**Milestone Report 2**

****Law Firm BPR Team****

****Project Name: Law Firm Billing Project****

****Report Date: November 29, 2012****

Team Members:

Albert Badalyan Database Administrator

Colin Kay Project Manager

Kaleb Martens GUI Developer

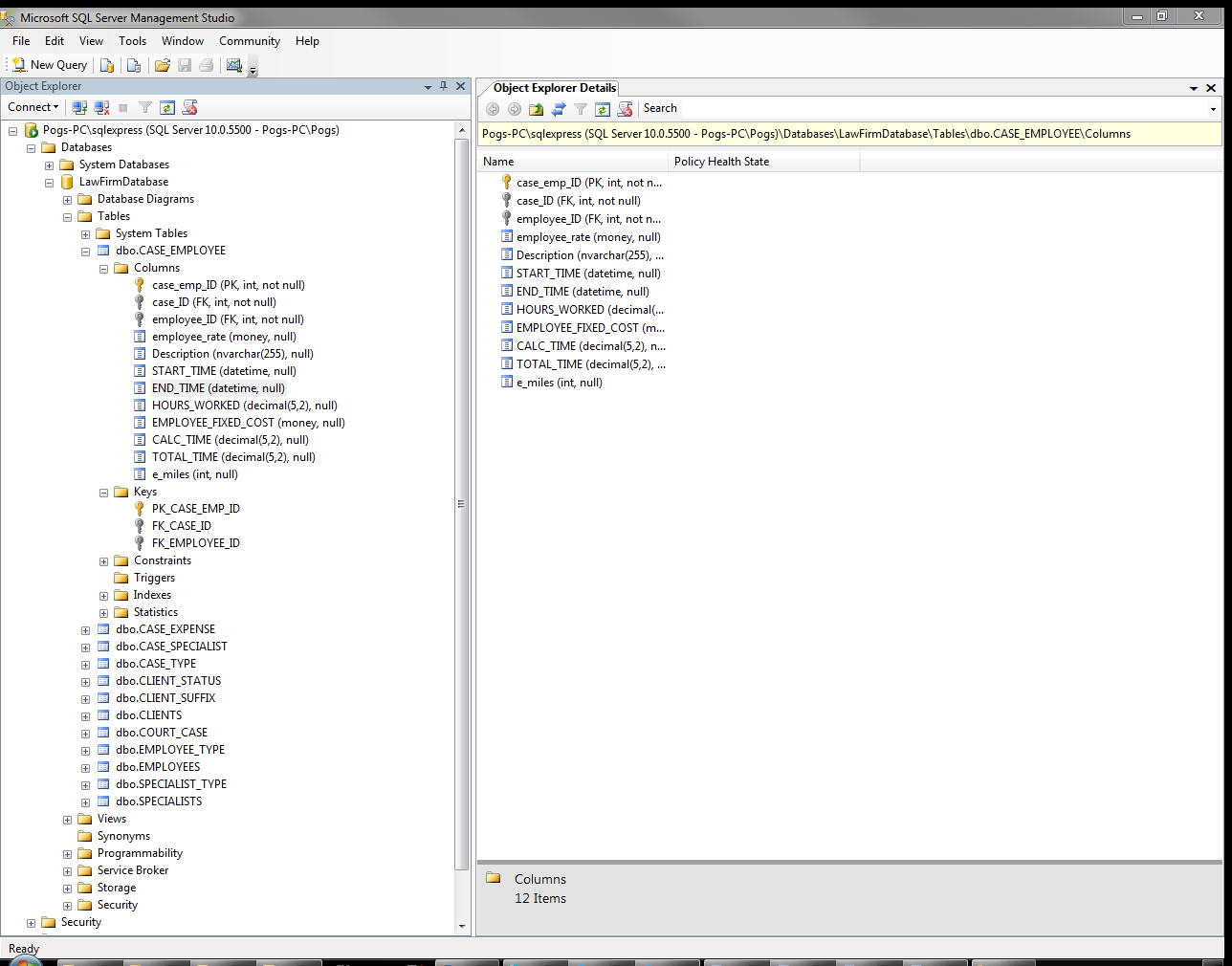
Pagoda Pang Technical Writer

Back End Design:

Much has changed within the last month, we figured out how things are calculated within the billing process. Our database has change drastically we added some new fields and even took out some. Calculated fields have been placed in the system, time can be tracked and funds can be viewed live. We also implemented the report feature within the application. Clicking on the button will launch the report feature, the report will display with the available data that has been inputted.

Details of what we have done from the back end of things are explained throughout thereport.

This is what we have currently in our database, things were added and some were removed. Throughout the project this was a normal routine for us, as we would go back and forth discussing what is really needed. It was hard for us to understand all of the billing details, even now we are baffled on how much goes on during this process.



Provided is the formula we will use to calculate the expenses per case, then identify the amount of funds available left for that case.

|  |  |  |  |
| --- | --- | --- | --- |
| Case Employee | | | |
| Start Time/End Time | Hours Worked | Hourly | Fixed Fee |
|  |  |  |  |
| Total Case Employee Cost per Case | | | |

|  |  |
| --- | --- |
| Specialist | |
| Hours | Cost |
|  |  |
| Total Specialist Cost + Total Case Employee Cost | |

|  |
| --- |
| Expenses |
| Cost |
|  |
| Total Expenses + Total Specialist Cost + Total Case Employee Cost |

The will be deducted from total fund amount to determine how much money is allocated per case.

Provided is the VB code which utilizes the expense formulas to perform the following:

1. Calculate the total cost for case employees, specialists, and any additional expenses, then display the calculated amount on the Court Case screen as “Expenses”
2. Deduct the total expenses from the amount allocated to that case type, then display the calculated amount on the Court Case screen as “Money Pool”

Namespace LightSwitchApplication

Public Class COURT\_CASE

Private Sub Total\_Expenses\_Compute(ByRef result As Decimal)

' Set result to the desired field value

Dim expenses As Integer

Dim emp\_fees As Integer

Dim emp\_fixed\_cost As Integer

Dim spc\_fees As Integer

Dim spc\_fixed\_cost As Integer

' Calculate fixed and hourly fees for employees

emp\_fees = CASE\_EMPLOYEEs.Sum(Function(f) f.Employee\_Rate \* f.CALC\_TIME)

emp\_fixed\_cost = CASE\_EMPLOYEEs.Sum(Function(g) g.EMPLOYEE\_FIXED\_COST)

' Calculate additional case expenses

expenses = CASE\_EXPENSEs.Sum(Function(e) e.expense\_cost)

' Calculate fixed and hourly fees for specialists

spc\_fees = CASE\_SPECIALISTs.Sum(Function(s) s.Specialist\_Rate \* s.hours)

spc\_fixed\_cost = CASE\_SPECIALISTs.Sum(Function(sp) sp.SPECIALIST\_FIXED\_COST)

result = expenses + emp\_fees + emp\_fixed\_cost + spc\_fees + spc\_fixed\_cost

End Sub

Private Sub Money\_Pool\_Compute(ByRef result As Decimal)

' Set result to the desired field value

If CASE\_TYPE Is Nothing Then

result = 0

Else

result = CASE\_TYPE.amount - Total\_Expenses

End If

End Sub

To launch the sp\_CalcTime stored procedure, VB code was applied to the sp\_CourtCase application table and Court\_CaseDETAIL screen . Provided is the VB code for the sp\_CalcTime application table:

Imports System.Data

Imports System.Data.SqlClient

Namespace LightSwitchApplication

Public Class ApplicationDataService

Private Sub sp\_CalcTime\_Inserting(entity As sp\_CalcTimeItem)

Using connection = New SqlConnection

Dim connectionStringName = Me.DataWorkspace.LawFirmData.Details.Name

connection.ConnectionString =

ConfigurationManager.ConnectionStrings(connectionStringName).ConnectionString

Dim procedure = "dbo.sp\_CalcTime"

Using command = New SqlCommand(procedure, connection)

command.CommandType = CommandType.StoredProcedure

connection.Open()

command.ExecuteNonQuery()

End Using

End Using

Me.Details.DiscardChanges()

End Sub

Provided is the VB for the Court\_CaseDETAIL screen that is executed when a record is being saved:

Private Sub Calc\_Time\_Execute()

' Write your code here.

Dim dataWorkspace = New DataWorkspace

Dim courtcase = Me.COURT\_CASEcase\_ID

Dim operation =

dataWorkspace.ApplicationData.sp\_CalcTime.AddNew()

operation.CaseID = courtcase

dataWorkspace.ApplicationData.SaveChanges()

End Sub

Private Sub COURT\_CASEDetail\_Saved()

' Write your code here.

Dim dataWorkspace = New DataWorkspace

Dim courtcase = Me.COURT\_CASEcase\_ID

Dim operation =

dataWorkspace.ApplicationData.sp\_CalcTime.AddNew()

dataWorkspace.ApplicationData.SaveChanges()

End Sub

Provided is the sp\_CalcTime stored procedure that is executed when a new expense record is saved on the Court\_CaseDETAIL screen:

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_CalcTime] Script Date: 11/28/2012 8:40:24 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER procedure [dbo].[sp\_CalcTime]

AS

/\*

Changelog:

Date Comment

------------------------------------------------------------------------------------------------------------

10/18/12: Currenly can calculate the time difference for a single Case\_Emp\_ID that is provided in the script.

Ultimately, this script will calculate the CALC\_TIME for each record in the dbo.case\_employee table.

10/28/12: Procedure will now perform the date-time calculation for any existing record with a CASE\_EMP\_ID. Also

will update the CALC\_TIME for each record unless the calculated time is < .5.

------------------------------------------------------------------------------------------------------------

\*/

/\* Declare variables to be used for calculations. \*/

DECLARE @sql1 nvarchar(255), @sql2 nvarchar(255), @date1 nvarchar(60), @date2 nvarchar(60), @calc decimal(7,2)

DECLARE @retval1 int, @retval2 int, @ParmDef1 nvarchar(255), @ParmDef2 nvarchar(255)

DECLARE @CaseEmpID int

/\* Declare curor. \*/

DECLARE CaseEmpID CURSOR FOR

SELECT case\_emp\_id FROM [dbo].[case\_employee];

OPEN CaseEmpID

FETCH NEXT FROM CaseEmpID

INTO @CaseEmpID

WHILE @@FETCH\_STATUS = 0

BEGIN

/\* Prepare query to capture output for @date1 (START\_TIME) from sp\_executesql. \*/

SET @sql1 = 'select @date1OUT = START\_TIME from [dbo].[case\_employee] where case\_emp\_ID =' +Convert(nvarchar,@CaseEmpID)+''

SET @ParmDef1 = N'@date1OUT datetime OUTPUT'

/\* Execute @sql1 and capture the output into @date1 (START\_TIME) for later calculations. \*/

EXEC sp\_executesql @sql1, @ParmDef1, @date1OUT=@date1 OUTPUT;

/\* Prepare query to capture output for @date2 (END\_TIME) from sp\_executesql. \*/

SET @sql2 = 'select @date2OUT = END\_TIME from [dbo].[case\_employee] where case\_emp\_ID =' +Convert(nvarchar,@CaseEmpID)+''

SET @ParmDef2 = N'@date2OUT datetime OUTPUT'

/\* Execute @sql1 and capture the output into @date2 (END\_TIME) for later calculations. \*/

EXEC sp\_executesql @sql2, @ParmDef2, @date2OUT=@date2 OUTPUT;

/\* Calculate and store the difference between the START\_TME and END\_TIME.

Also, divide the minutes calculated by 60 and convert to float for preserve decimal values.\*/

set @calc = Convert(Float,DATEDIFF(minute, @date1, @date2)/60.0)

/\*

Append the calculated difference to the record with matching case\_emp\_ID.

Update the CALC\_TIME if @calc is greater, or equal to, .5. Otherwise, notify user.

\*/

IF @calc >= .5

BEGIN

update [dbo].[case\_employee]

set CALC\_TIME = @calc

where case\_emp\_ID = @CaseEmpID

END

ELSE

update dbo.case\_employee

set CALC\_TIME = NULL

where case\_emp\_ID = @CaseEmpId

--PRINT 'The following Case\_Emp\_ID was not updated due to a negative value (END\_TIME not present):' + Convert(varchar(10),@CaseEmpID)

FETCH NEXT FROM CaseEmpID INTO @CaseEmpID

END

CLOSE CaseEmpID

DEALLOCATE CaseEmpID

For the users to be able to render Court appointed report, the following VB script was developed. This code allows the user to utilize a “Launch Report” link on the Court\_CaseDETAIL screen, which launches the court appointed report within the default internet browser. Once the report is loaded, the user can specify the specific date range of expenses that will be included on the report.

Private Sub Print\_Execute()

' Write your code here.

Dispatchers.Main.BeginInvoke(

Sub()

Dim case\_ID

case\_ID = Me.COURT\_CASEcase\_ID

' Provide the URL for the report that you want to view

Dim uri As New Uri("http://amb-t61/ReportServer\_EXPADV/Pages/ReportViewer.aspx?%2fReport+Project7%2fReport1&rs:Command=Render&CASE\_ID=" & case\_ID)

If (AutomationFactory.IsAvailable) Then

' This is a desktop app, so shell to the default browser

Dim shell = AutomationFactory.CreateObject("Shell.Application")

shell.ShellExecute(uri.ToString)

ElseIf (Not System.Windows.Application.Current.IsRunningOutOfBrowser) Then

' This is a web app, so navigate to the page

System.Windows.Browser.HtmlPage.Window.Navigate(uri, "\_blank")

End If

End Sub)

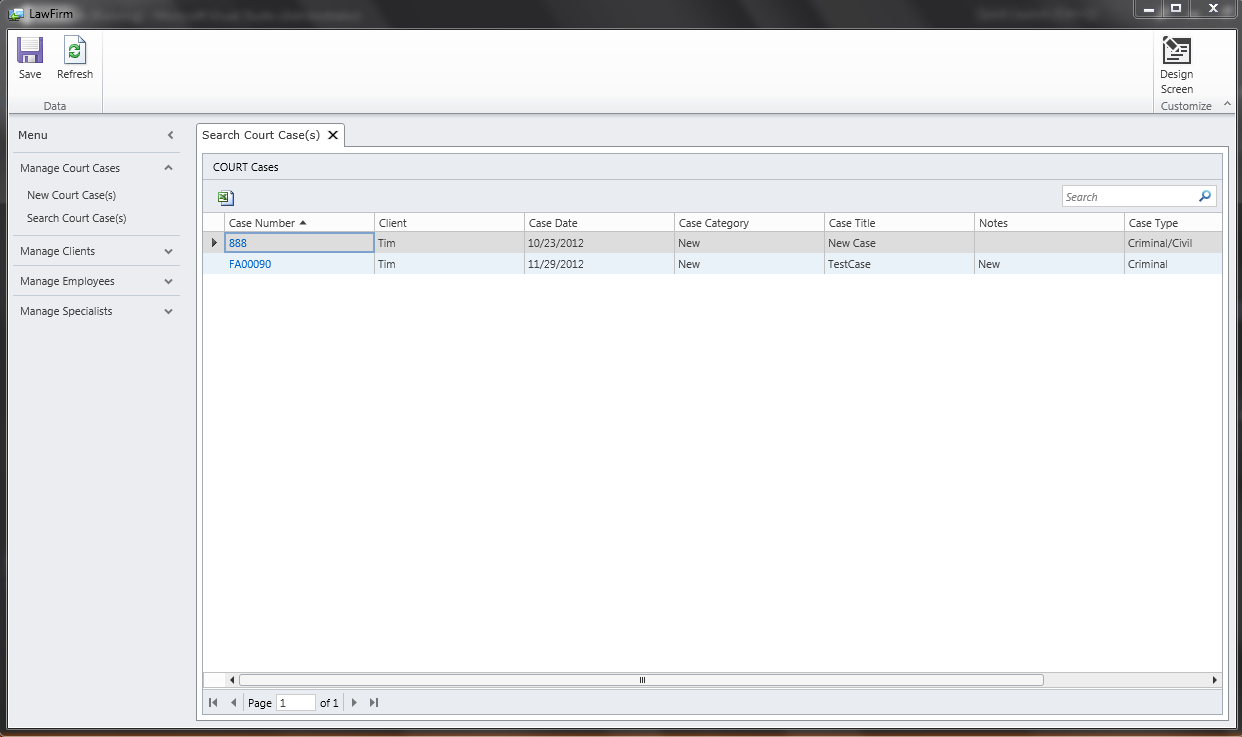
End Sub

GUI Design:

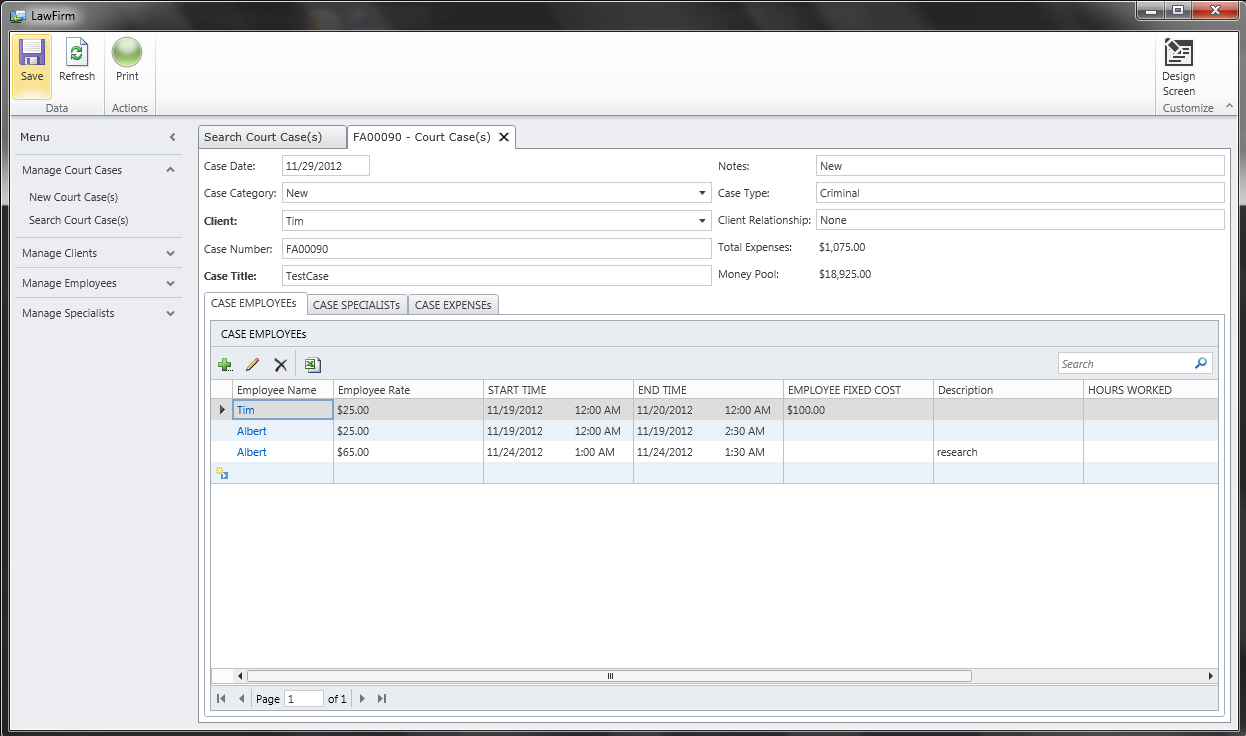
The front end of the application hasn't changed much in look and feel. However we did rearrange some of the fields and buttons to create a better workflow. Everything is consolidated under a billing category our sponsors use. Categories are broken up into specific topics such as cases, clients, employees, and specialists.

We were able to display the current amount of funds left in a case. The neat part about this is that calculations are done live and the user can see the funds decrease as they enter in more expenses. There is still some cleaning up to do, overall the scope requirements have been met.

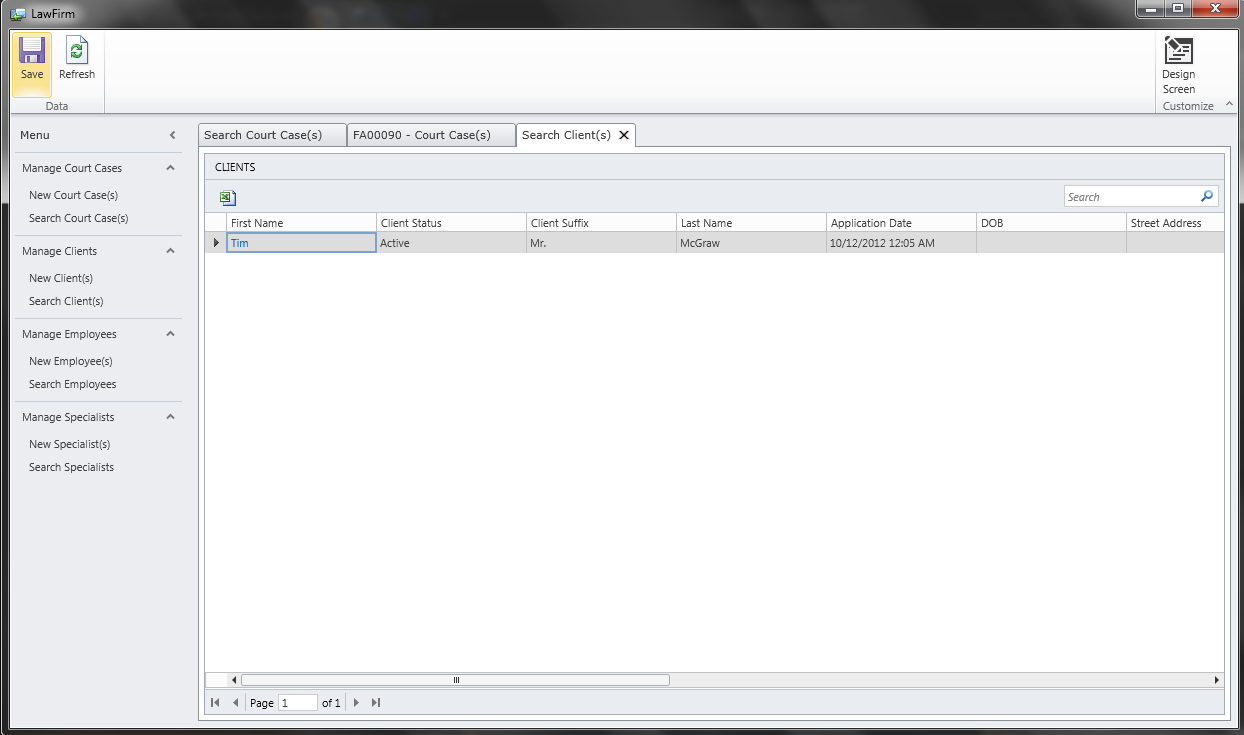
Court Case search screen:



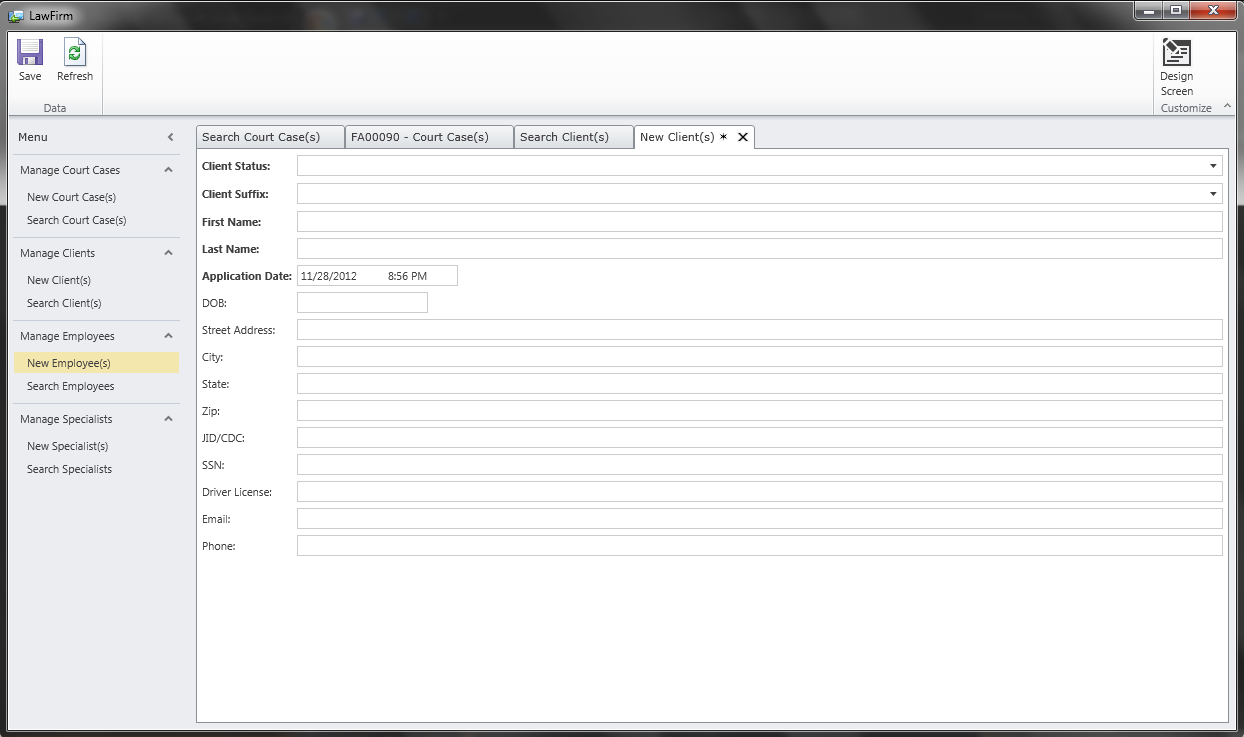
Court Case detail screen which shows Money Pool and Expenses:



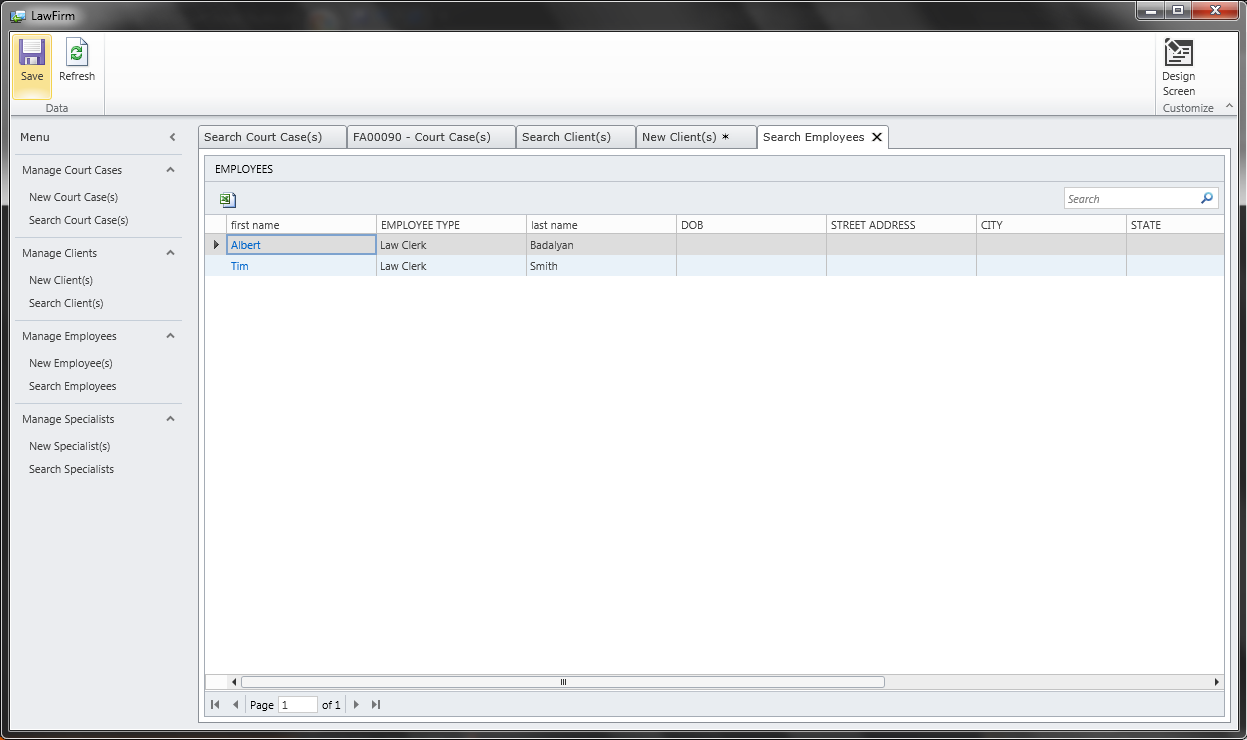
Client search screen:



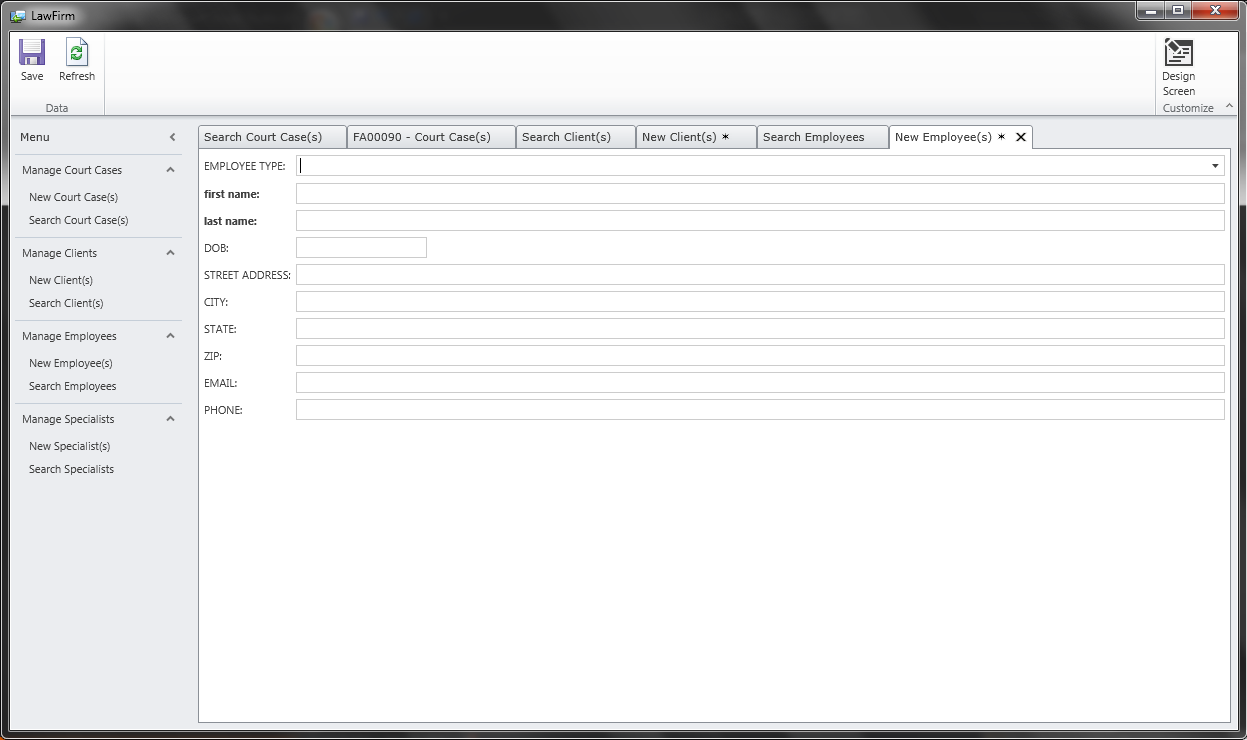
New client data entry screen:



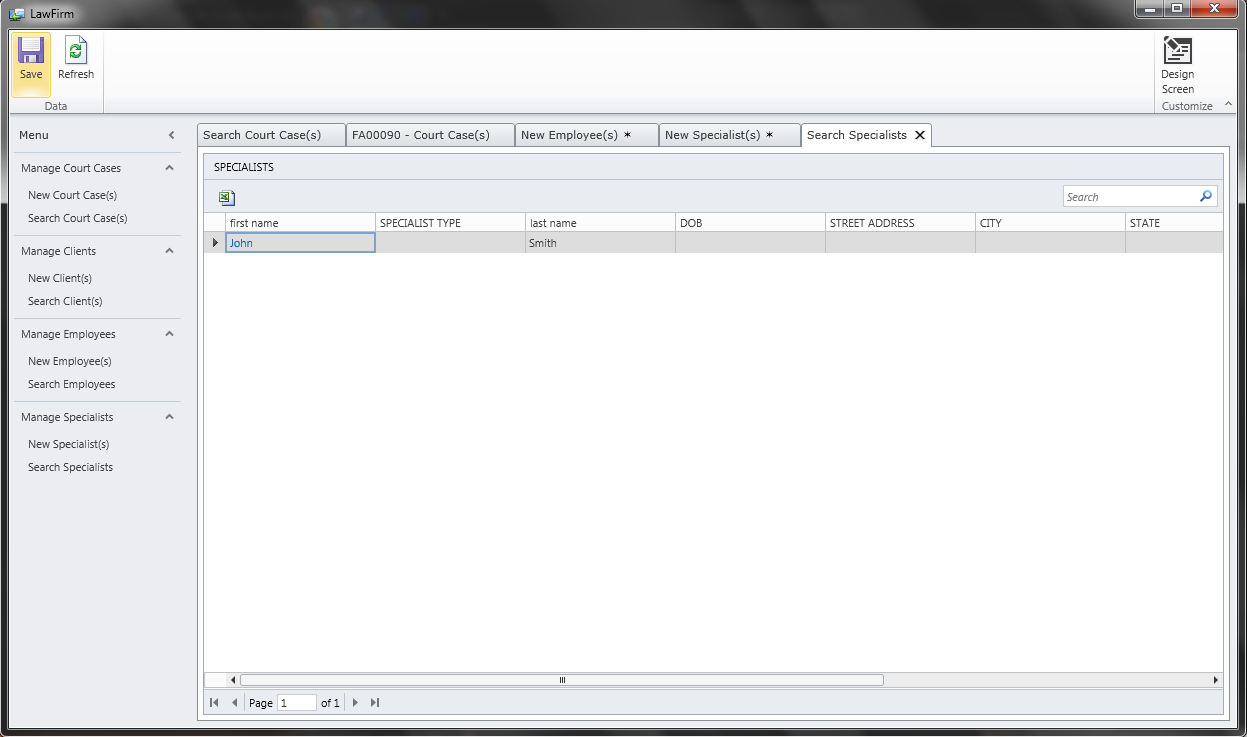
Employee search screen:



New employee data entry screen:



Specialist search screen:



New specialist data entry screen:

