Treball Final de Carrera - .NET

Gestió de despeses personals

Observacions

Universitat Oberta de Catalunya

Xavier Miñana Garcia
16/12/2013
Consultor: Jairo Sarrias Guzman

Índex de continguts

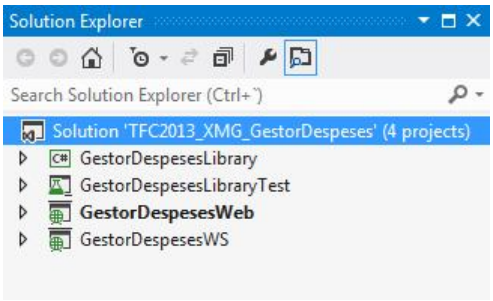
1 Introducció

El present document conté informació rellevant referent al desenvolupament del producte. Per motius de limitació de hardware en la present entrega hi han dues solucions de visual studio amb versions diferents, aquest document descriu el contingut de cadascuna.
Un altre punt que tracta aquest document es la seguretat del sistema implementat i quines són les mesures de seguretat que s'han dut a terme per augmentar-la el màxim possible.
També s'inclou un petit llistat de coses que s'haurien de millorar en el producte però que per raons de temps no s'han pogut dur a terme en la present entrega.

2 Solució Visual Studio 2012

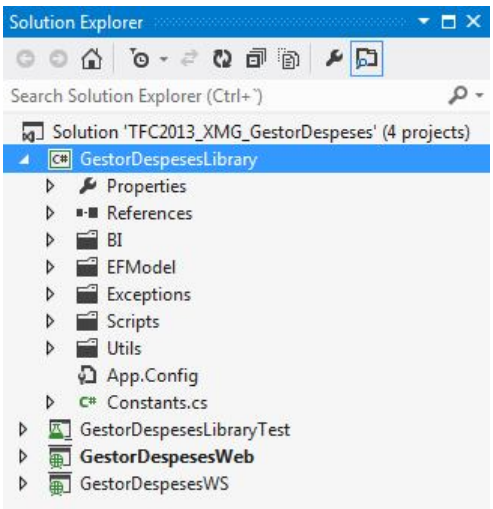
Aquesta solució inclou tot exceptuant l'aplicació per a Windows Phone que s'ha hagut de crear al *Visual Studio 2010 Express for Windows Phone*.

En aquesta solució hi ha inclosos el següents projectes:



2.1 GestioDespesesLibrary

És un projecte que genera com a resultat una llibreria que conté tota la capa d'accés a dades i la capa de negoci de l'aplicació.



BI inclou una sèrie de classes i mètodes per poder dur a terme totes les tasques de negoci que requereix l'aplicació. Les classes son les següents:

- **CoreBI**: Conté mètodes per poder enviar correus electrònics i llegir i modificar la configuració del sistema.

- **ExpensesBI:** Conté mètodes per a gestionar despeses i fulls de despeses, com crear una despesa, modificar l'estat d'un full de despeses...
- **ExpensesTypesBI:** Conté els mètodes per a gestionar tipus de despeses, com crear un tipus de despesa, modificar un tipus de despesa...
- **UsersBI:** Conté mètodes per gestionar els usuaris, com crear usuari, esborrar usuari, modificar usuari...
- **WSSessionsBI:** Conté mètodes per a crear i controlar les sessions del usuaris.

EFModel inclou tota la capa d'accés a dades. Per l'accés a les dades s'ha utilitzat Entity Framework com a ORM i s'ha utilitzat un mode de treball anomenat *Model First*, que consisteix en crear el model i després generar les classes necessàries a partir del model. Aquesta carpeta inclou el model i les classes generades.

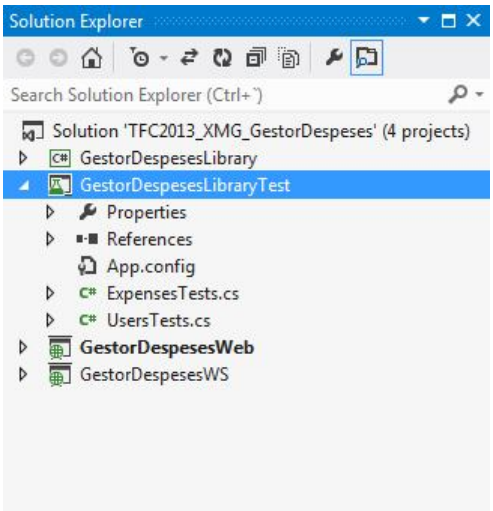
Exceptions inclou dos tipus d'excepció que s'han creat que seran les úniques que la capa de negoci llençarà. Una és per errors controlats (s'intenta fer una acció que no es permet, alguna limitació del model, ...) i l'altre és per quan hi ha un comportament no esperat a la capa de negoci.

Scripts inclou dos scripts sql, un per una vegada generada la base de dades a partir de l'script que genera el Entity Framework, completar-la amb la creació de claus úniques per a certes columnes i l'altre per fer una pre-càrrega de dades a la base de dades.

Utils inclou una classe amb utilitats generals per la resta parts.

2.2 GestioDespesesLibraryTest

Aquest projecte conté probes unitàries del projecte *GestorDespesesLibrary*. És un projecte que està per terminar i només inclou els tests de la part corresponent a Usuaris.

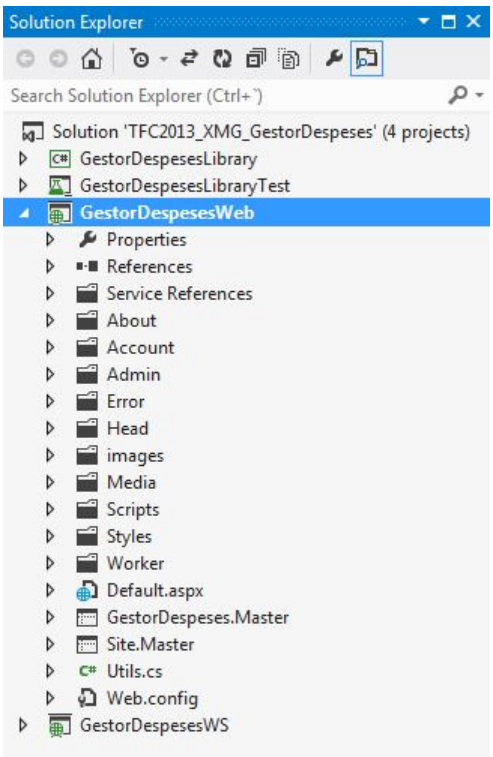


2.3 GestioDespesesWeb

Aquest projecte genera com a resultat una aplicació web que serveix per que els usuaris interactuïn amb el sistema.

Per a cada plana de l'aplicació existeix un fitxer .aspx (que es la part visual) i un fitxer .cs (que seria la part de controlador). Aquesta part de controlador gairebé la única tasca que fa

és enviar peticions als serveis web per a realitzar les accions pertinents i processar la informació rebuda per mostrar-la a les vistes.



Hi ha 6 carpetes que corresponen al mapa de la web:

- **About:** Informació del producte.
- **Account:** Login, LogOff i recuperació de credencials.
- **Admin:** Gestió d'usuaris, pagament de fulls validats i configuració del sistema.
- **Error:** Pàgines d'error de l'aplicació.
- **Head:** Gestió dels fulls de despeses per validar.
- **Worker:** Gestió del full de despeses personal i històric de despeses.

Hi ha dues pàgines "Master":

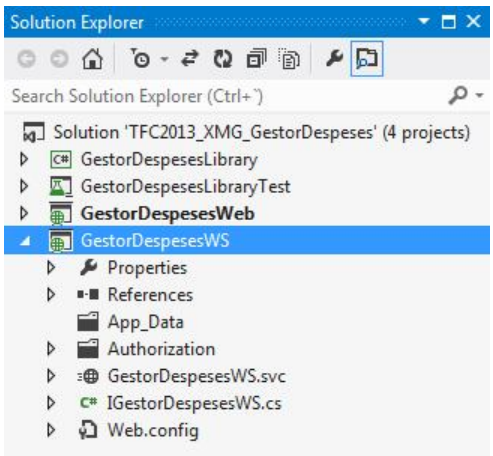
- **Site.Master:** Pàgina màster global per totes les pàgines.
- **GestorDespeses.Master:** Pàgina màster per a aquelles pàgines on l'usuari ja hagi estat autenticat. Aquesta pàgina màster té com a pàgina màster *Site.Master*.

Default.aspx és la pàgina que és carrega per defecte al entrar en la web i té com a única funcionalitat redirigir al usuari a una pàgina o una altre depenent si esta autenticat o no i depenent del rol d'usuari que tingui.

Service References és una referència al servei web.

2.4 GestioDespesesWS

Aquest projecte genera un servei web per exposar remotament tota la funcionalitat de la plataforma i que l'aplicació web i l'aplicació per a Windows Phone puguin realitzar les tasques corresponents.



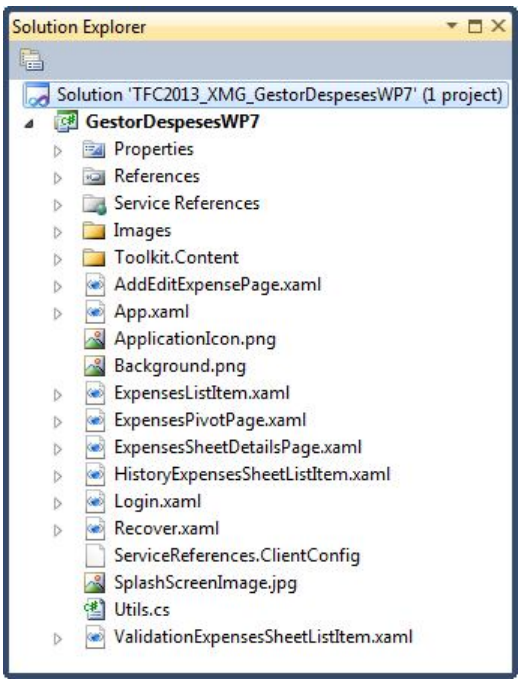
Aquest projecte té una dependència amb el projecte *GestorDespesesLibrary*.

Conté un mòdul d'autorització (que s'explica al apartat *Seguretat*), una classe de tipus interface on es troba tota la definició del servei web i una classe que implementa aquesta interface.

El servei només conté un enllaç (binding) de tipus *basicHttpBinding*, és un tipus d'enllaç sense autenticació ni xifrat de dades. En el apartat de *Seguretat* s'expliquen les mesures adoptades per dotar de seguretat a aquesta part del sistema.

3 Solució Visual Studio 2010

Aquesta solució inclou un únic projecte que correspon a l'aplicació per a dispositius Windows Phone.



Per a cada pantalla de l'aplicació existeix un fitxer .xaml (que es la part visual) i un fitxer .cs (que seria la part de controlador). Aquesta part de controlador fa crides al servei web depenent de les tasques que hagi de fer i modifica la informació que es mostra a la part visual.

Aquest projecte té dues dependències externes que son:

- **Microsoft.Phone.Controls.Toolkit:** Llibreria de Silverlight per a Windows Phone (versió de Octubre 2011). En el projecte s'utilitza per els camps de tipus autocompletar i per els camps de tipus data.
- **Coding4Fun.Toolkit.Controls:** Llibreria que s'utilitza per el control de Loading que apareix quan s'estan carregant dades del servei web, el about i per el dialog que conté un textbox per poder rebutjar una despesa.

Tots dos toolkits necessiten que al projecte s'inclogui la carpeta *Toolkit.Content* que conté dues imatges per els botons.

A imatges hi ha les imatges dels botons de la barra d'aplicacions que apareix en l'aplicació.

Service References és una referència al servei web.

4 Seguretat

4.1 Autenticació

Només existeix un endpoint per connectar els clients del servei web al servei, i té un binding de tipus *basicHttpBinding*. Aquest tipus de binding no incorpora cap tipus de seguretat, ni a nivell d'autenticació ni a nivell de xifratge de dades, per tant qualsevol amb connectivitat al servei podria fer crides als mètodes que els servei exposa sense cap tipus d'autorització.

El servei web exposa diferents mètodes, hi ha uns quants que no requereixen autenticació (login i tots els referents a la part de recuperació de credencials) i per la resta si que es requereix.

Hi ha dues solucions que a aquest problema:

1. Desenvolupar un servei web amb dos contractes i associar cadascun a un endpoint diferent. Un endpoint tindria un binding de tipus *basicHttpBinding* per les operacions de recuperació de credencials (login no faria falta perque ja ho incorporaria el binding de l'altre endpoint). L'altre endpoint s'hauria d'escollir un dels tipus de seguretat que proporciona WCF per serveis web, per exemple *wsHttpBinding* que ja incorporaria la part d'autenticació, confiabilitat i seguretat de la missatgeria. Aquesta solució implica:
    - Generar un certificat i fer que els clients confiïn en aquest.
    - Desenvolupar un mòdul d'autenticació custom que faci l'autenticació dels usuaris.
    - Configurar dos endpoints i separar el desenvolupament en dos contractes.
    - Escollir un tipus de context per el servei que suporti sessions i peticions concurrents.

2. Desenvolupar el nostre propi sistema d'autenticació i control de sessions. En el propi endpoint que ja disposem s'ha de generar un mètode per login i un altre per logout.

El mètode de login accepta usuari i contrasenya i retorna com a resultat satisfactori un token de sessió que s'ha d'incorporar com a paràmetre en tots aquells mètodes que requereixen autenticació. El token té un temps de vida determinat que es reinicia cada cop que es rep una petició amb aquest token. L'únic que s'ha de tenir en conte és que la comunicació ha d'anar xifrada d'alguna manera, això ho aconseguim modificant el binding del endpoint, com per exemple modificant la propietat *BasicHttpSecurityMode* del nostre binding a *Transport*, que ens ofereix confidencialitat e integritat de la missatgeria. Aquesta solució implica:

- Generar un sistema de control de sessions.
- Afegir al contracte els mètodes de login i logout.
- Modificar la resta de mètodes del contracte per que acceptin un paràmetre nou que serà el token de sessió.
- Modificar el mode de seguretat del binding.
- Generar un certificat i fer que els clients confiïn en aquest.

Per raons de temps s'ha escollit la segona opció, a falta de modificar el binding per que la missatgeria vagi xifrada. En el seu moment vaig determinar que el cost d'utilitzar una o una altre és semblant però que en la segona, en cas no arribar a temps per l'entrega em costaria menys desfer que no pas la primera.

Per un projecte no acadèmic segurament hagués optat per la primera opció i així utilitzar les eines que ofereix WCF per aquests temes.

4.2 Autorització

S'ha implementat un mòdul d'autorització al servei web per tal que, quan s'intenti cridar a una operació del contracte i a partir del token i dels paràmetres de la pròpia crida, es pugui determinar si es té autorització per realitzar la operació o no.

En concret hi ha 4 mètodes per autoritzar.

- **isAllowed:** amb el token i el tipus d'usuari que es requereix per fer la operació, determina si l'usuari del token es del tipus d'usuari específic.
- **isAllowed:** amb el token i un identificador d'usuari, determina si l'usuari del token té accés a les dades del usuari corresponent al identificador.
- **isAllowedExpenseSheet:** amb el token i un identificador de full de despeses, determina si l'usuari del token té accés a les dades del full de despeses corresponent al identificador.
- **isAllowedExpense:** amb el token i un identificador de despesa, determina si l'usuari te accés a les dades de la despesa corresponent al identificador.

5 Millores

- Modificar el mode de seguretat del binding a *Transport* per garantir la confidencialitat de les peticions.
- Afegir un nou endpoint amb un binding de tipus *netNamedPipeBinding* amb el mateix contracte al servei web, per tal que la web que s'allotja a la mateixa màquina es connecti al servei web mitjançant aquest canal. Per serveis que es troben en una mateixa màquina aquest tipus de binding és el més eficient.

- Que l'usuari pugui modificar les seves credencials quan està autenticat. Ara mateix la contrasenya només es pot canviar utilitzant la funcionalitat de recuperar credencials o demanant a l'administrador que la canviï.
- Que l'usuari pugui eliminar la imatge d'una despesa. Ara mateix es pot modificar la imatge d'una despesa però no es pot eliminar. Per fer-ho s'ha d'esborrar la despesa i crear-la de nou sense afegir-hi la imatge.
- Que un usuari de tipus Responsable (en validacions) i un usuari de tipus Administrador (en pagaments) puguin veure la imatge d'una despesa.
- Al visualitzar les despeses d'un full de despeses veure una columna que indiqui si la data correspon a un dia del cap de setmana o no.
- Al generar una despesa i seleccionar una data que es mostri si és festiu o cap de setmana i que es pugui veure l'increment que s'aplicarà en aquest cas.
- Afegir un sistema de notificacions a l'aplicació de Windows Phone per rebre alertes de quan un full de despeses ha estat validat, rebutjat o pagat. També per aquells usuaris de tipus Responsable notificar-los que un usuari al seu càrrec ha posat un full de despeses a validar.
- Afegir comprovacions bàsiques en els inputs dels formularis. Ara mateix no es comprova en client les coses bàsiques, com que el camp estigui buit, que camp numèric de text sigui un número o que una direcció de correu tingui el format correcte, tot això es comprova a la part servidor però es pot reduir tràfic innecessari fent aquestes comprovacions.
- Fer que el mòdul d'autorització sigui més complert, afegint més condicions com per exemple si s'accedeix per lectura o per lectura i escriptura.
- Finalitzar el projecte de *GestorDespesesLibraryTest* afegint-hi els tests unitaris que cobreixin tota la funcionalitat de la llibreria.
- Configurar el plug-in TableTools per poder exportar el contingut d'una taula a diferents formats o enviar a imprimir. Els components necessaris es troben dins de la solució al projecte *GestorDespesesWeb*, però les proves que s'han fet són funcionalment correctes però provoquen que es desquadrin les taules.