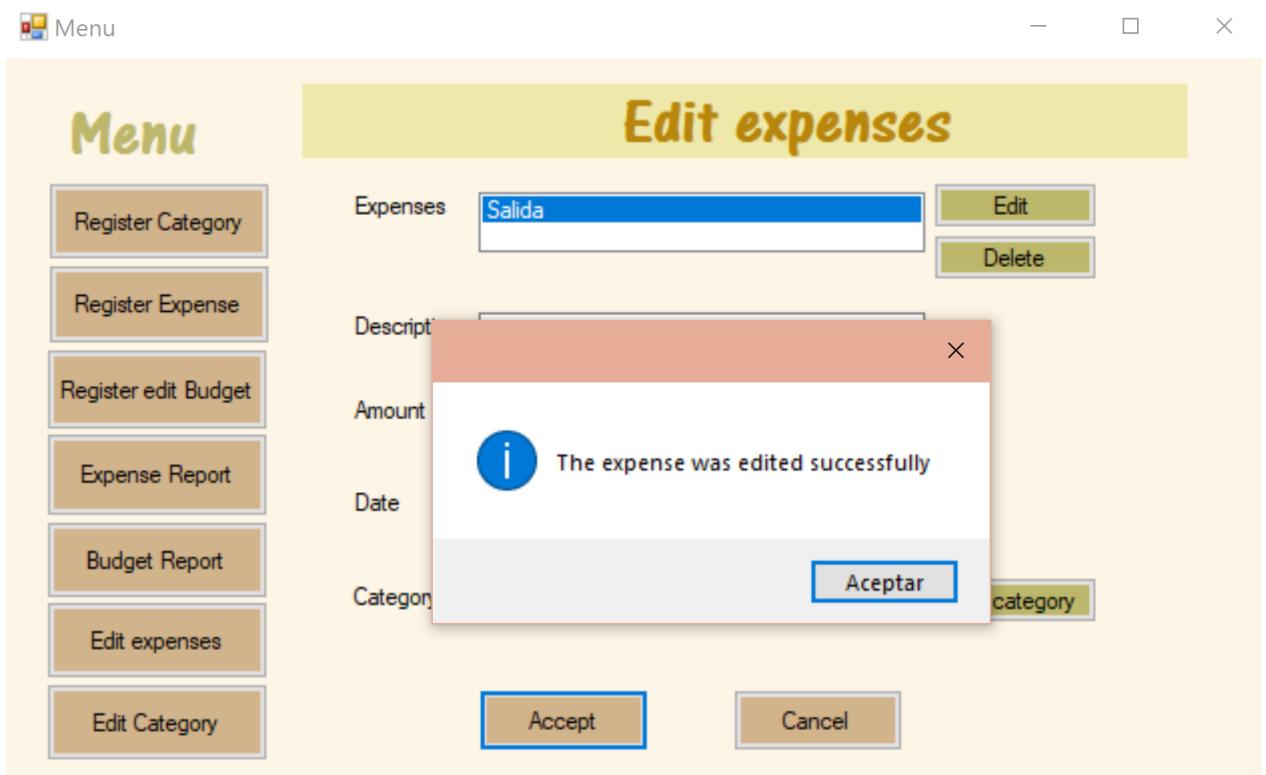
Anexo 13



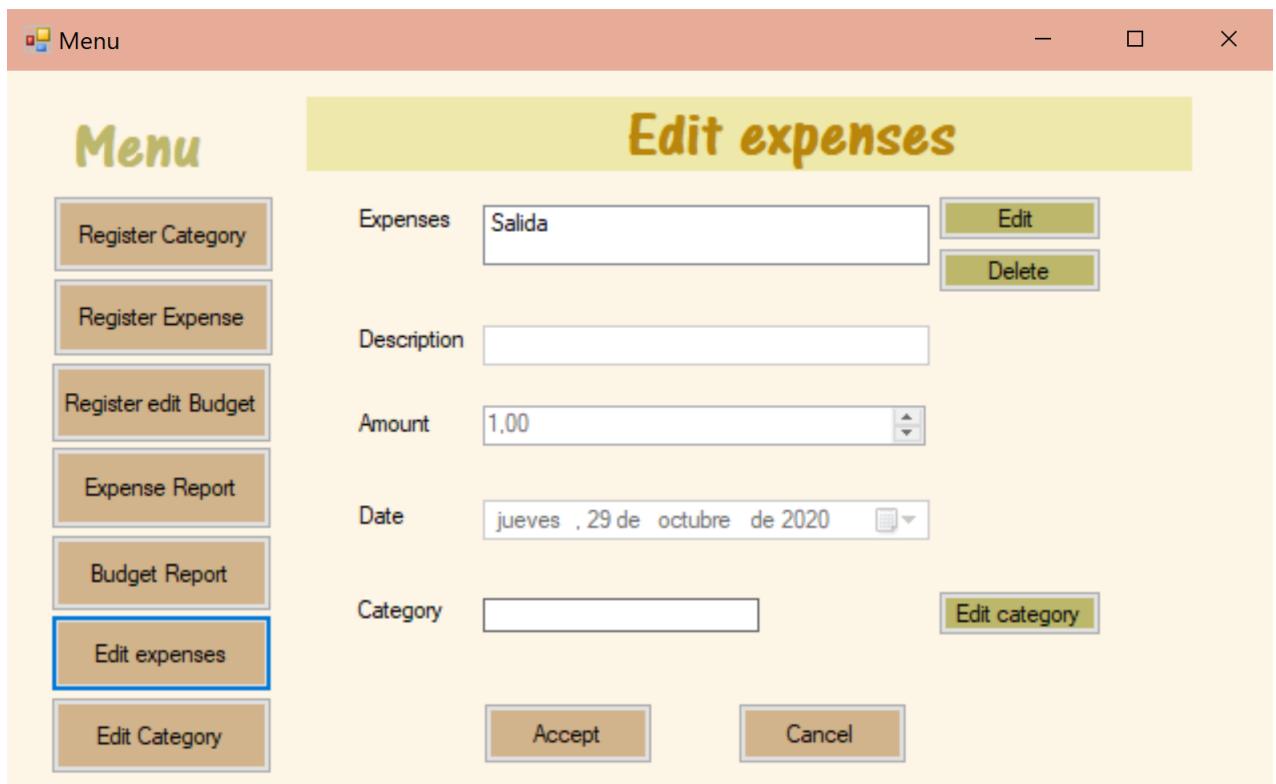
Anexo 16



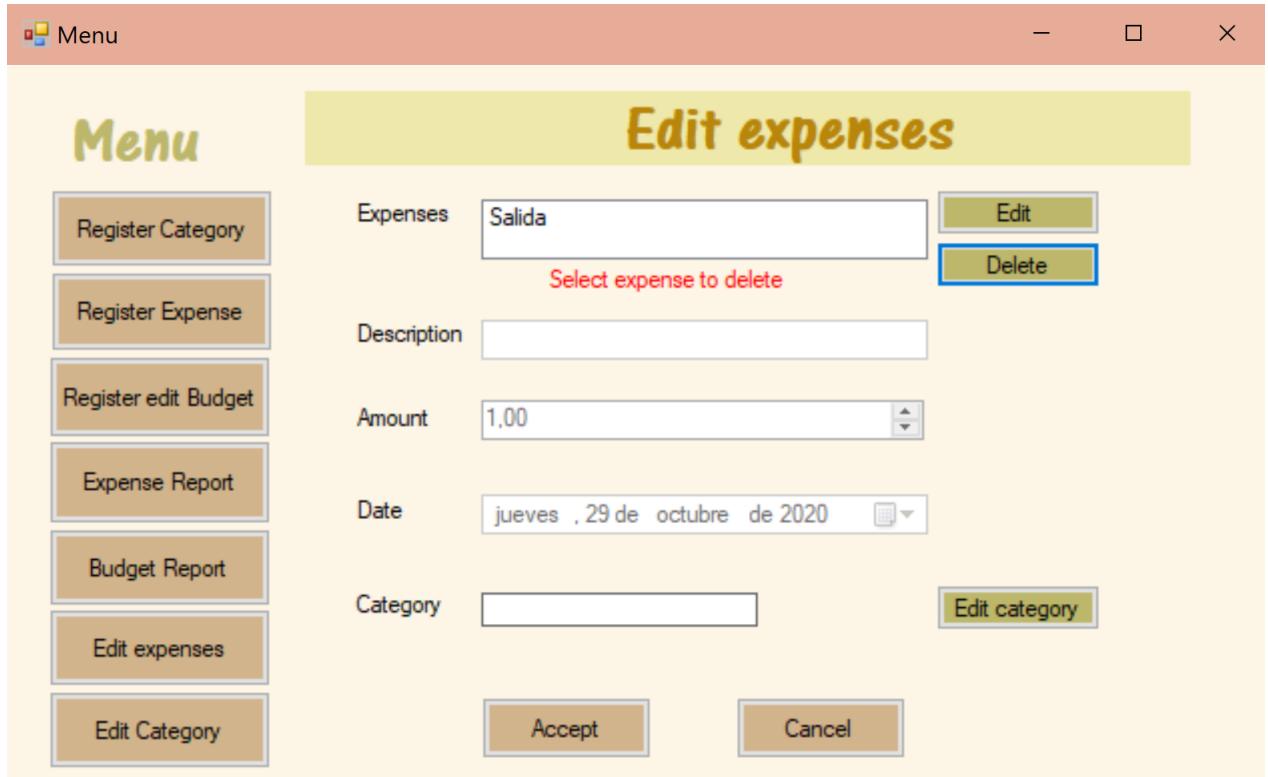
Anexo 17



Anexo 18



Anexo 19



Anexo 20

Menu

Edit expenses

Register Category

Register Expense

Register edit Budget

Expense Report

Budget Report

Edit expenses

Edit Category

Expenses: Salida

Description:

The name must be between 3 and 20 characters long.

Amount: 1,00

Date: jueves , 29 de octubre de 2020

Category: Entretenimiento

Edit category

Accept

Cancel

Anexo 21

Menu

Edit expenses

Register Category

Register Expense

Register edit Budget

Expense Report

Budget Report

Edit expenses

Edit Category

Expenses: Salida

Description: cuando fuimos al cine

The name must be between 3 and 20 characters long.

Amount: 1,00

Date: jueves , 29 de octubre de 2020

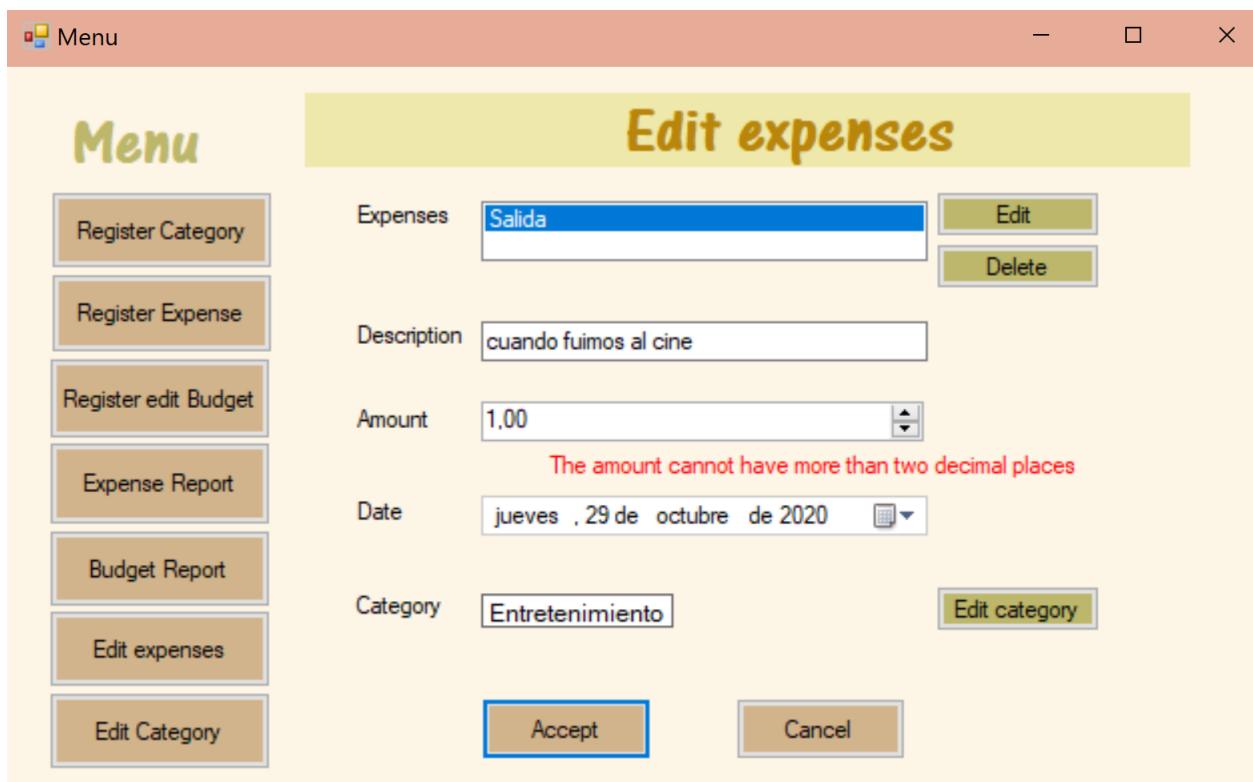
Category: Entretenimiento

Edit category

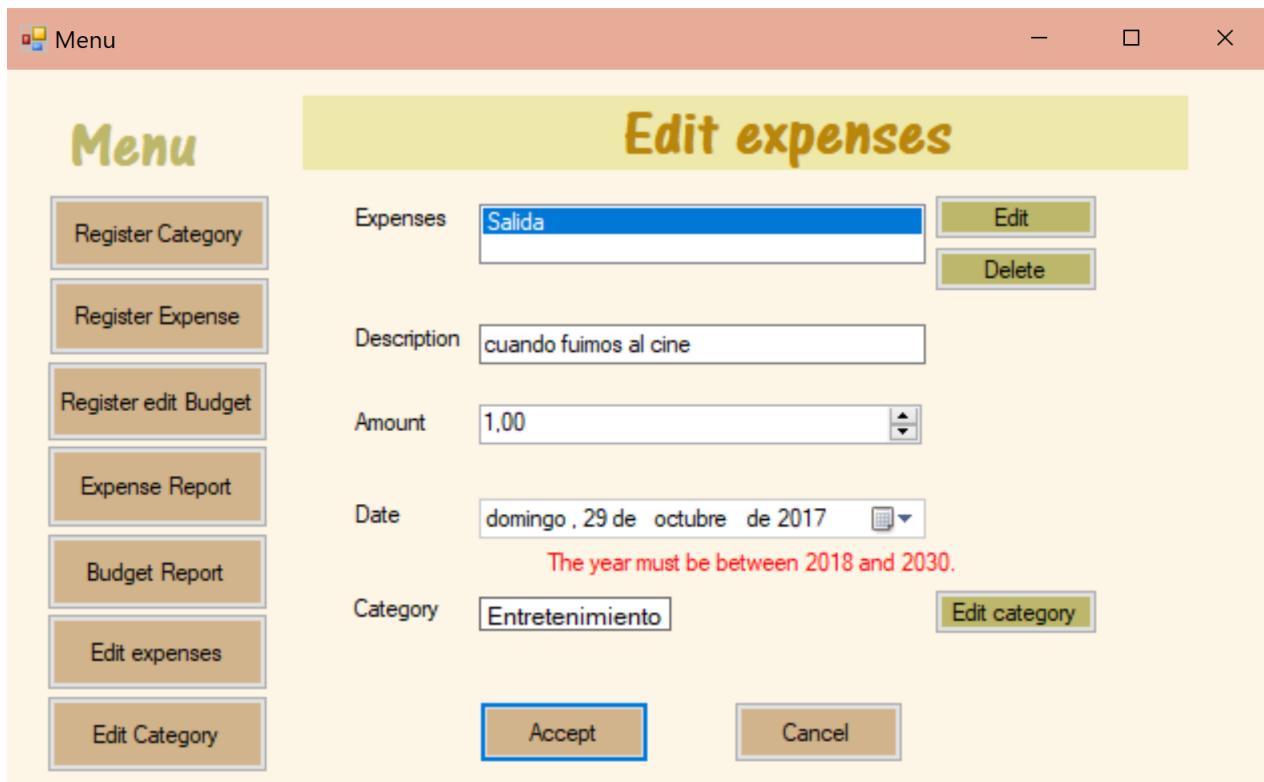
Accept

Cancel

Anexo 22



Anexo 23



Anexo 24

Menu

Edit expenses

Register Category	Expenses	Salida	Edit
Register Expense	Description	cuando fuimos al cine	Delete
Register edit Budget	Amount	1,00	
Expense Report	Date	miércoles, 29 de octubre de 2031	The year must be between 2018 and 2030.
Budget Report	Category	Entretenimiento	Edit category
Edit expenses		Accept	Cancel
Edit Category			

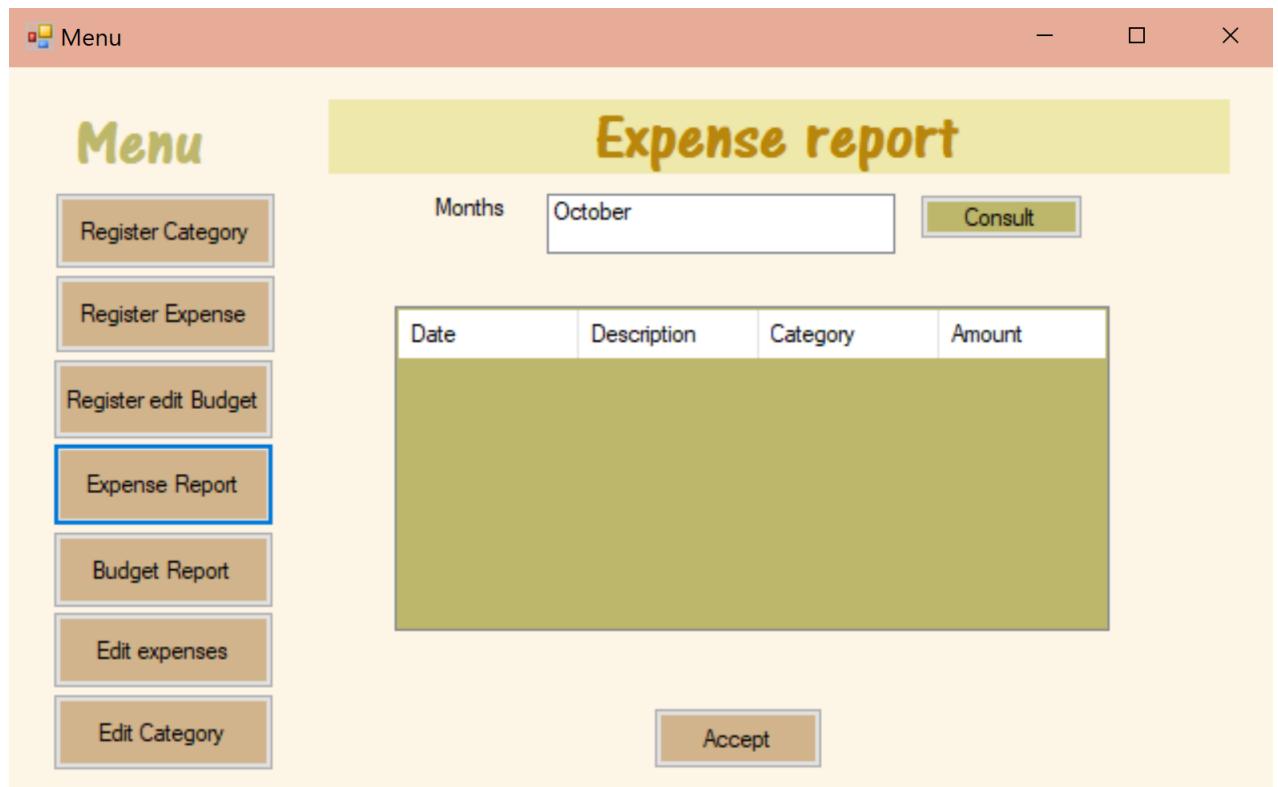
Anexo 25

Menu

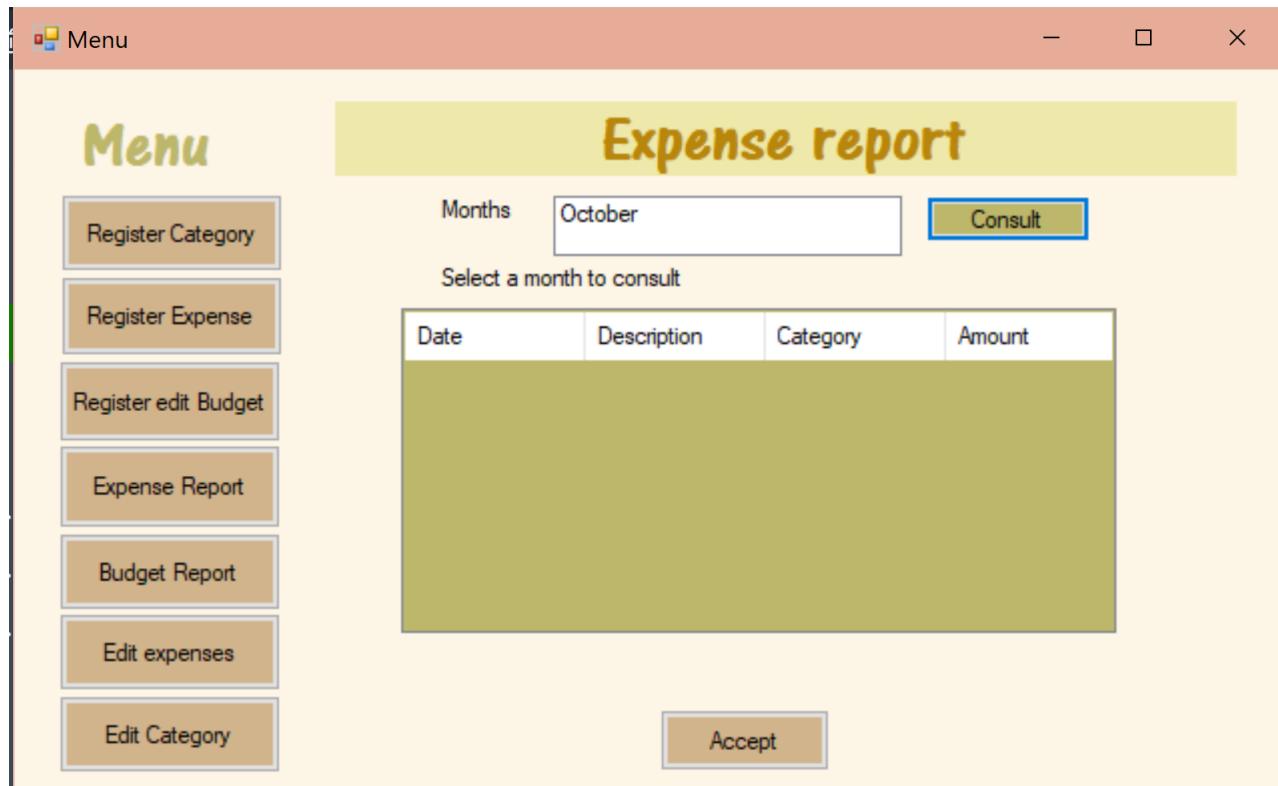
Edit expenses

Register Category	Expenses	Salida	Edit
Register Expense	Description	cuando fuimos al cine	Delete
Register edit Budget	Amount	1,00	
Expense Report	Date	martes , 29 de octubre de 2030	
Budget Report	Category	Entretenimiento	Edit category
Edit expenses		You must select a category	
Edit Category		Accept	Cancel

Anexo 26



Anexo 27



Anexo 28



Anexo 29

