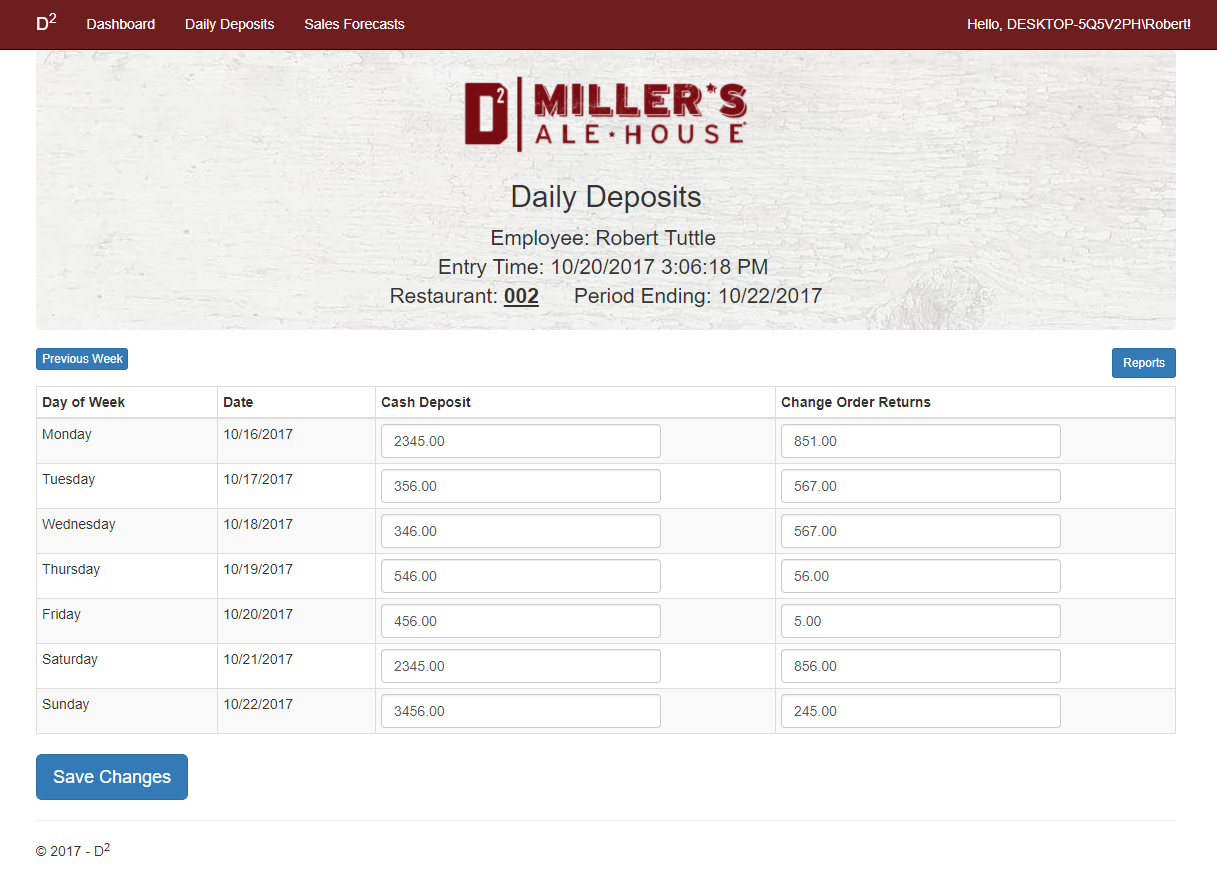
**DSquared Deposit Application**

In summary, this application is a simple tool that allows store managers for a Miller’s Ale House location to record daily deposit records for their associated restaurant. The application is intended to be hosted on an intranet environment and subsequently authorizes each user with Windows Authentication to supply relevant details for application processes and queries. Once a user is successfully authenticated, a query is made against an external database that hosts information about the Employees that log into this application (e.g. Restaurant Number). This information is then used to populate the internal database associated with this application that creates or updates weekly records for Cash Deposits and Miscellaneous Deposits associated with the Employee that is currently authorized via Windows.

**Daily Deposit Data Entry Page**

This is the first page that appears when accessing the application after being successfully authorized via Windows Authentication.



The following details are supplied at the top of the page:

* The DSquared Logo for Miller’s Ale House
* The name of the application: Daily Deposits
* The current Employee that is logged in via Windows Authentication
* The time that the user initially accessed this page
* The unique Restaurant Number associated with the authenticated Employee
* The date in which the weekly deposit period ends

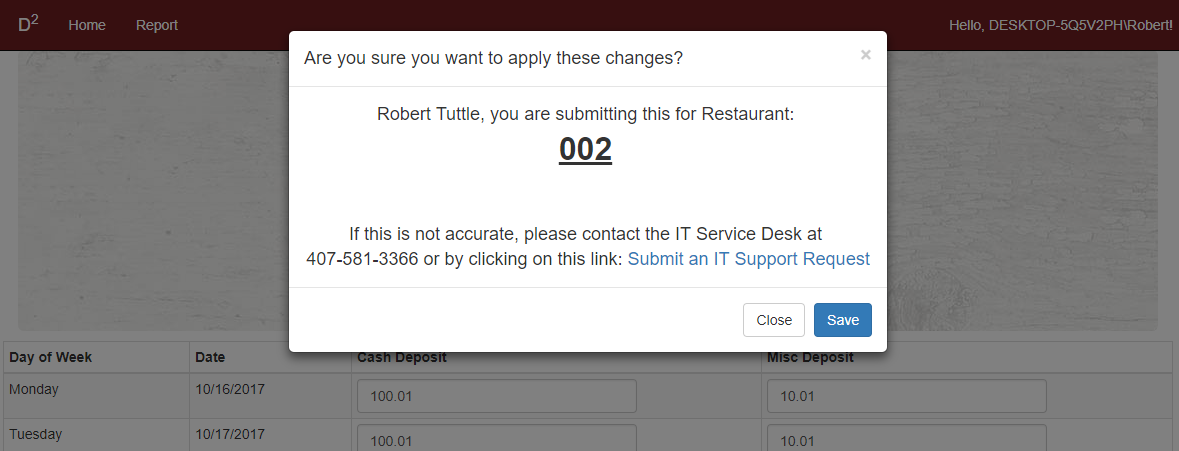
Below, there is a table that displays the day of the week, the date, and two fields for data entry (cash deposits and miscellaneous deposits). In the code implementation, the current week is determined (Monday through Sunday), and subsequently retrieves a set of pre existing records of deposits for each day in the specified week range. The first time a user saves data for the week, every record for each day is inserted into the database. After the first save, the user is just updating the pre existing records that were created on the initial save.

Immediately above the table that allows the user to record deposit information, there are two buttons labeled “Previous Week” and “Reports.”

The “Previous Week” button will allow the user to go back to the previous week of deposit entries for editing; however, this functionality restricts the user to only access the week immediately before the current week. This is controlled by logic based off of the current weekday when the user is accessing this feature; there is no way for a front end user to manipulate this function in any way, and subsequently access records further than the week that immediately precedes the current week. Additionally, once on the screen that is displaying the previous week, a button named “Current Week” will replace “Previous Week,” allowing the user to easily navigate back to the current week of deposit entries.

The “Report” button will allow a user to navigate to the reporting page, and this feature is detailed in the next section of this document.

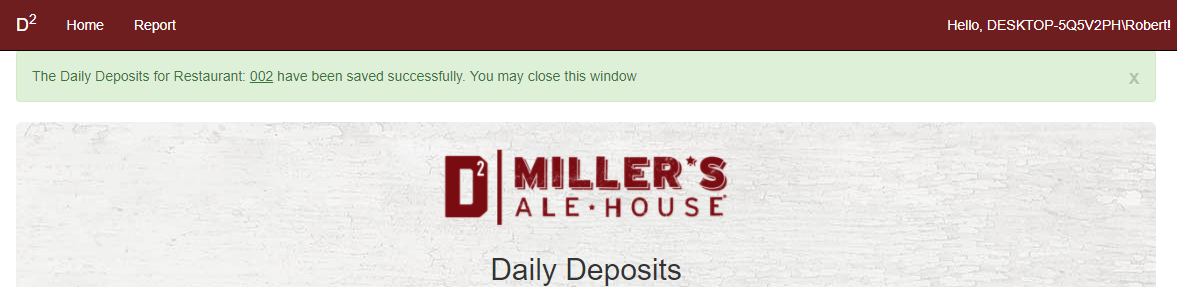
Upon clicking the save button at the bottom of the data entry area, this modal pop up appears:



This screen enables the user to double check that the data they are entering is correctly associated with the proper store, and if the user determines that this is inaccurate, it provides a phone number or an external link to the IT Support Request system. The save button on this modal dialogue is what will actually make database changes based on what the user has entered in the edit fields.

The “Daily Deposits” button on the navigation header will direct the user back to this page.

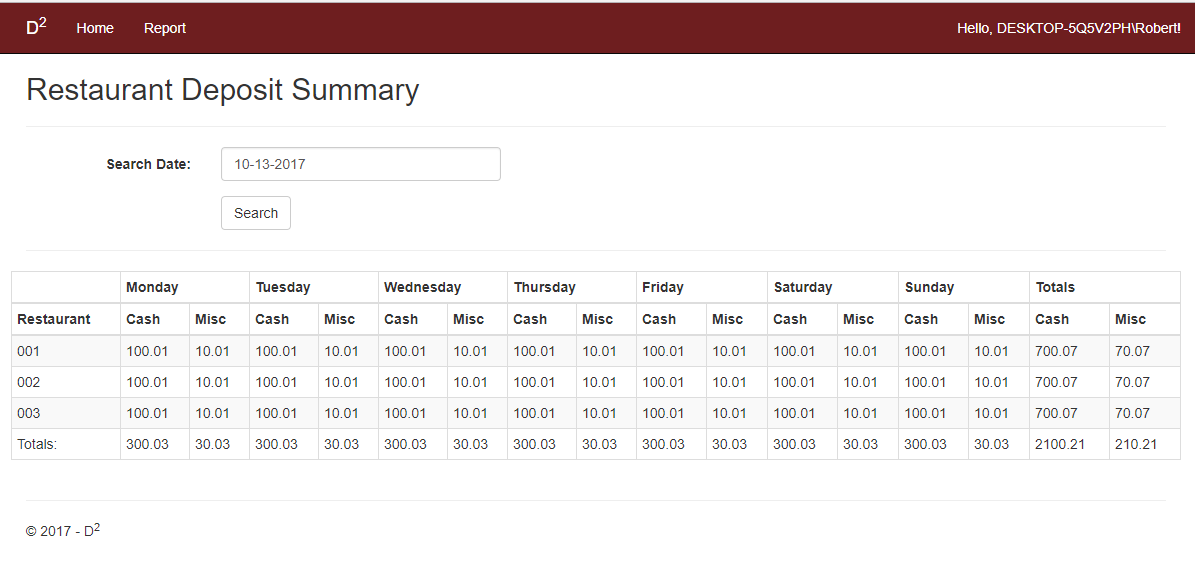
Subsequently, once the save button is clicked and the data was successfully entered, this verification message will appear at the top of the page below the header:



The user will then see the updated values reflected in the edit fields for the specified weekday below.

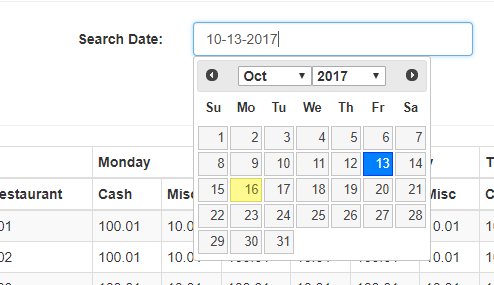
**Daily Deposit Report Page**

Upon clicking the “Report” button above the table on the Daily Deposits data entry page, this page is displayed:



This page shows a summary of all deposits made from all stores for a specified week, as well as weekly totals and grand totals for all stores broken down by Cash and Miscellaneous deposits.

Upon clicking the input field at the top labeled “Search Date” a date picking tool will appear:

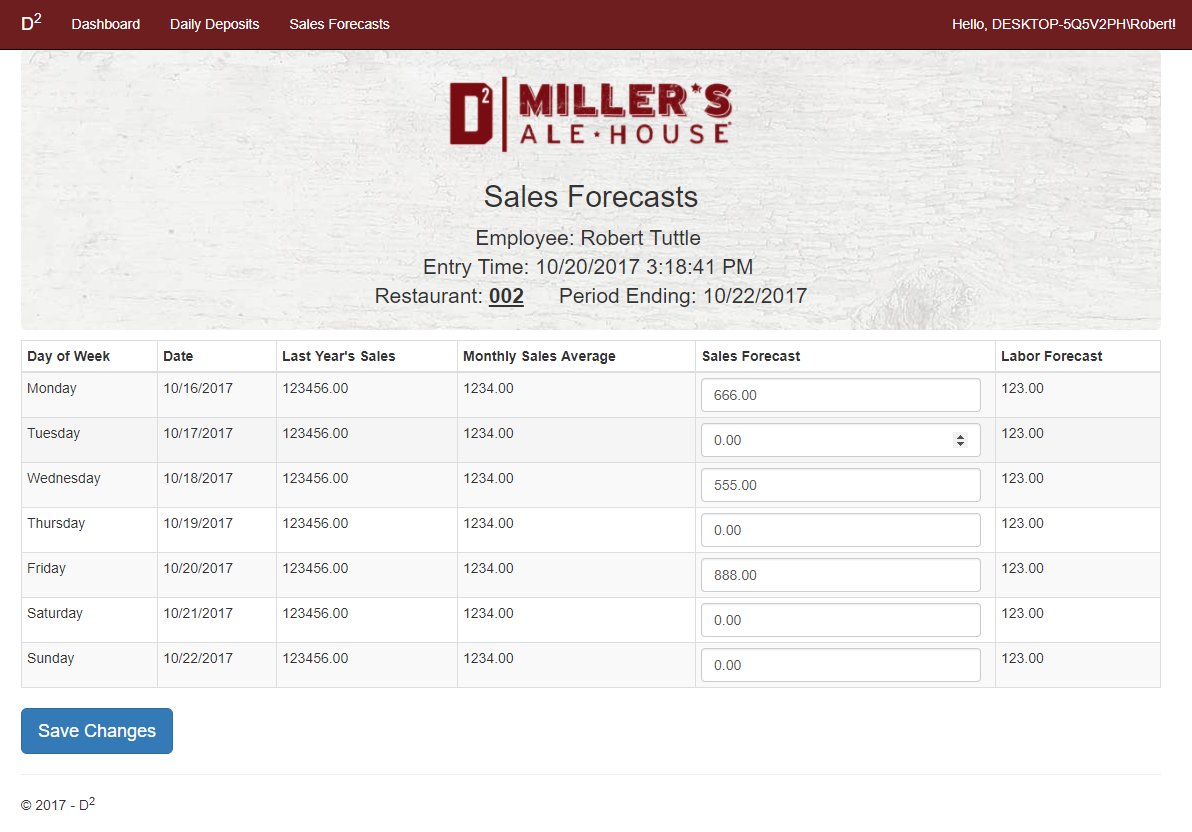


The blue highlighted date identifies the date that is selected in the editor, and the yellow highlighted date identifies the current date.

Upon searching with a selected date, the application will determine what week range (Monday through Sunday) the specified date is associated with, and then will display relevant deposit data associated with that week.

**Sales Forecast Entry Page**

Upon clicking the “Sales Forecast” link in the navigation header at the top of the application, this page will appear:



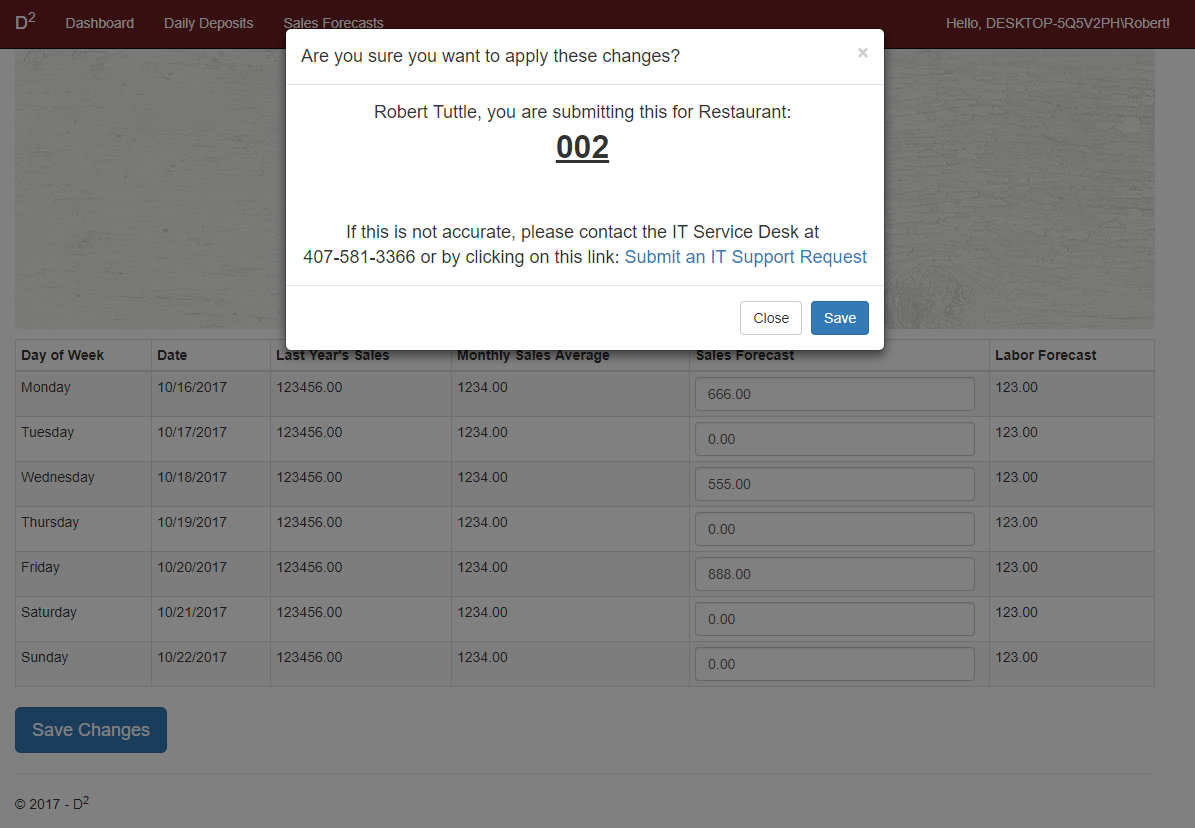
The following details are supplied at the top of the page:

* The DSquared Logo for Miller’s Ale House
* The name of the application: Sales Forecasts
* The current Employee that is logged in via Windows Authentication
* The time that the user initially accessed this page
* The unique Restaurant Number associated with the authenticated Employee
* The date in which the weekly sales forecast period ends

Like the Daily Deposit data entry page, this feature behaves very similar, but for sales predictions instead of actual recorded deposits.

Below the details at the top, there is a table that displays the day of the week, the date, last year’s sales, monthly sales averages, an area to enter sales forecasts, as well as a labor forecast. In the code implementation, the current week is determined (Monday through Sunday), and subsequently retrieves a set of pre existing records of sales forecasts for each day in the specified week range. Just like the Daily Deposits page, the first time a user saves data for the week, every record for each day is inserted into the database. After the first save, the user is just updating the pre existing records that were created on the initial save.

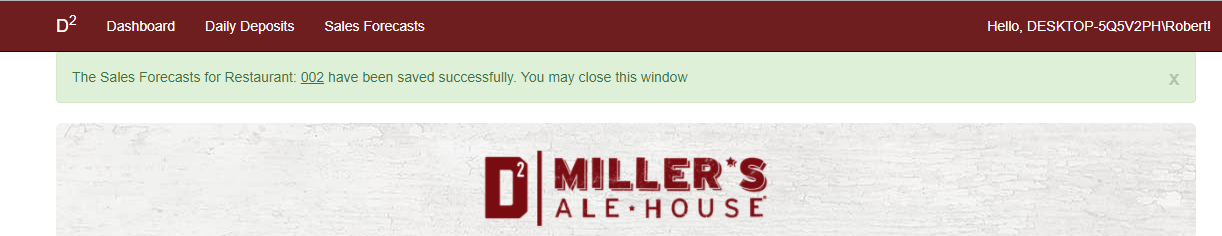
Upon clicking the save button at the bottom of the data entry area, this modal pop up appears:



This screen enables the user to double check that the data they are entering is correctly associated with the proper store, and if the user determines that this is inaccurate, it provides a phone number or an external link to the IT Support Request system. The save button on this modal dialogue is what will actually make database changes based on what the user has entered in the edit fields.

The “Sales Forecasts” button on the navigation header will direct the user back to this page.

Subsequently, once the save button is clicked and the data was successfully entered, this verification message will appear at the top of the page below the header:



The user will then see the updated values reflected in the edit fields for the specified weekday below.

**Daily Deposit Table Schema**

This is the create script that represents the schema for the data stored in the DailyDeposits Table:

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[DailyDeposits](

[Id] [int] IDENTITY(1,1) NOT NULL,

[BusinessDate] [datetime] NOT NULL,

[StoreNumber] [nvarchar](3) NULL,

[GlAccount] [int] NOT NULL,

[Amount] [decimal](18, 2) NOT NULL,

[UpdatedDate] [datetime] NULL,

[UpdatedBy] [nvarchar](50) NULL,

[CreatedDate] [datetime] NULL,

[CreatedBy] [nvarchar](50) NULL,

CONSTRAINT [PK\_dbo.DailyDeposits] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

**Sales Forecast Table Schema**

This is the create script that represents the schema for the data stored in the SalesForecasts Table:

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[SalesForecasts](

[Id] [int] IDENTITY(1,1) NOT NULL,

[BusinessDate] [datetime] NOT NULL,

[StoreNumber] [nvarchar](3) NULL,

[ActualPriorYear] [decimal](18, 2) NOT NULL,

[AvgPrior4Weeks] [decimal](18, 2) NOT NULL,

[LaborForecast] [decimal](18, 2) NOT NULL,

[UpdatedDate] [datetime] NULL,

[UpdatedBy] [nvarchar](50) NULL,

[CreatedDate] [datetime] NULL,

[CreatedBy] [nvarchar](50) NULL,

[ForecastAmount] [decimal](18, 2) NOT NULL,

CONSTRAINT [PK\_dbo.SalesForecasts] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[SalesForecasts] ADD DEFAULT ((0)) FOR [ForecastAmount]

GO