



Odin Health

Business Manager

A2 Computing Project 2015

William Shen

Table of Contents:

Definition – Nature of the Problem	16
Identification of the Problem	16
Investigation and Analysis	17
Field List	17
Accounts	17
Travel Expenses	17
Business Expenses	17
Revenue	18
Wages and PAYE (KiwiSaver deductions)	18
System Flowcharts	19
Travel Expenses	19
Business Expenses	20
Revenues	20
Wages	21
Discussing Platforms for the Solution to Run on	22
Exploring the Different Platforms in Depth	22
Final Decision	23
Requirements Specification	24
General Requirements	24
Required Specifications	24
Hardware and Software Requirements	26
The Objectives of the Solution	26
Design – Nature of the Solution	28
Objectives (Design Specification)	28
Design Philosophy	29
Database Design	30
Accounts Table	30
© William Shen	2

Travel Expense Application Table	31
Business Expense Application Table	32
Travel Expense Table	33
Business Expense Table	34
Revenues Table	35
Investment Table	35
Wages Table	36
Estimated Size of the Database	37
System Flowcharts	38
Adding or Updating an Account	38
Applying for Business or Travel Expenses.	38
Approving Applications for Expenses	39
Updating the Database Location	40
Backing up the Database	40
Contacting Technical Support	41
Adding Travel Expenses	41
Adding Business Expenses	42
Adding Revenues	42
Adding Investments	43
User Interface Design	44
Tool Strip	44
Login Form	45
Contact Support	46
Account Manager	47
User Settings	48
Main Menu	49
Travel Application – Overview	50
Travel Application – New	51
Travel Application – View and Edit	53

Business Expense Application – Overview	55
Business Expense Application – New	56
Business Expense Application – View and Edit	58
Travel Expenses – Overview	60
Travel Expenses – New, View and Edit	61
Business Expenses – Overview	63
Business Expenses – New	64
Business Expenses – View and Edit	65
Revenues – Overview	67
Revenues – New	68
Revenues – View and Edit	69
Investment	71
Wages – Overview	72
Wages – New	74
Wages – View and Edit	75
Report Design	76
Business Expenses	76
Travel Expenses	76
Wages	76
Testing Plan	77
About Form	78
Account Manager Form	78
Business Expenses – Overview Form	79
Business Expenses – New Form	80
Business Expenses - View and Edit Form	81
Business Expense Applications – Overview Form	82
Business Expense Application – New Form	83
Business Expense Application - View and Edit Form	84
Contact Support Form	85

Investment Form	85
Investment Form	86
Login Form	86
Main Menu Form	87
Revenues – Overview Form	88
Revenues – New Form	89
Revenues - View and Edit Form	90
Search Box Form	90
Settings Form	91
Settings Variables Form	91
Status Box Form	92
Travel Application – Overview Form	92
Travel Application – New Form	93
Travel Application - View and Edit Form	94
Travel Expense – Overview Form	95
Travel Expense Form	96
Wages - Overview Form	98
Wages – New Form	99
Wages - View and Edit Form	99
User Agreement	100
Intended Benefits	101
Automatic Emailing	101
Efficiency	101
Advanced Data Viewing and Filtering System	101
Automatic Calculation	102
Printing of Documents	102
Limits of the Scope of the Solution	103
Outdated Forms	103
Compatibility	103

Database Access	103
Storage	104
Upgrading the System	104
Development	105
Data Structures	105
Accounts Table	105
Business Expense Applications Table (BEA)	105
Business Expenses Table	105
Investments Table	105
Revenues Table	106
Travel Expense Applications Table (TEA)	106
Travel Expenses Table	106
Wages Table	106
Points of Difference	107
Program Form Layouts	108
Login Form	108
Contact Support	108
Account Manager	109
User Settings	109
Settings Variables	110
Main Menu	110
Travel Application – Overview	111
Travel Application – New	111
Travel Application – View and Edit	112
Business Expense Application – Overview	112
Business Expense Application – New	113
Business Expense Application – View and Edit	113
Travel Expenses – Overview	114
Travel Expenses – New, View and Edit	114

Business Expenses – Overview	115
Business Expenses – New	116
Business Expenses – View and Edit	116
Revenues – Overview	117
Revenues – New	117
Revenues – View and Edit	118
Investment	118
Investment – Edit	119
Wages – Overview	119
Wages – New	120
Wages – View and Edit	120
Search Box Form	120
Status Box Form	121
About Form	121
Points of Difference	121
Report Layouts (for Printing)	122
Business Expenses	122
Travel Expenses	122
Wages	122
Programming	123
Problems Encountered	123
Listed in Alphabetical Order	124
About.vb	124
AccountManager.vb	124
Business Expense - Overview.vb	130
Business Expense - New.vb	140
Business Expense - View and Edit.vb	144
Business Expense Application - Overview.vb	147
Business Expense Application - New.vb	151

Business Expense Application - View and Edit.vb	156
Contact.vb	160
Investment.vb	163
Investment - Edit.vb	166
ListView Sorting Class (ListViewSort.vb)	168
Login.vb	169
MainMenu.vb	173
Revenues - Overview.vb	176
Revenues - New.vb	181
Revenues - View and Edit.vb	183
Search Box Form (SearchBox.vb)	185
User Settings (Settings.vb)	190
Settings Variables (SettingsVariables.vb)	192
Status Box Form (StatusBox.vb)	193
Text Formatting Class (textFormatting.vb)	194
Travel Application - Overview.vb	195
Travel Application - New.vb	199
Travel Application - View and Edit.vb	204
Travel Expense - Overview.vb	209
Travel Expense – New, View and Edit (Travel Expense.vb)	216
Wages - Overview.vb	228
Wages - New.vb	237
Wages - View and Edit.vb	240
Testing	243
Testing Schedule	243
About.vb	243
AccountManager.vb	244
Business Expenses - Overview.vb	245
Business Expenses - New.vb	246

Business Expenses - View and Edit.vb	247
Business Expense Applications - Overview.vb	248
Business Expense Application - New.vb	249
Business Expense Application - View and Edit.vb	250
Contact.vb	251
Investment.vb	251
Investment - Edit.vb	252
Login.vb	252
MainMenu.vb	253
Revenues - Overview.vb	254
Revenues - New.vb	255
Revenues - View and Edit.vb	256
SearchBox.vb	256
Settings.vb	257
SettingsVariables.vb	257
StatusBox.vb	258
Travel Application - Overview.vb	258
Travel Application - New.vb	259
Travel Application - View and Edit.vb	260
Travel Expense - Overview.vb	261
Travel Expense.vb	262
Wages - Overview.vb	264
Wages - New.vb	265
Wages - View and Edit.vb	265
Testing Evidence	266
Fig 1.1	266
Fig 1.2	266
Fig 2.2	267
Fig 2.1	267

Fig 2.4	268
Fig 2.3	268
Fig 2.6	269
Fig 2.7	269
Fig 2.5	269
Fig 2.9	270
Fig 2.8	270
Fig 2.10	270
Fig 2.12	271
Fig 2.11	271
Fig 2.13	271
Fig 2.14	272
Fig 3.1	273
Fig 3.2	273
Fig 3.3	274
Fig 3.4	274
Fig 3.5	275
Fig 3.6	275
Fig 3.7	276
Fig 3.8	276
Fig 4.1	277
Fig 4.2	277
Fig 4.3	278
Fig 4.4	278
Fig 4.5	278
Fig 4.6	279
Fig 4.9	279
Fig 4.8	279
Fig 4.7	279

Fig 5.1	280
Fig 5.3	280
Fig 5.2	280
Fig 5.4	281
Fig 5.6	281
Fig 5.5	281
Fig 6.1	282
Fig 6.2	282
Fig 6.3	283
Fig 6.4	283
Fig 7.1	284
Fig 7.2	284
Fig 7.3	284
Fig 7.4	285
Fig 7.5	285
Fig 7.6	286
Fig 7.7	286
Fig 8.1	287
Fig 8.2	287
Fig 8.3	287
Fig 8.4	288
Fig 8.5	288
Fig 8.6	289
Fig 9.1	290
Fig 9.2	290
Fig 9.3	291
Fig 9.4	291
Fig 9.5	292
Fig 10.1	293

Fig 10.2	293
Fig 10.3	294
Fig 10.4	294
Fig 10.5	295
Fig 10.6	295
Fig 11.1	296
Fig 11.2	296
Fig 11.3	296
Fig 11.4	297
Fig 12.1	298
Fig 12.2	298
Fig 12.3	298
Fig 13.1 to 13.5	299
Fig 13.6 to 13.10	300
Fig 13.11	301
Fig 13.12	301
Fig 13.13	301
Fig 14.1	302
Fig 14.2	302
Fig 14.3	302
Fig 14.4	303
Fig 14.5	303
Fig 14.6	303
Fig 15.1	304
Fig 15.2	304
Fig 15.3	304
Fig 15.4	305
Fig 15.5	305
Fig 15.6	306

Fig 16.1	306
Fig 16.2	306
Fig 16.3	307
Fig 16.4	307
Fig 16.5	307
Fig 17.1	308
Fig 17.2	308
Fig 18.1	309
Fig 18.3	309
Fig 18.2	309
Fig 18.4	309
Fig 18.5	310
Fig 18.8 	310
Fig 18.6	310
Fig 18.7	310
Fig 19.1	311
Fig 20.1	311
Fig 19.3	311
Fig 19.2	311
Fig 21.1	312
Fig 21.3	312
Fig 21.2	312
Fig 21.4	313
Fig 22.2	313
Fig 22.1	313
Fig 22.3	314
Fig 22.4	314
Fig 22.5	315
Fig 22.6	315

Fig 22.7	316
Fig 23.1	316
Fig 23.2	316
Fig 23.3	317
Fig 23.4	317
Fig 23.5	317
Fig 23.6	318
Fig 24.1	318
Fig 24.2	319
Fig 24.3	319
Fig 24.4	320
Fig 24.5	320
Fig 25.1	321
Fig 25.2	321
Fig 25.5	322
Fig 25.6	322
Fig 25.8	323
Fig 25.9	323
Fig 25.11	324
Fig 25.12	324
Fig 25.15	325
Fig 26.1	325
Fig 26.2	326
Fig 26.3	326
Fig 26.5	327
Fig 26.6	327
Fig 27.1	328
Fig 27.3	329
Fig 27.4	329

Fig 27.5	330
Fig 28.3	331
Fig 28.5	332
Video Evidence	333
Installation	337
Implementation Plan	337
Discussing Types of Implementation	337
Chosen Implementation	337
Implementation Schedule	338
Training Details	339
Testing Details	339
Changeover Details	340
Acceptance Statement	340
Evaluation	341
Discussion of the degree of success in meeting the original objectives	341
Advantages	341
Limitations	341
Discussing Objectives	342
Evaluation of the client's and user's response to the system	345
General Requirements	345
Hardware and Software Requirements	346
Objectives of the Solution	347
Current System Flaws	348
User Comments	349
Client Comments	349
Final Developer Comments	350
User Acceptance Document	351

Definition – Nature of the Problem

Identification of the Problem

Currently, Odin Health uses a huge number of spreadsheets for tracking its business expenses, tracking expenses, wages, etc. This creates a very inefficient system in which a lot of time is wasted formatting and creating the spreadsheets as well as locating and collating them.

The current spreadsheet solution is very unorganised and requires the manual input of data across forms. Applications for reimbursements and such must also be manually sent via email or approved through phone calls. For a business, time is money, and it is vitally important that a system is as efficient as possible.

Thus as shown, the current system is inefficient and unorganised and a new system is required to keep all the data together and allow for automatic calculation and tasks.

Investigation and Analysis

Field List

Following the interviews with my client and analysis of the data which he has provided, I have compiled a list of fields used in the current system. Most of these fields are required and will be “transferred” to the new system along with some new additional fields. However, some of these fields may need to be removed as they are redundant or not required anymore.

These fields will be discussed in the second interview. The client and I will go over them and look for any fields that need to be removed, modified or added and add them to the requirements specification.

Accounts

- Username
- Password
- Full Name

Travel Expenses

- Tracking Number
- Employee Name (who the expense is for)
- Date
- Description
- Type (type of expense)
- Bill Client (if the client is to be invoiced for the expense)
- Payment Method (how the expense was paid initially)
- Amount
- Exchange Rate
- Amount in NZD

Business Expenses

- Date
- Company
- Cheque
- Total
- GST
- Debit

Revenue

- Company (that is purchasing the good or service)
- Purchase Order Number
- Invoice Number
- Invoice Date
- Item or Description (good or service)
- Quantity
- Unit Price (excluding GST and in USD)
- GST
- Total

Wages and PAYE (KiwiSaver deductions)

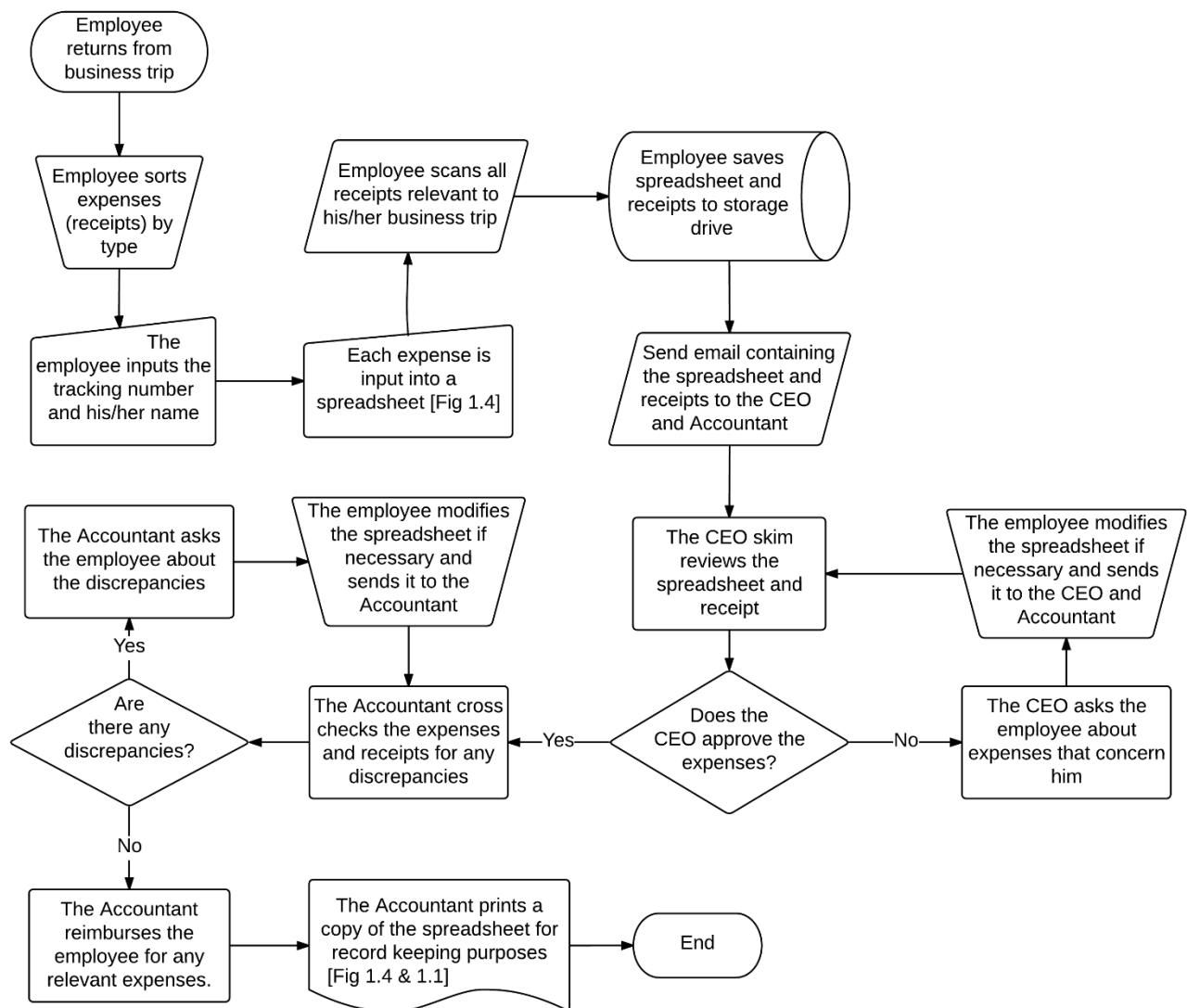
- Date
- Employee Name
- Total
- Salary
- PAYE
- Pay GST

System Flowcharts

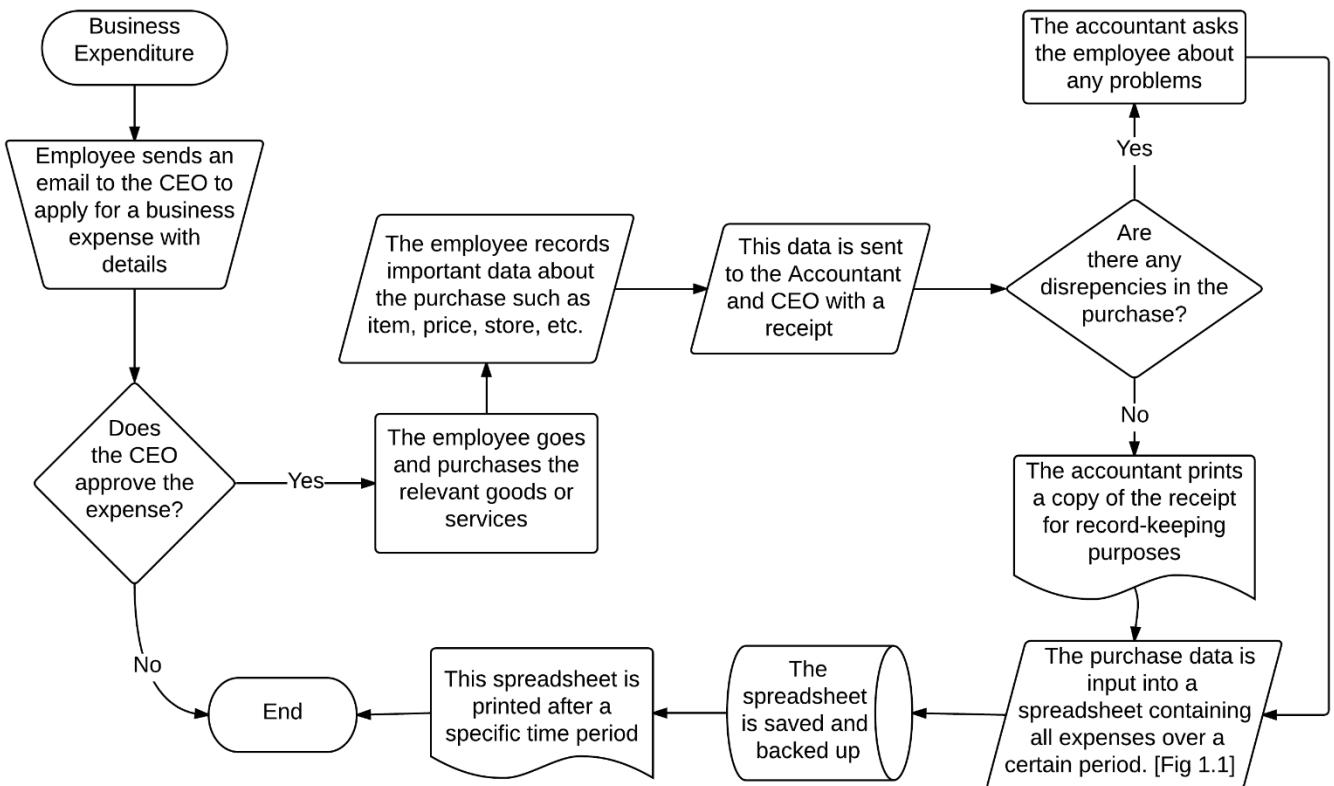
Following the interview and analysis of my client's data, I have created these system flowcharts describing all the steps the client goes through in his current business system for managing expenses and revenues.

These flowcharts will be an integral part of the next interview in which I will confirm the process of the current system with my client and further confirm and discuss what parts he wants to update, change or add.

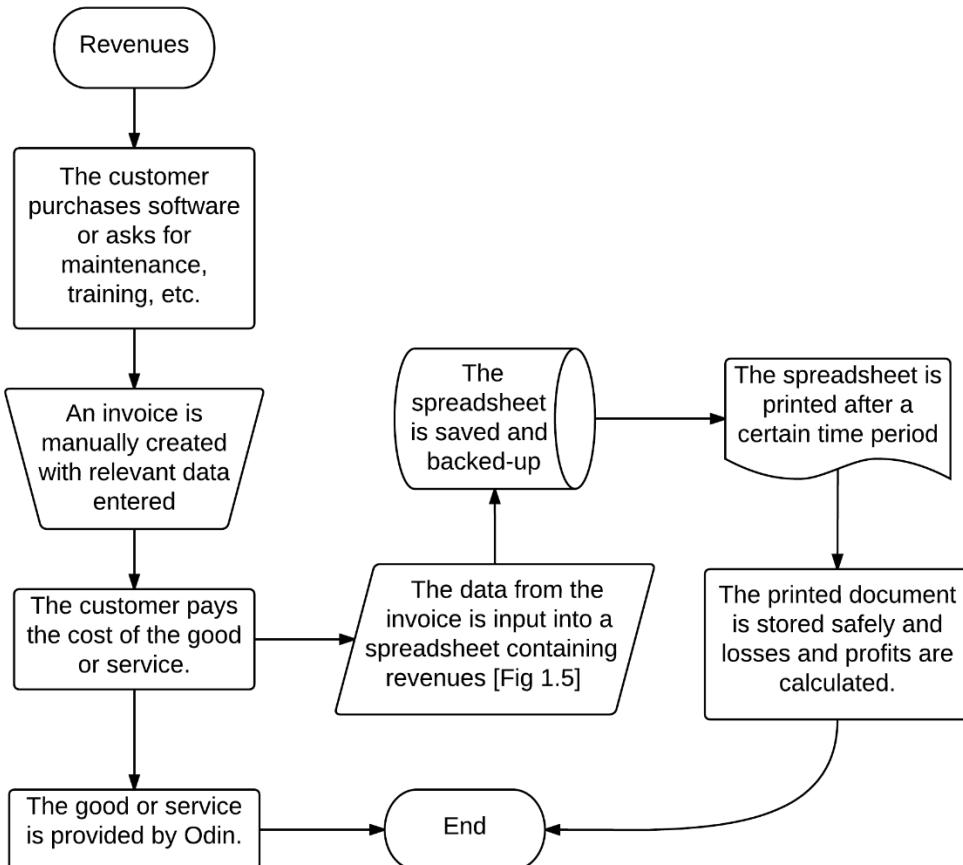
Travel Expenses



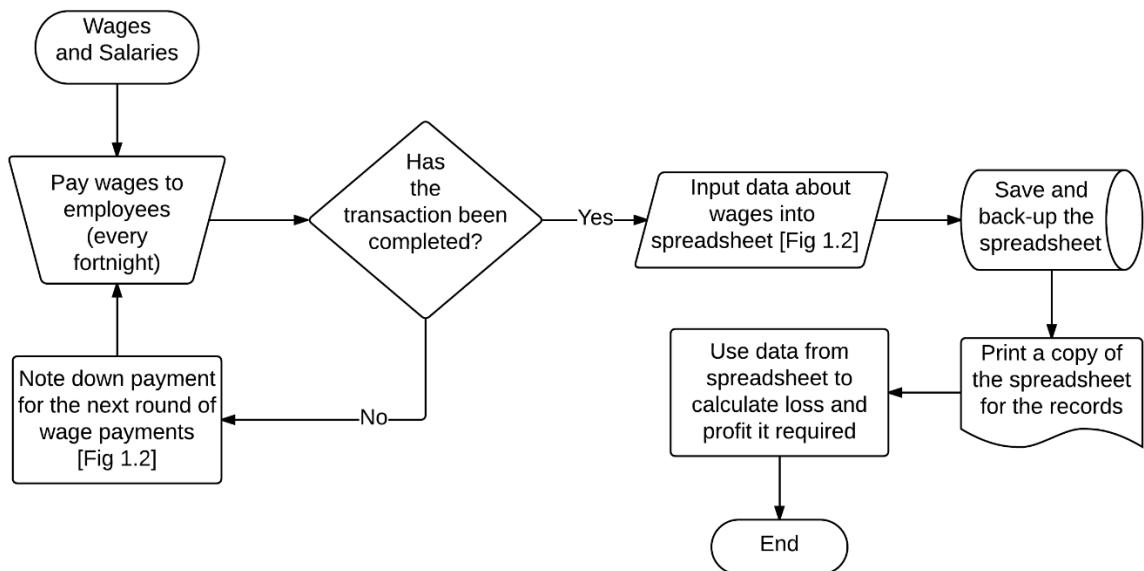
Business Expenses



Revenues



Wages



Discussing Platforms for the Solution to Run on

It is vitally important that the final solution can be used by all the employees of Odin Health. If an incorrect platform is chosen for the solution then the product will be inefficient and a waste as not everyone will use it. All the different platforms have their own benefits and drawbacks.

Thus, it is important to consider all options with the client and decide on the best one so that all employees may use the solution and to fulfil the requirements of the client. Here I have a brief interview with the client to find out the kind of hardware the company currently uses.

Exploring the Different Platforms in Depth

iOS or Android Application

Advantages	Disadvantages
Very portable and mobile.	Not everyone uses the same operating system.
	Limited in functionality.
	More difficult to push updates.

Therefore we rule the option of an iOS or Android application out.

Website

Advantages	Disadvantages
Very portable	Relies on the internet to work.
Anyone with a login can use the system.	Security issues unless encryption is used.
	The system only needs to be used in the office.

Therefore we rule the option of a Website out as there is no need for one for the client as they only need to access the solution from the office.

Microsoft Windows Based Program

Advantages	Disadvantages
<p>Very portable</p> <p>All employees have Windows on their computers.</p> <p>Provides many different functions such as printing</p> <p>Allows data input from many different data sources</p>	<p>High resource requirements</p> <p>Cannot be used on a smartphone or Mac (Boot camp required)</p> <p>Difficult to edit code and push revised builds of the solution</p>

A Windows Based Program contains more comparative advantages compared to the other platforms. Therefore, the client and I agreed on such a program. This would be accessible to all employees and would require both the .NET Framework and Access OLE DB Drivers.

Final Decision

Both the client and I thought that the most efficient and feasible option would be to go with a Microsoft Windows Based Program. This would allow all the employees to access the solution which is vitally important.

I explained to the client that the solution would be coded in Visual Basic .NET with Microsoft Access as the database management software to manage and edit the database if required.

This kind of platform would be the best to fit the client's requirements and provide many output formats. I would be able to hide unnecessary options from the user and make most of the functions work "behind the scenes".

Requirements Specification

The new proposed system must be able to fulfil the needs and requirements of the clients. It is important that the client and I agree on the same requirements otherwise the new system would be inefficient and perhaps a waste of time. The requirements specification is based upon the decision to use Visual Basic .NET and Microsoft Access 2013.

General Requirements

1. The system must be efficient and quick to use.
2. Data should be stored and organised quickly and practically.
3. The user interface must be unambiguous and easy to navigate and use.
4. Reports must be able to be printed.
5. Back-ups of the database must be easy to perform.

Required Specifications

The new system must be able to store, organise and sort all of the following fields. Additionally, the user must be able to easily edit fields and add new records. The system must be able to prepare reports and sort them and allow the user to print these. Furthermore, the system must allow for the searching of records by different field parameters and display results.

The required 'fields' (data) the program will store are:

Text in 'Blue' are revisions and additions which were discussed in the 2nd interview

Accounts

- Username
- Password
- Full Name

Travel Expense Application

- Tracking Number (automatically assigned)
- Employee Name (who is applying)
- Duration of Trip (in days)
- Reason for Travel
- Estimated Cost of Accommodation
- Domestic Travel Costs
- International Travel Costs
- Total Estimated Cost
- Approved (approved by CEO)

Business Expense Application

- Tracking Number (automatically assigned)
- Employee Name (who is applying)
- Description of Item
- Quantity
- Unit Price
- Total Price
- Organisation of Purchase (who the good is purchased from)
- Reason for Expense

Travel Expenses

- Tracking Number (previous automatically assigned)
- Employee Name (who the expense is for)
- Date **Incurred**
- Description
- Type (type of expense)
- Bill Client (if the client is to be invoiced for the expense)
- **Reimburse** (if employee is to be reimbursed for the expense)
- Payment Method (how the expense was paid initially)
- Amount (**in foreign currency or NZD**)
- Exchange Rate (**manually entered**)
- Amount in NZD

Business Expenses

- Tracking Number (previously automatically assigned)
- Employee Name (who the expense is for)
- Date **Incurred**
- Description of Item
- Quantity
- Organisation of Purchase (who the good is purchased from)
- **Reimburse** (if employee is to be reimbursed for the expense)
- Payment Method (how the expense was paid initially)
- Total (**including GST in NZD**)

Revenue

- Company (that is purchasing the good or service)
- Purchase Order Number
- Invoice Number

- Invoice Date
- Description (good or service)
- Quantity
- Unit Price (excluding GST and in NZD)
- GST Paid
- Total (excluding GST and in NZD)

Wages and PAYE (KiwiSaver deductions)

- Period for Payment
- Date Paid
- Employee Name
- Salary
- PAYE (KiwiSaver deductions)

Hardware and Software Requirements

The new system must be able to be used on any computer running Windows 7, 8 or 8.1. A computer with a mouse, keyboard and printer would be necessary to utilise the system to its full potential.

The main database must be stored on the network so all employees can access it. Furthermore, an option to store the database locally (i.e. save it to a different directory), so that data may be accessed when the network cannot be accessed, is essential.

The new system must also be able to print reports to any printer that the client wishes. The required information must be able to be easily viewed and sorted for this printing.

The new system will be built on Microsoft Visual Basic .NET while using Microsoft Office Access 2013 to manage the database. This would require all Users computers to have the .NET Framework installed and relevant Microsoft Access Drivers or Microsoft Access 2013 to enable OLE DB (Object Linking and Embedding, Database) database connections.

The Objectives of the Solution

The solution must be able to:

1. Produce printable reports with the relevant data.
2. Add, delete and update Usernames, Passwords and Full Names (Accounts).

3. Provide a security barrier to prevent unauthorised access to the system.
4. Limit access to some sections of the system to selected Users.
5. Change the location of the database and back-up the database.
6. Allow users to send direct emails to the developer for Technical Support.
7. Allow users to apply for Travel or Business Expenses by filling in the relevant fields and sending a direct email to the CEO asking for approval.
8. Allow the CEO to approve user applications through the new system and send relevant emails if the application is approved,
9. Add, delete and search these applications for expenses and automatically assign tracking numbers.
10. Allow users to add, modify and delete travel and business expenses.
11. Add, edit and delete revenues and wages and all their relevant fields.
12. Display all this data in a table-based view.
13. Allow the entry, addition and deletion of data through a form-based interface.
14. Sort data based on specific fields.
15. Query data based on specific fields.

Design – Nature of the Solution

Objectives (Design Specification)

These are the objectives of the solution which were previously discussed and agreed on in the Requirements Specifications.

The solution must be able to:

1. Produce printable reports with the relevant data.
2. Add, delete and update Usernames, Passwords and Full Names (Accounts).
3. Provide a security barrier to prevent unauthorised access to the system.
4. Limit access to some sections of the system to selected Users.
5. Change the location of the database and back-up the database.
6. Allow users to send direct emails to the developer for Technical Support.
7. Allow users to apply for Travel or Business Expenses by filling in the relevant fields and sending a direct email to the CEO asking for approval.
8. Allow the CEO to approve user applications through the new system and send relevant emails if the application is approved.
9. Add, delete and search these applications for expenses and automatically assign tracking numbers.
10. Allow users to add, modify and delete travel and business expenses.
11. Add, edit and delete revenues and wages and all their relevant fields.
12. Display all this data in a table-based view.
13. Allow the entry, addition and deletion of data through a form-based interface.
14. Sort data based on specific fields.
15. Query data based on specific fields.

Design Philosophy

After some discussions with my client, I have drafted up the design philosophies of the system which will help improve user friendliness and efficiency.

1. The design must utilise the corporate colour scheme (dark blue and light blue)
2. The design must be intuitive to use, with clear consistent form structures so that the user can easily and quickly navigate around the program.
3. The design must make tasks easy to complete and communicate with the user in the user's own language.
4. The design must make all include all options required for a given task visible without distracting the user with redundant controls or information.
5. The design of the new system ensures that there is no information overload.
6. The design should keep the users of the system informed of its actions, changes in status, errors, etc. through an unambiguous language familiar to the user.
7. The design should maintain consistency and thus allowing users to reduce the need to search for controls and rethink options.
8. The design must utilise a font that is aesthetically pleasing yet easy to read.

The User Interface Design found later in the design section is based upon this design philosophy. As you will probably note, the design is consistent across multiple forms as shown by the location of titles, controls, etc.

The font chosen for the solution is Segoe UI with varying strengths of boldness (e.g. semilight, semibold). Here is an example of the Segoe UI typeface.

The Quick Brown Fox Jumps Over The Lazy Dog.

The Quick Brown Fox Jumps Over The Lazy Dog.

The Quick Brown Fox Jumps Over The Lazy Dog.

Database Design

For the design of the tables, I have chosen not to use foreign keys. Although foreign keys are useful in allowing 'not valid' data enter a database system, foreign key constraints mean that at times the system will not allow the deletion of data.

For example, if an employee named 'Tom Jenkins' makes a travel expense application and then leaves the company, then his information must be deleted from the Accounts Table. A foreign key would not allow this to happen as there is still data in the application table with the foreign key name 'Tom Jenkins'. Thus as shown, foreign keys are not suitable for this type of system and instead other primary keys will be used.

Accounts Table

This is the Accounts Table which holds all the login information for the employees. The primary key for this table is the Username field which would be unique for every single employee.

There is an extra field on top of the fields required called 'Admin' which will be used to determine whether a user has high-access levels to access certain parts of the program which 'normal' users cannot.

Field Name	Data Type	Estimated Size	Validation
Username	String	40	Uniqueness Check
Password	String	50	Presence Check
FullName	String	80	Character Check
Admin	Boolean	1	

Estimated size for one record = 171 bytes

Travel Expense Application Table

This is the table for applying for a travel expense. The tracking number is automatically assigned by the system. The fields correspond with those discussed between the client and me previously.

There is an addition of the field 'Email' and 'Destinations', 'Email' is so that there is an email address to send an approval email back to while the 'Destinations' field allows the user to indicate where he or she is going to go for this business trip.

Field Name	Data Type	Estimated Size	Validation
TrackingNumber	String	10	Format Check
FullName	String	80	Limited Choices
Email	String	255	Format Check
Destinations	String	80	Presence Check
Duration	Integer	4	Range Check
Reason	String	255	Presence Check
AccomodationCosts	Currency	10	Numerical Only
DomesticTravelCosts	Currency	10	Numerical Only
InternationalTravelCosts	Currency	10	Numerical Only
TotalCost	Currency	10	Numerical Only
Approved	Boolean	1	

Estimated size for one record = 725 bytes

Business Expense Application Table

This is the table which will hold the Business Expense Applications. The tracking number is automatically generated while all the other fields have been taken from the fields list agreed upon in the requirements specification.

There have been minor alterations in the naming of these fields and the addition of the field 'Email' which will be used as the receiving address when the CEO approves an application.

Field Name	Data Type	Estimated Size	Validation
TrackingNumber	String	10	Format Check
FullName	String	80	Limited Choices
Email	String	255	Format Check
ItemDescription	String	100	Presence Check
Quantity	Integer	5	Range Check
UnitPrice	Currency	10	Numerical Only
TotalPrice	Currency	10	Numerical Only
Vendor	String	50	Numerical Only
Reason	String	255	Numerical Only
Approved	Boolean	1	

Estimated size for one record = 776 bytes

Travel Expense Table

This is the table which will be used to record the travel expenses which the employee will enter after he or she has returned from a business trip. The primary key on this field will be an extra auto number field as there will be many entries for the TrackingNumber on this table.

All the fields in this form are taken from the requirements specification with minor name modifications. There is however an addition which is the field 'ForeignCurrency', this field is used to store the different abbreviations of currencies (e.g. USD, RMB, AUD, etc.)

Field Name	Data Type	Estimated Size	Validation
TrackingNumber	String	10	Format Check
FullName	String	80	Limited Choices
DateIncurred	Date	8	Format Check
ExpenseType	String	20	Limited Choices
Description	String	100	Presence Check
BillClient	Boolean	1	
PaymentMethod	String	30	Limited Choices
ForeignAmount	Currency	10	Numerical Only
ForeignCurrency	String	3	Limited Choices
ExchangeRate	Integer	10	Numerical Only
NZDAmount	Currency	10	Numerical Only
Reimburse	Boolean	1	

Estimated size for one record = 283 bytes

Business Expense Table

This is the Business Expense table which will be used to record all the business expenses incurred by the company. The primary key for this table would be the TrackingNumber which would be unique for every business expense which has been previously assigned.

There have been a few modifications to the fields from the requirements specification. The Quantity, UnitPrice and TotalPrice have been added to reflect the data available in the expense application form. Furthermore, some of the names of the fields have been modified to reflect a more efficient database system.

Field Name	Data Type	Estimated Size	Validation
TrackingNumber	String	10	Format Check
FullName	String	80	Limited Choices
DateIncurred	Date	8	Format Check
Description	String	100	Presence Check
Quantity	Integer	5	Range Check
UnitPrice	Currency	10	Numerical Only
TotalPrice	Currency	10	Numerical Only
Reimburse	Boolean	1	
PaymentMethod	String	30	Numerical Only
Vendor	String	50	Presence Check

Estimated size for one record = 304 bytes

Revenues Table

The revenues table will be used to store data on revenues collected from sales invoices. There have been minor modifications to the field names but otherwise, the fields are all the same.

The primary key for this table would be a special 'ID' which would be unique for every single sales invoice. This will be used instead of the PONumber as there may be multiple entries for one purchase order as there could be multiple goods or services the customer bought.

Field Name	Data Type	Estimated Size	Validation
Customer	String	50	Presence Check
PONumber	String	15	Uniqueness Check
InvoiceNumber	String	15	Presence Check
InvoiceDate	Date	8	Format Check
Description	String	100	Presence Check
Quantity	Integer	5	Range Check
UnitPrice	Currency	10	Numerical Only
TotalPrice	Currency	10	Numerical Only

Estimated size for one record = 213 bytes

Investment Table

This is a completely new table which will be used to store data about investment from other companies. The primary key for this field would be an auto number 'ID' field.

Field Name	Data Type	Estimated Size	Validation
Investor	String	50	Presence Check
Date	Date	8	Format Check
InjectionAmount	Currency	10	Numerical Only

Estimated size for one record = 68 bytes

Wages Table

This is the table which will store the data containing the wage payments. There have been some alterations to the fields from the requirements specification as the client later expressed some new fields he wanted to add in.

Firstly, the Period for Payment has been split into a start date and end date for ease of use and calculation. There have been minor modification to some field names and additions of 'GrossPay', 'KiwiSaver' and 'NetPay'. The NetPay = GrossPay – PAYE – KiwiSaver.

Field Name	Data Type	Estimated Size	Validation
StartDate	Date	8	Format Check
EndDate	Date	8	Format Check
FullName	String	80	Character Check
DatePaid	Date	8	Format Check
GrossPay	Currency	10	Numerical Only
PAYE	Currency	10	Numerical Only
KiwiSaver	Currency	10	Numerical Only
NetPay	Currency	10	Numerical Only

Estimated size for one record = 144 bytes

Estimated Size of the Database

In order to calculate the estimated size of the database in total, we must consider how many records there will be in each of the appropriate database tables. Note: all these calculations are based on very upper limits.

For the 'Accounts' table, there is currently 5 employees but this should gradually increase over time to approximately 20 to 30 employees. Thus we can estimate a size of $171 \times 30 = 5130$ Bytes.

In regards to the Travel Expense Application 'table', we can estimate that there will be around 5 business trips every month and thus 60 business trips a year. Thus we can calculate $725 \times 60 = 43500$ Bytes.

For the Business Expense Applications Table, we can estimate that there will be approximately 15 to 20 business expense purchases a month (240 applications a year); hence we can calculate an approximate size of $776 \times 240 = 186240$ Bytes.

In regards to the recording of Travel Expenses, given that there will be a maximum of 60 business trips a year, and each business trip will contain approximately 20 to 30 different items, we can estimate a size of approximately $283 \times 60 \times 30 = 509400$ Bytes.

For the Business Expense Table, given that all the applications are approved, there will be 240 applications a year. Each business expense would only contain one record, thus we can calculate a size of approximately $304 \times 240 = 72960$ Bytes.

For the revenues table, we can estimate a maximum of around 100 sales a year. This could give an estimated size of $213 \times 100 = 21300$ Bytes.

In regards to the Investment Table, we can estimate approximately 10 to 20 rounds of investment a year. Thus, we can calculate an estimated size of approximately $68 \times 20 = 1360$ Bytes.

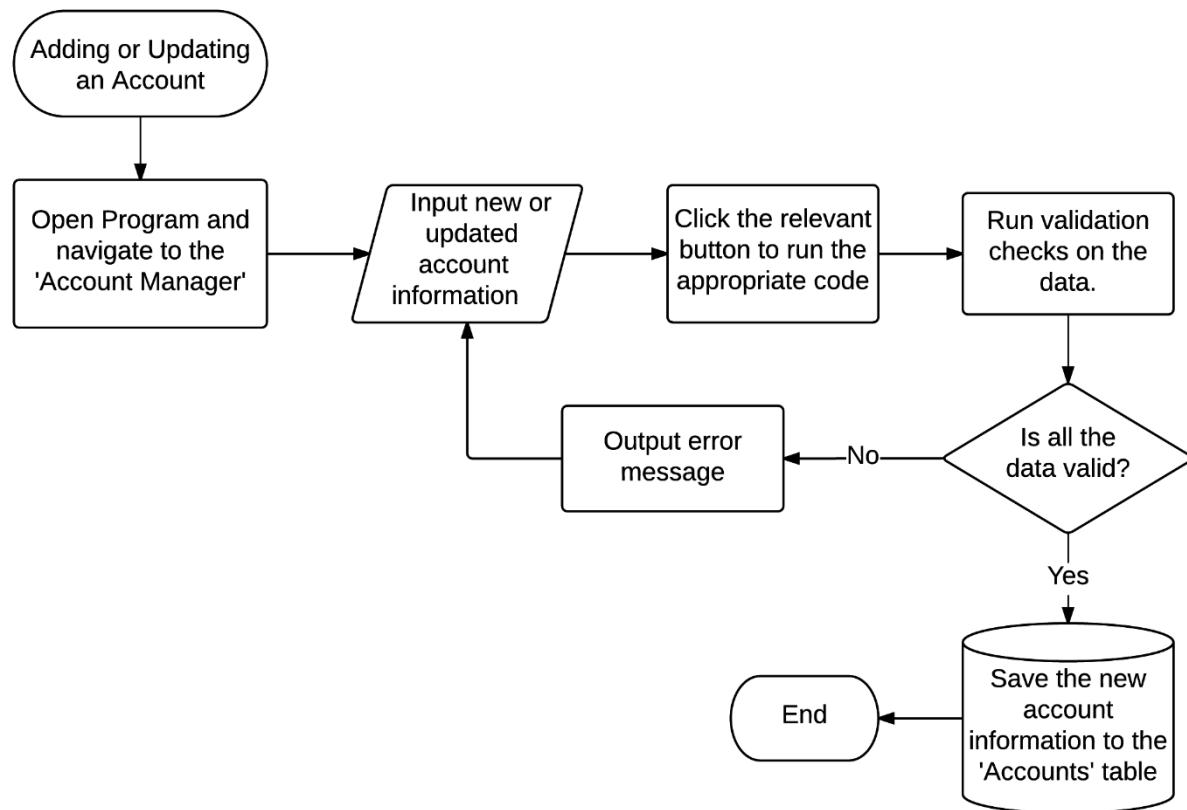
Since wages are usually paid every fortnight, there will be approximately 26 payments each employee. Given a company workforce size of 20 employees, we can calculate the Wages Table as approximately $144 \times 26 \times 20 = 74880$ Bytes.

Total Estimate Size of Records in the Database = $5130 + 43500 + 186240 + 509400 + 72960 + 21300 + 1360 + 74880 = 914770$ Bytes = 0.915 Megabytes.

System Flowcharts

Adding or Updating an Account

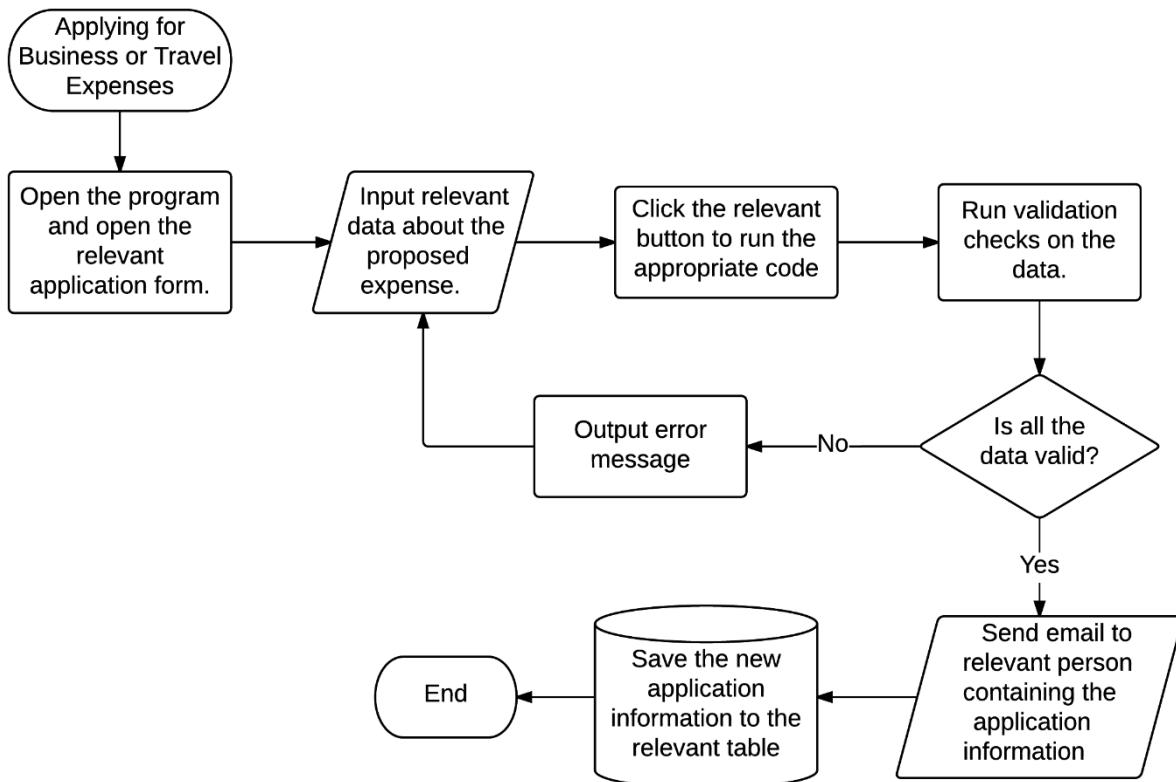
Adding or updating a login account is an important task to perform when either a new employee enters a company, or someone's login details becomes vulnerable. The deletion of an account is relatively similar except the user chooses the Username from a drop-down list and then presses the button 'Delete'.



Applying for Business or Travel Expenses.

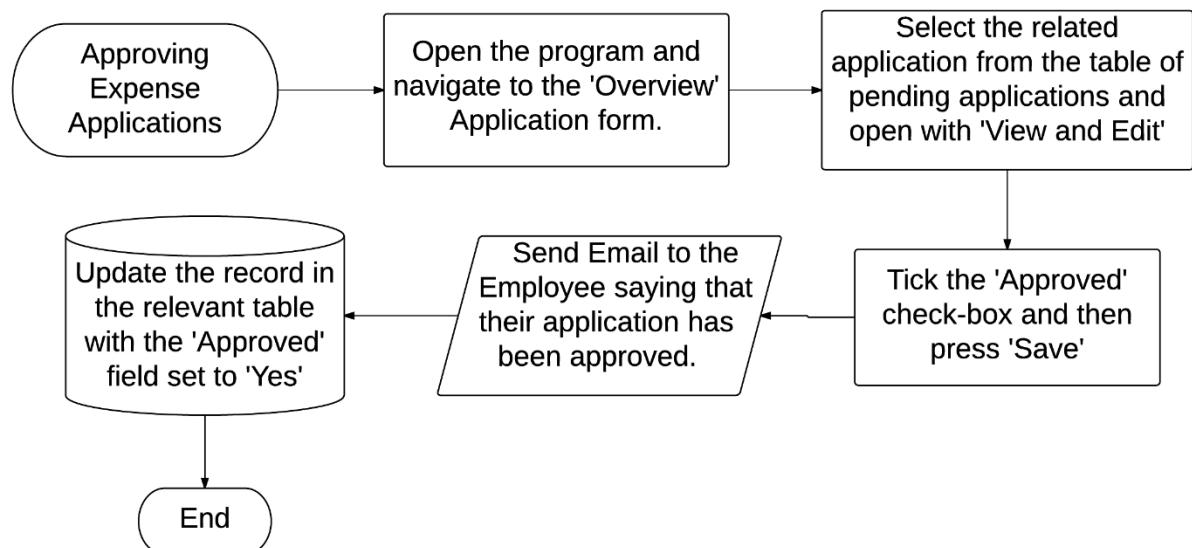
It is important that employees are able to directly and easily apply for either travel or business expenses. The proposed system would increase efficiency as an application would be directly emailed to the relevant person (e.g. CEO) when it is submitted.

The system flowchart may be found on the next page.



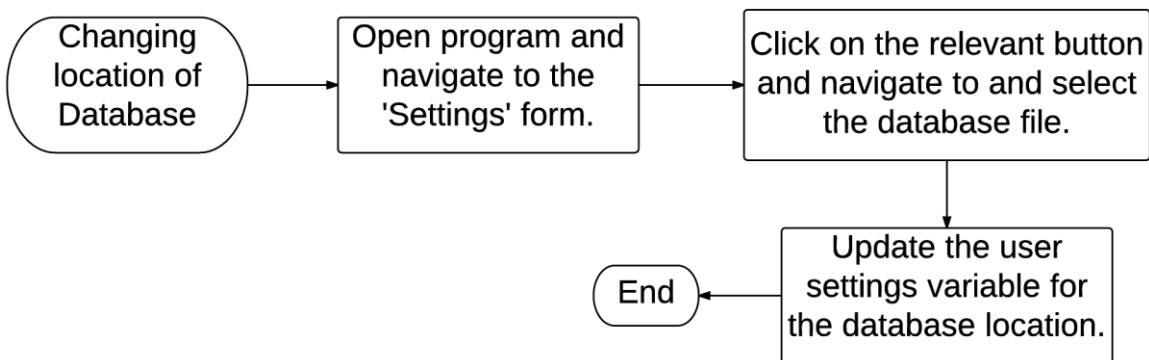
Approving Applications for Expenses

The new system must provide a way for the CEO or relevant person to approve the expense applications of the employees. As shown by the system flowchart, this process is very intuitive and helpful as it allows the sending of an email to the employee indicating that their application has been approved.



Updating the Database Location

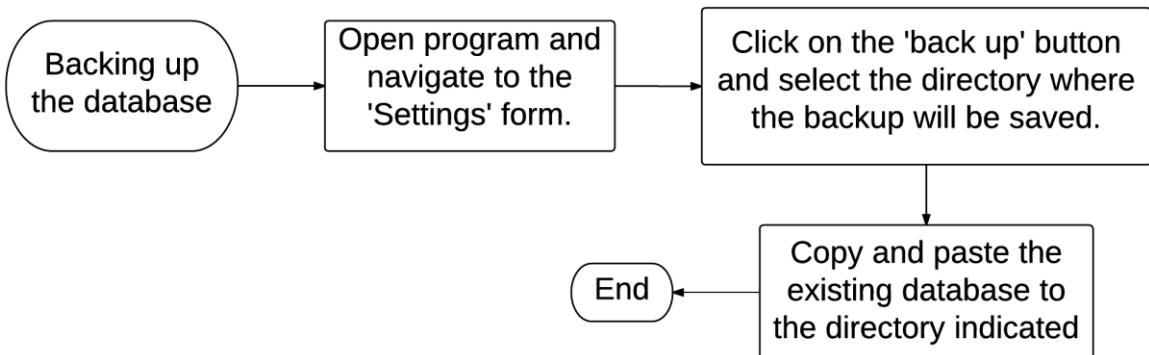
This is the system flowchart for updating or changing the location of the database. The program will utilise an 'Open File Dialogue' which would be easy and efficient to use for any user.



Backing up the Database

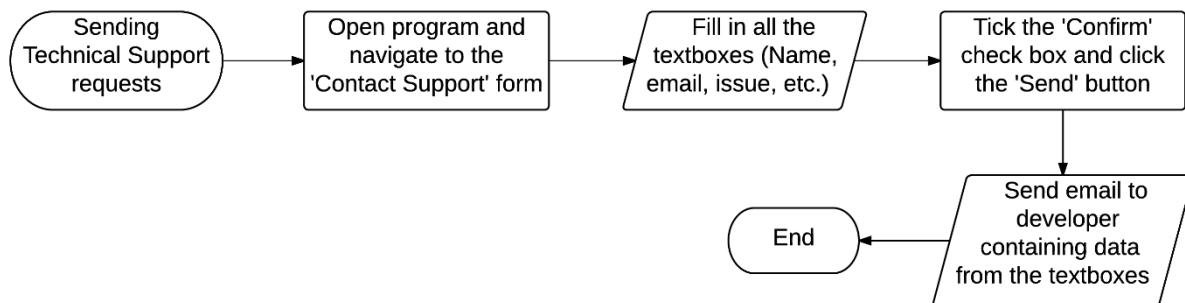
Backing up the database is important for cases where the main database file becomes corrupted and thus unusable. It is advised that the client makes a backup at least every week.

The way the proposed system will back up the database is by copying the current database and pasting it to a location which the user has indicated.



Contacting Technical Support

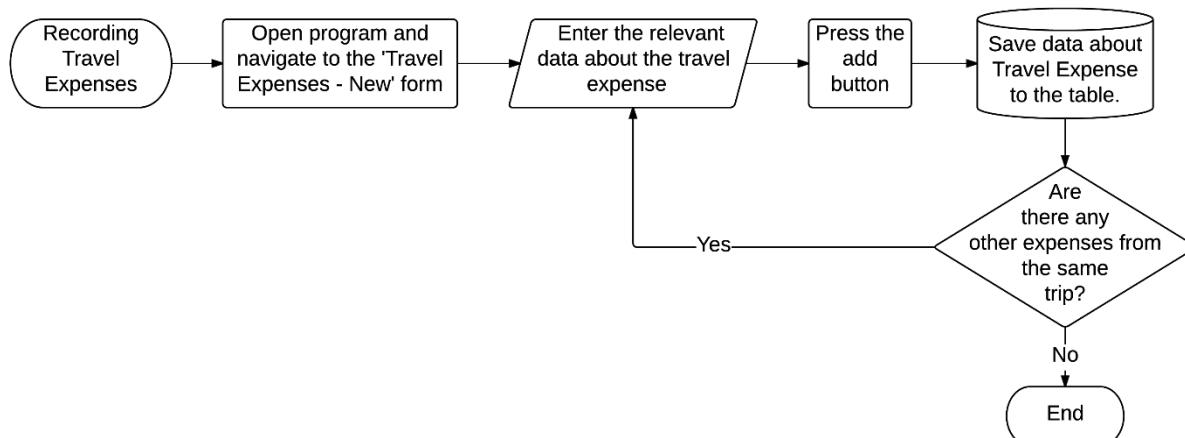
It is important for the client to relay any problems they are experiencing to the developer who can then go on to fix these glitches or bugs and then ‘push’ a new version of the solution to the client. Furthermore, general enquiries about the program can be sent to the developer.



Adding Travel Expenses

All travel expenses are costs incurred to Odin Health and must be carefully recorded by all employees. Furthermore, these travel expenses contain information that is required in order to reimburse the employees for their purchases.

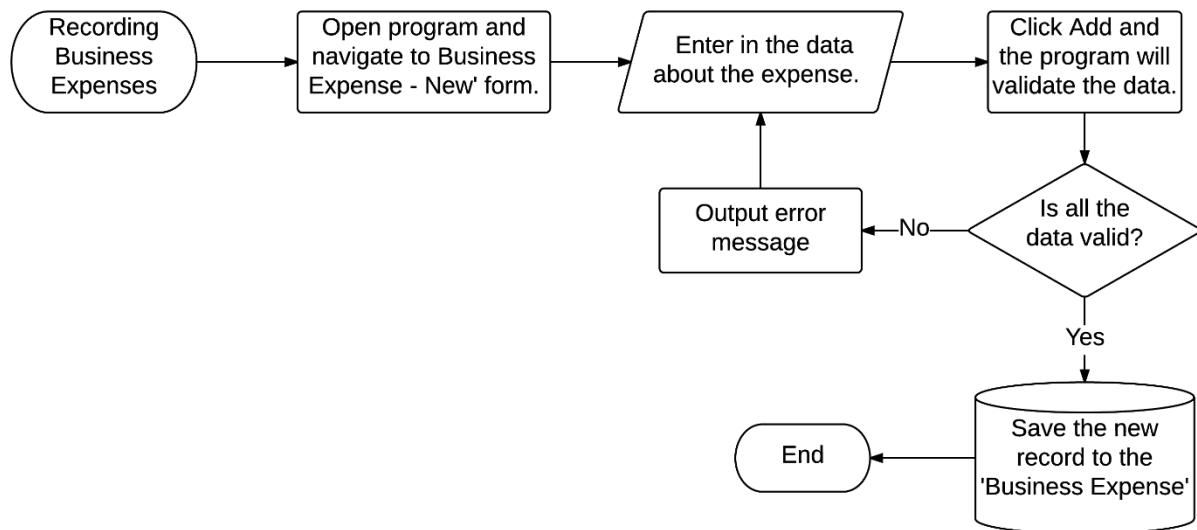
The program will allow the user to add multiple entries via the same tracking number by using a unique ID for the primary key instead of the tracking number itself. The program will additionally allow the user to print out a copy of their expenses once they have completed adding them for a single tracking number.



Adding Business Expenses

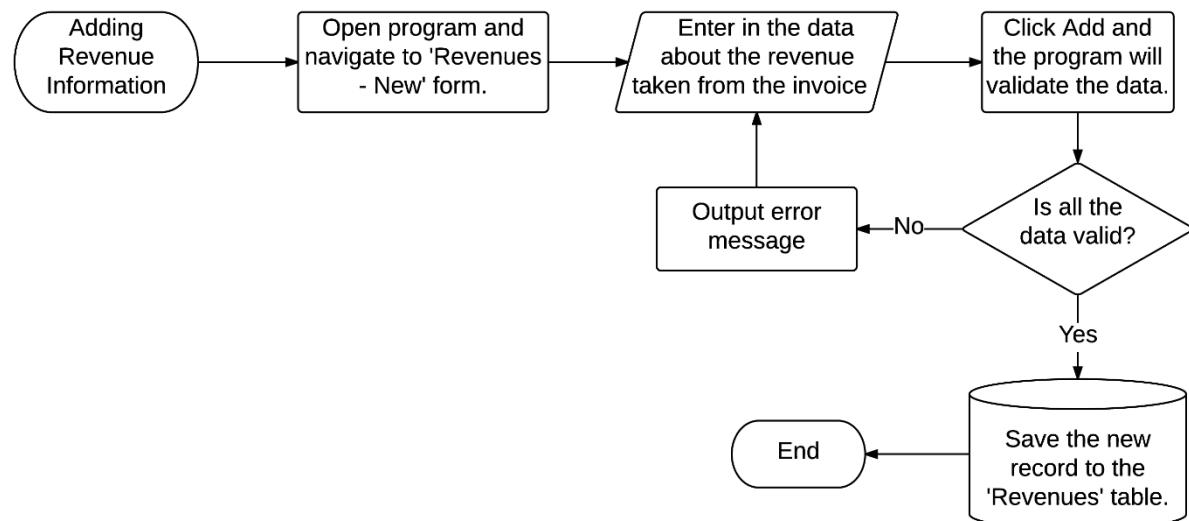
The business expenses incurred by the company must be recorded for legal purposes and so that employees may be reimbursed. The unique identifier for these business expenses is the tracking number.

This proposed system would greatly increase the efficiency of recording business expenses as new spreadsheets do not need to be created.



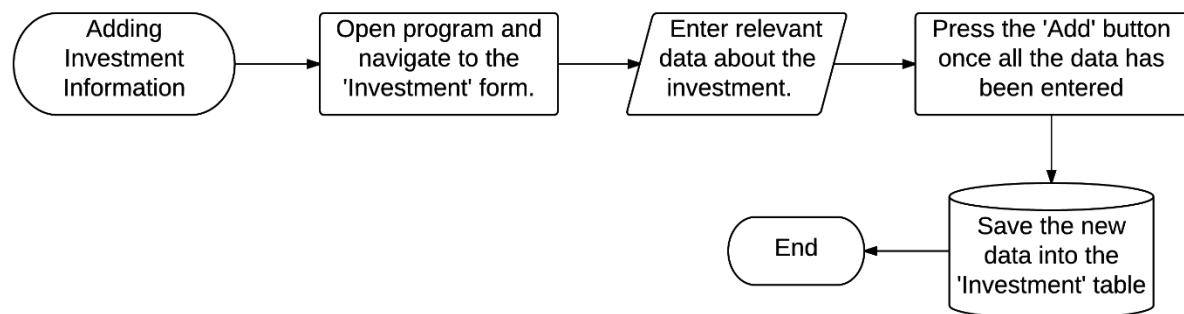
Adding Revenues

Revenues are an integral part of a business which allows them to continue with their operations, expand and research on new technologies. These must be carefully recorded to allow indication of overspending in a company. The data on this form would be directly taken from the 'Sales Invoice'.



Adding Investments

Investments indicate that the general consensus of a company is good and that it is in a good business health and position. It is vitally important that these are recorded as with revenues so that Odin can keep track of their costs and profits and check if they are running a surplus or a deficit.



User Interface Design

Tool Strip

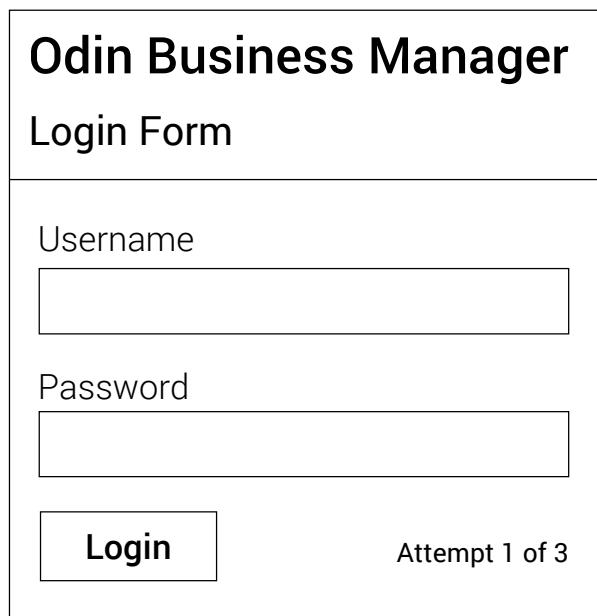
In order to allow for an ease of navigation, I have decided to implement a **Tool Strip** [Fig.D0]. The tool strip will allow access to almost all the forms on the program and will allow additional commands such as 'Logout' and 'Exit'. The tool strip means that less time is required to move between forms thus improving efficiency.

File	View	Help
Account Manager	Main Menu Account Manager Settings Contact Support	Forgot Account Contact Support About
Settings	Travel Application Business Expense Application	
Logout Exit	Travel Expenses Business Expenses	Revenues Investment Wages

Fig.D0

Login Form

This is the **Login Form [Fig.D1]** of the program. Its purpose is to allow the user to enter his or her login details in order to gain access to the system. The form itself does not hold any information but will connect to the database to check if the login details are correct when the button is clicked. Thus, there is no need for the validation of data here.



The image shows a login form titled "Odin Business Manager Login Form". It contains two input fields: "Username" and "Password", each with a corresponding empty rectangular input box. Below these fields are two buttons: a white "Login" button on the left and a grey "Attempt 1 of 3" button on the right.

Fig.D1

Contact Support

This is the form for contacting the developer of the program for support and is hence named the **Contact Support Form [Fig.D2]**. This form allows users to send messages regarding any problems they are facing or general feedback. Validation is required to check that all the text boxes are filled and that the email address is in the correct format. Verification is required by the 'checkbox' which is used to indicate that the email is not spam but an honest enquiry.

Contact Support

Name

Email Address

Message

0/1000 Confirm **Send**

Fig.D2

Validation required:

- Name – Must not exceed 80 characters.
- Email Address – Must be in the correct format.
- Message – Must not exceed 1000 characters.

Account Manager

This is the **Account Manager** [Fig.D3] of the program. It allows the user to add, delete, update and view accounts to the database so that changes in the employees can be catered to.

The user must verify his or her choice when he adds, updates or deletes an account. The program will validate that the username and full name is in the correct format and a password is typed in before performing the relevant operation.

In order to populate the combination boxes, the program will connect to the database once the form opens or any changes are made and fill them with all the existing usernames in the database.

Account Manager	
New Account Username <input type="text"/> Password <input type="text"/> Full Name <input type="text"/> <input type="checkbox"/> Admin? <input type="button" value="Add"/>	Delete Account Username <input type="text"/> <input type="button" value="▼"/> <input type="checkbox"/> Confirm <input type="button" value="Delete"/>
Update Account Username <input type="text"/> <input type="button" value="▼"/> Password <input type="text"/> Full Name <input type="text"/> <input type="checkbox"/> Admin? <input type="button" value="Update"/>	View Accounts <input type="button" value="Clear"/> <input type="button" value="View"/>

Fig.D3

Validation required:

- Username – Must contain 40 characters or less.
- Password – Must contain 50 characters or less.
- Full Name – Must contain 80 characters or less.
- Admin – Checkbox must be either checked or unchecked.

User Settings

This is the **Settings Form [Fig.D4]**. Its purpose is to allow the user to select a different database for data to be read from and also allows the user to back-up the existing database to a directory of his or her choice. As shown in the design the program will clearly display when the database was last backed up.

No validation or verification is required on this form at all.

The diagram shows a window titled "Settings". Inside, under "Database Location", there is a large empty rectangular input field and a "Browse" button. Under "Backup Database", it says "Last backed up on 12/04/2015" next to a "Backup" button. At the bottom are "Help" and "Exit" buttons.

Fig.D4

Main Menu

This is the **Main Menu Form [Fig.D5]**. It contains large circular buttons with icons for ease of navigation and contains all the options to open other forms. No validation of data is required on this form at all.

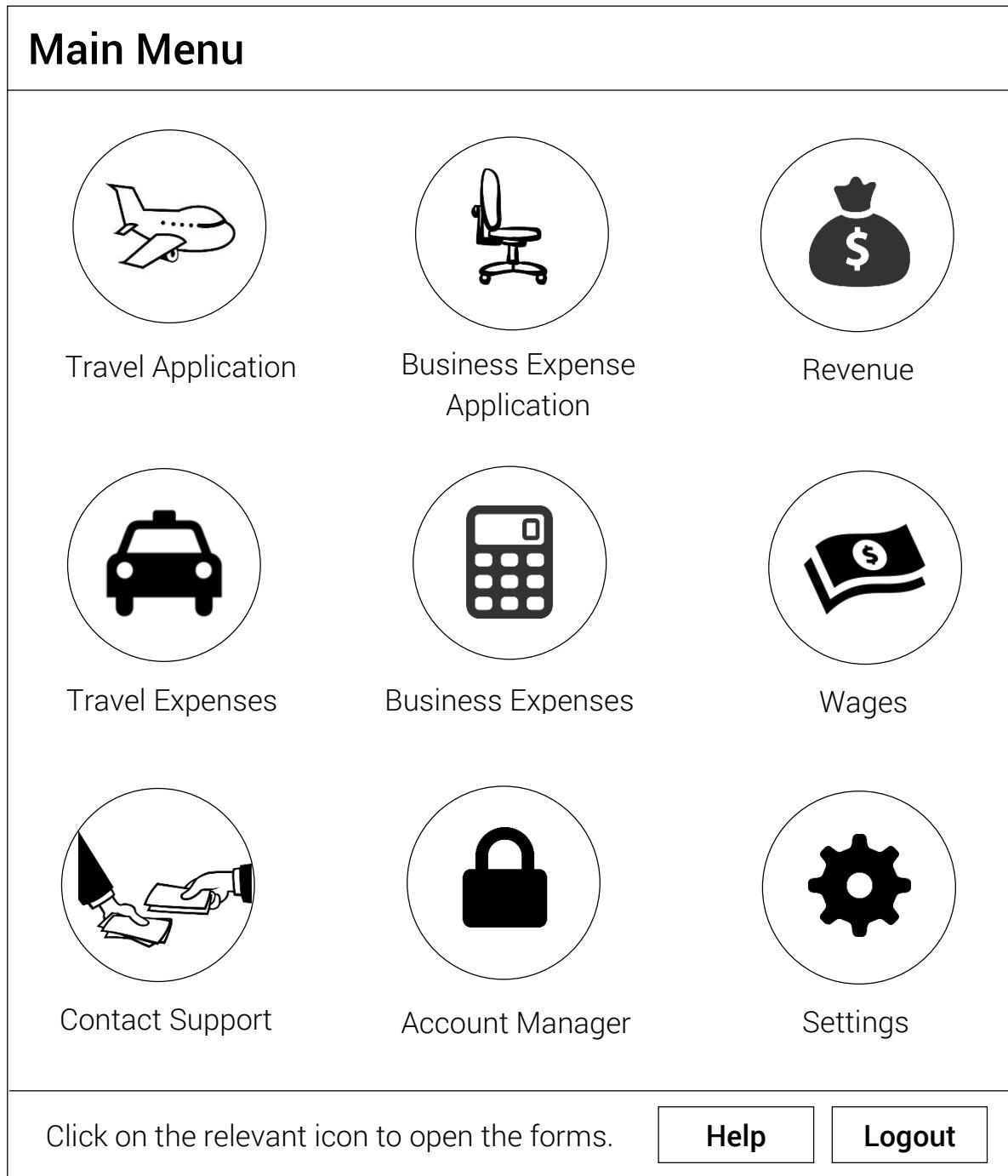


Fig.D5

Travel Application – Overview

This is the main **Travel Application (Overview) Form** [Fig.D6]. Its function is to give an overview (the most important fields) of the travel applications such as pending applications and past applications and contains controls that will allow the user to view, edit and delete these existing applications.

This form also allows the user to search for travel applications by Tracking Number and has a button to open the new Travel Application form. There is no need for advanced filtering or printing as the travel applications are not needed for financial reports and such.

Travel Application – Overview				
Pending Applications (Unapproved)				
Tracking Number	Employee Name	Destinations	Total Cost	Actions
				△
				▽
Past Applications (Approved)				
Tracking Number	Employee Name	Destinations	Total Cost	Actions
				△
				▽
Select a row from the tables above				
View or Edit	Delete	Search	New Application	

Fig.D6

Travel Application – New

This is the form for applying for travel expenses and is called the **Travel Application (New) Form [Fig.D7]**. It contains all the relevant fields which need to be filled in by an employee when he or she makes an application.

The ‘Employee Name’ combination box is populated with all the Full Names that exist inside the Accounts table of the database while the ‘Tracking Number’ is automatically generated by the program.

Travel Application - New	
Employee Name	Tracking Number
<input type="text"/>	<input type="text"/> ▽
Email Address	Duration
<input type="text"/>	<input type="text"/> Days
Destinations (Countries and Cities) <input type="text"/>	
Reason for Trip <input type="text"/>	
<i>Please estimate the following costs:</i>	
Domestic Travel Costs <input type="text"/> NZD	International Travel Costs <input type="text"/> NZD
Accommodation Costs <input type="text"/> NZD	Total Costs <input type="text"/> NZD
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.	
<input type="button" value="Submit"/>	

Fig.D7

Validation required:

- Employee Name – Selected from a list of employee names loaded from the FullName field of the ‘Accounts’ table or entered in. Must not exceed 80 characters.
- Tracking Number – Must be in the correct format (e.g. TV12345678).
- Email – Must be in the form of an email address.
- Destinations – Must not exceed 80 characters.
- Duration – Must range between 1 and 9999 and be numerical only.
- Reason – Must not exceed 280 characters.
- Accommodation Costs – Must be numerical and not exceed 10 characters.
- Domestic Travel Costs – Must be numerical and not exceed 10 characters.
- International Travel Costs – Must be numerical and not exceed 10 characters.
- Total Cost - Must be numerical and not exceed 11 characters.

Travel Application – View and Edit

This is the **Travel Application (View and Edit)** form [Fig.D8]. It allows the user to view a travel application in detail and make edits if required. The 'Approved' checkbox is used by the CEO to approve applications and will only be enabled for an 'Admin'. All the textboxes and such will be automatically filled by the program which will grab the relevant data from the database as according to the tracking number.

Travel Application – View and Edit			
Employee Name	Tracking Number		
<input type="text"/>	<input type="text"/>		
Email Address	Duration		
<input type="text"/>	<input type="text"/>	Days	
Destinations (Countries and Cities) <input type="text"/>			
Reason for Trip <input type="text"/>			
<i>Please estimate the following costs:</i>			
Domestic Travel Costs	International Travel Costs		
<input type="text"/> NZD	<input type="text"/> NZD		
Accommodation Costs	Total Costs		
<input type="text"/> NZD	<input type="text"/> NZD		
<input type="checkbox"/> Approved?	Close	Delete	Save

Fig.D8

Validation required:

- Employee Name – Cannot be changed unless User is admin. Must not exceed 80 characters.
- Tracking Number – Cannot be changed.
- Email – Must be in the form of an email address.
- Destinations – Must not exceed 50 characters.
- Duration – Must range between 1 and 9999 and be numerical only.
- Reason – Must not exceed 280 characters.
- Accommodation Costs – Must be numerical and not exceed 10 characters.
- Domestic Travel Costs – Must be numerical and not exceed 10 characters.
- International Travel Costs – Must be numerical and not exceed 10 characters.
- Total Cost - Must be numerical and not exceed 11 characters.
- Approved – Must be either unchecked or checked.

Business Expense Application – Overview

This is the overview form for **Business Expense Applications** [Fig.D9]. It gives an overview of important fields for each application which is split into pending applications and past applications. The user can select on one of these and then use the controls below to 'view or edit' or 'delete' them.

The form also contains controls for the user to open the new application form or search business expense applications by tracking number. There is no need for advanced filtering or printing as this is not needed in financial reports.

Business Expense Application – Overview				
Pending Applications (Unapproved)				
Tracking Number	Employee Name	Description	Total Price	
				△
				▽

Past Applications (Approved)				
Tracking Number	Employee Name	Description	Total Price	
				△
				▽

Select a row from the tables above

[View or Edit](#) [Delete](#) [Search](#) [New Application](#)

Fig.D9

Business Expense Application – New

This is the form for a **New Business Expense Application** [Fig.D10]. It allows the employee to submit an application for approval and contains all the relevant fields from the database that need to be filled in.

The ‘Employee Name’ combination box is populated with all the Full Names that exist inside the Accounts table of the database while the ‘Tracking Number’ is automatically generated by the program. All the fields must be filled in order for the application to submit.

Business Expense Application – New		
Employee Name		Tracking Number
<input type="text"/>	▼	<input type="text"/>
Email Address	<input type="text"/>	
Item Description		Quantity
<input type="text"/>		<input type="text"/>
Unit Price	Total Price	Vendor
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/>
Reason for Purchase <input type="text"/>		
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.		<input type="button" value="Submit"/>

Fig.D10

Validation required:

- Employee Name – Selected from a list of employee names loaded from the FullName field of the ‘Accounts’ table or entered. Must not exceed 80 characters.
- Tracking Number – Must be in the correct format (e.g. BU12345678).
- Email – Must be in the form of an email address.
- Item Description – Must not exceed 100 characters.
- Quantity – Must range between 1 and 99999 and be numerical only.
- Unit Price – Must be numerical and not exceed 10 characters.
- Total Price – Must be numerical and not exceed 14 characters.
- Vendor – Must not exceed 50 characters.
- Reason – Must not exceed 280 characters.

Business Expense Application – View and Edit

This is the **Business Expense Application (View and Edit)** Form [Fig.D11]. It allows the user to edit and view their applications if required. The 'Approved' checkbox is used by the CEO to approve applications and will only be enabled for an 'Admin'.

All the textboxes will be automatically filled by the program which will grab the relevant data from the database as according to the tracking number. The form also contains controls for closing, deleting and saving the application.

Business Expense Application – View and Edit		
Employee Name	Tracking Number	
<input type="text"/>	<input type="text"/>	
Email Address		
<input type="text"/>		
Item Description	Quantity	
<input type="text"/>	<input type="text"/>	<input type="text"/>
Unit Price	Total Price	Store
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/>
Reason for Purchase		
<input type="text"/>		
<input type="checkbox"/> Approved?	<input type="button" value="Close"/>	<input type="button" value="Delete"/>
<input type="button" value="Save"/>		

Fig.D11

Validation required:

- Employee Name – Cannot be changed unless user is admin. Must not exceed 80 characters.
- Tracking Number – Cannot be changed.
- Email – Must be in the form of an email address.
- Item Description – Must not exceed 100 characters.
- Quantity – Must range between 1 and 99999 and be numerical only.
- Unit Price – Must be numerical and not exceed 10 characters.
- Total Price – Must be numerical and not exceed 14 characters.
- Vendor – Must not exceed 50 characters.
- Reason – Must not exceed 280 characters.
- Approved – Must be either unchecked or checked.

Travel Expenses – Overview

This is the **Travel Expense (Overview)** form [Fig.D12]. Its purpose is to allow the user to receive an overview of the travel expenses which are drawn from the database and show the most important fields.

Furthermore, this form contains controls which allow the user to view and edit a travel expense in further detail, delete it, search or print for a travel expense through the tracking number and create a new travel expense.

No validation or verification of data is required on this form.

Fig.D12

Travel Expenses – New, View and Edit

This form is the **Travel Expenses (New, View and Edit)** form [Fig.D13]. The purpose of this form is to allow the user to either make a new travel expense based on the tracking number or view or edit a previous travel expense.

It shows all the expenses entered for one tracking number and also gives the control to print the expenses with automatic calculation and formatting.

Travel Expense – New, View and Edit																																								
Tracking Number	Employee Name																																							
<input type="text"/>	<input type="text"/>																																							
Date Incurred	Expense Type	Payment Method																																						
<input type="text"/> ▽	<input type="text"/> ▽	<input type="text"/> ▽																																						
Item Description			<input type="checkbox"/> Bill Client? <input type="checkbox"/> Reimbursed?																																					
Foreign Amount	Type	Exchange Rate	NZD Amount	NZD																																				
<input type="text"/>	<input type="text"/> ▽	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/> <input type="button" value="Add"/>																																				
<table border="1"><thead><tr><th>Date Incurred</th><th>Expense Type</th><th>Description</th><th>NZD</th><th>.....</th><th>△</th></tr></thead><tbody><tr><td></td><td></td><td></td><td></td><td></td><td>▽</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr></tbody></table>					Date Incurred	Expense Type	Description	NZD	△						▽																								
Date Incurred	Expense Type	Description	NZD	△																																			
					▽																																			
<input type="button" value="Print"/>	<input type="button" value="Delete"/>	<input type="button" value="View & Edit"/>																																						

Fig.D13

Validation required:

- Tracking Number – Must not exceed 10 characters.
- Employee Name – Must be selected from a list of employee names loaded from the FullName field of the ‘Accounts’ table or entered in. Must not exceed 80 characters.
- Date Incurred – Must be in the form of a date.
- Expense Type – Must be selected from a list of predefined expense types.
- Payment Type – Must be selected from a list of predefined payment methods.
- Item Description – Must not exceed 100 characters.
- Bill Client – Must be either checked or unchecked.
- Reimbursed – Must be either checked or unchecked.
- Foreign Amount – Must be numerical and must not exceed 10 characters.
- Foreign Currency Type – Must be selected from a list of predefined currencies or entered in. Must be text only, in capital letters and be exactly 3 characters long.
- Exchange Rate – Must be numerical and not exceed 10 characters.
- NZD Amount – Calculated automatically.

Business Expenses – Overview

This is the **Business Expenses (Overview)** form [Fig.D14]. It gives an overview of the business expenses including the ones that need to be reimbursed to the employee. This form also gives controls to view or edit, delete, print and search for these expenses as well as create a new expense.

The data on the tables is drawn from the 'Business Expense' table of the database with expenses not marked as 'Reimbursed' put into the pending segment of the form. No validation or verification is required on this form.

Fig.D14

Business Expenses – New

This is the **Business Expenses (New)** form [Fig.D15]. The use of this form is to allow the user to add a new business expense. It contains all the fields indicated earlier in the fields list with no modifications apart from the names.

Once all the textboxes and combination boxes have been filled in or selected the user can then 'Add' this new record into the database.

Business Expense - New		
Tracking Number	Employee Name	
<input type="text"/>	<input type="text"/>	
Date Purchased	Vendor	Payment Method
<input type="text"/> <input type="button" value="▼"/>	<input type="text"/>	<input type="text"/> <input type="button" value="▼"/>
Item Description	Quantity	
<input type="text"/>	<input type="text"/>	
Unit Price	Total Price	<input type="checkbox"/> Reimbursed
<input type="text"/> NZD	<input type="text"/> NZD	<input type="checkbox"/> Reimbursed
<input type="button" value="Close"/>		<input type="button" value="Add"/>

Fig.D15

Validation required:

- Tracking Number – Must not exceed 10 characters.
- Employee Name – Loaded automatically by checking the Business Expense Applications table. Must not exceed 80 characters.
- Date Purchased – Must be in the form of a date.
- Vendor – Must not exceed 50 characters.
- Payment Type – Must be selected from a list of predefined payment methods.
- Item Description – Must not exceed 100 characters.

- Quantity – Must be numerical and range from 1 to 99999.
- Unit Price – Must be numerical and not exceed 10 characters.
- Total Price – Must be numerical and not exceed 14 characters.
- Reimbursed – Must be either checked or unchecked.

Business Expenses – View and Edit

This form is the **Business Expenses (View and Edit) Form** [Fig.D16]. Its purpose is to allow the user to view the data on a business expense in more detail and allow him or her to edit it if necessary. To access this form, the user selects a business expense on the overview form and then clicks the “View and Edit” button. The textboxes in this form are automatically filled in with the data relevant to the Tracking Number.

Business Expense – View and Edit		
Tracking Number	Employee Name	
<input type="text"/>	<input type="text"/>	
Date Purchased	Vendor	Payment Method
<input type="text"/> 	<input type="text"/>	<input type="text"/> 
Item Description		Quantity
<input type="text"/>		<input type="text"/>
Unit Price	Total Price	
<input type="text"/> NZD	<input type="text"/> NZD	<input type="checkbox"/> Reimbursed
Close		Save

Fig.D16

Validation required:

- Tracking Number – Cannot be changed.

- Employee Name – Cannot be changed unless user is admin. Must not exceed 80 characters.
- Date Purchased – Must be in the form of a date.
- Vendor – Must not exceed 50 characters.
- Payment Type – Must be selected from a list of predefined payment methods.
- Item Description – Must not exceed 100 characters.
- Quantity – Must be numerical and range from 1 to 99999.
- Unit Price – Must be numerical and not exceed 10 characters.
- Total Price – Must be numerical and not exceed 14 characters.
- Reimbursed – Must be either checked or unchecked.

Revenues – Overview

This is the summary form the revenues called the **Revenues (Overview) Form [Fig.D17]**. Firstly, it allows the user to view revenue information of the most important fields such as purchase order number, date, customer, etc. Furthermore, it allows the user to delete the revenue or open another form to view and edit it in closer detail.

The form also contains controls to search or create a new record and also contains a control to filter the results in the table by dates. This is extremely useful if the user wants to view revenues between certain time periods.

Revenues – Overview				
PO No.	Invoice No.	Date	Customer
				△
				▽

New Record **Search** **Delete** **View and Edit**

Filter By:

Payments Between & ▽ ▽

Clear **Filter**

Fig.D17

Validation required:

- Payments Between – Must be in the format of a date.

Revenues – New

This is the form that will be used to add a new record regarding revenues and is called the **Revenues (New) Form [Fig.D18]**. It contains all the fields indicated earlier in the fields list excluding the GST Paid as this has no effect on Odin but on the customer. All this information is sourced from the sales invoice.

The unique goods combination box will allow further goods to appear on the form so that more items can be entered into the revenues database.

Revenues - New		
PO Number	Invoice Number	Invoice Date
<input type="text"/>	<input type="text"/>	<input type="text"/> <input type="button" value="▼"/>
Customer (Purchaser)	Unique Goods	
<input type="text"/>	<input type="text"/> <input type="button" value="▼"/>	
Description (good or service) #1	Quantity	
<input type="text"/>		<input type="text"/>
Unit Price (exc GST)	Subtotal	
<input type="text"/>	NZD	<input type="text"/> NZD
.....		
<input type="button" value="Close"/>	<input type="button" value="Add"/>	

Fig.D18

Validation Required:

- Purchase Order Number – Must be not exceed 10 characters.

- Invoice Number – Must not exceed 10 characters.
- Invoice Date – Must follow the format of a date.
- Customer (Purchaser) – Must not exceed 50 characters.
- Description – Must not exceed 100 characters.
- Quantity – Must range between 1 and 99999 and must be numerical.
- Unit Price – Must be numerical and not exceed 10 characters.
- Subtotal – Must be numerical and not exceed 10 characters.

Revenues – View and Edit

This form is the **Revenues (View and Edit) Form** [Fig.D19]. Its purpose is to allow the user to view the data on a revenue in more detail and allow him or her to edit it if necessary. To access this form, the user selects a revenue on the overview form and then clicks the “View and Edit” button. The textboxes in this form are automatically filled in with the data relevant to the PO Number.

Revenues – View and Edit		
PO Number	Invoice Number	Invoice Date
<input type="text"/>	<input type="text"/>	<input type="text"/> 
Customer (Purchaser)	Unique Goods	
<input type="text"/>	<input type="text"/> 	
Description (good or service) #1	Quantity	
<input type="text"/>	<input type="text"/>	
Unit Price (exc GST)	Subtotal	
<input type="text"/> NZD	<input type="text"/> NZD	
.....		
Close	Save	

Fig.D19

Validation Required:

- Purchase Order Number – Cannot be changed
- Invoice Number – Must not exceed 10 characters.
- Invoice Date – Must follow the format of a date.
- Customer (Purchaser) – Must not exceed 50 characters.
- Description – Must not exceed 100 characters.
- Quantity – Must range between 1 and 99999 and must be numerical.
- Unit Price – Must be numerical and not exceed 10 characters.
- Subtotal – Must be numerical and not exceed 10 characters

Investment

This form is the **Investment Form [Fig.D20]**. Its purpose is to show and record all investment data. Since the investment data is not very important in a record-keeping sense, only basic fields are required. The table-based layout will give an overview of the investments that have taken place over a period of time.

Investment			
Date	Investor	Amount	
			△
			▽

Investor			
<input type="text"/>			
Investment Date	Injection Amount	NZD	Add
<input type="text"/> ▽	<input type="text"/>	<input type="text"/>	<input type="button" value="Add"/>

Fig.D20

Validation required:

- Investor – Must not exceed 80 characters
- Investment Date – Must be in the form of a date.
- Injection Amount – Must be numerical and not exceed 10 characters.

Wages – Overview

This is the main form for viewing wages and is called the '**Wages – Overview Form [Fig.D21]**'. It gives an overview of important fields for each of the wage payments. The user can select on one of these and then use the controls below to 'view or edit' or 'delete' them.

The form also contains controls for the user to open the new wages form or to filter the results in the table so he or she can print a more customised wages sheet from specific dates or for a specific employee.

Wages – Overview				
From	To	Name	Date Paid
				△
				▽

New Record **Delete** **View and Edit** **Print**

Filter By:

Payments Between ▽ & ▽

Employee Name ▽

Close **Clear Filters** **Filter**

Fig.D21

Validation required:

- Payments Between – Must be in a date format.
- Employee Name – Must be selected from a list of employee names loaded from the FullName field of the 'Accounts' table or inputted by the user. Must not exceed 80 characters.

Wages – New

This is the **Wages (New) Form [Fig.D22]** which will be used to add new information about Wage payments. It allows the accountant to enter in the relevant data about wage payments by using intuitive calendar drop-downs and automatic calculation of the 'Net Pay'.

All the textboxes and combination boxes must be filled in order for the new data to be saved into the database.

Wages – New		
Period for Payment		
From	<input type="text"/> ▼	To <input type="text"/> ▼
Employee Name	Date Paid	
<input type="text"/>		<input type="text"/> ▼
Gross Pay	PAYE	KiwiSaver
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/> NZD
Net Pay	<input type="text"/> NZD	<input type="button" value="Add"/>

Fig.D22

Validation required:

- Period for Payment – Must be in a date format.
- Employee Name – Must be selected from a list of employee names loaded from the FullName field of the 'Accounts' table or entered by user. Must not exceed 80 characters.
- Date Paid – Must be in the form of a date.
- Gross Pay – Must be numerical and not exceed 10 characters.
- PAYE – Must be numerical and not exceed 10 characters.
- KiwiSaver – Must be numerical and not exceed 10 characters.
- Net Pay – Must be numerical and not exceed 10 characters.

Wages – View and Edit

This is the **Wages (View and Edit) form** [Fig.D23]. It allows the user to view and edit wage payments if required. Furthermore, this form allows the user to view wage payments in further detail than the 'overview' form.

All the textboxes and such will be automatically filled by the program which will fetch the relevant data from the database according to what the user selected in the overview form.

Wages – View and Edit		
Period for Payment		
From	<input type="text"/>	▽
To	<input type="text"/>	▽
Employee Name	Date Paid	
<input type="text"/>	<input type="text"/> ▽	
Gross Pay	PAYE	KiwiSaver
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/> NZD
Net Pay	<input type="text"/> NZD	<input type="button" value="Delete"/> <input type="button" value="Save"/>

Fig.D23

Validation required:

- Period for Payment – Must be in a date format.
- Employee Name – Cannot be changed unless user is admin. Must not exceed 80 characters.
- Date Paid – Must be in the form of a date.
- Gross Pay – Must be numerical and not exceed 10 characters.
- PAYE – Must be numerical and not exceed 10 characters.
- KiwiSaver – Must be numerical and not exceed 10 characters.
- Net Pay – Must be numerical and not exceed 10 characters.

Report Design

Business Expenses

Business Expenses						
Tracking No.	Name	Date	Description	Total	Vendor	Reimburse
BS123456	WS	16/02/15	Item	\$4000	Store	True
BS123456	WS	16/02/15	Item	\$4000	Store	True

Travel Expenses

Travel Expenses – TV123456						
Date	Description	Type	Amount	XRT	NZD	Repaid
11/02/15	Hotel	Hotel	RMB \$3000	0.25	\$640	Yes
11/02/15	Food	Meals	RMB \$500	0.25	\$105	Yes

Wages

Wages						
Dates	Name	Date Paid	Gross Pay	PAYE	KiwiSaver	Net Pay
14/01/15 to 16/01/15	WS	16/02/15	\$10000	\$1000	\$1000	\$8000
18/01/15 to 20/01/15	WS	16/02/15	\$20000	\$1000	\$1000	\$18000

Testing Plan

It is vitally important that the proposed system is tested to check if it is working and if it is suitable to be implemented. Thus, the testing data should be decided on beforehand and the expected output planned. Here is a brief outline of the testing plan.

The testing strategy I have adopted ensures that all parts of the program will be able to handle typical and extreme data whilst rejecting erroneous data. As shown in the User Interface Design, many forms require validation and this will play a large role in determining whether data is valid or not.

Furthermore, all controls must work precisely and different combinations of commands will be tested to ensure that the system runs smoothly. To test the program, data from previous Travel Expenses, Wage payments, Revenues, etc. will be input into the program and the output will be analysed to see if it meets the requirements specification. These documents have kindly been provided by the client and the expected results have also been delivered; snippets of them may be found in the 'Current Documentation' section earlier in this document. The printed output will also be tested to ensure that it produces the correct data values and is not distorted.

There will be several more parts of the program which will be subject to testing to ensure that the system created works fully. Further analysis of the testing plan and actually carrying it out will occur later on in this project once the system has been developed.

The next page contains the tables including different paths that will be taken to test the program. Please note that these tables have been updated as the program was being developed so they may contain new forms or controls.

About Form

Test	Expected Outcome	Result	Evidence
Close Button	Closes the form		
Queries & Support Label	Opens Contact Support form		

Account Manager Form

Test	Type of Data	Expected Outcome	Result	Evidence
Enter Button with correct password in textbox	N/A			
Enter Button with incorrect password in textbox	N/A			
Add Button with relevant values in textboxes	Normal			
Add Button with extreme values in the textboxes	Extreme			
Add Button with no values in the textboxes	Invalid			
Delete Button with confirm checkbox checked	N/A			
Delete Button with confirm checkbox unchecked	N/A			
View Button	N/A			
Clear Button	N/A			
Update Button with relevant values in textboxes	Normal			
Update Button with extreme values in the textboxes	Extreme			
Update Button with no values in the textboxes	Invalid			
Close Button	N/A			
Lock Controls Button	N/A			

Business Expenses – Overview Form

Test	Expected Outcome	Result	Evidence
New Expense Button clicked	Business Expense - New Form opened		
Search Button clicked and existing Tracking Number entered.	Business Expense – View and Edit Form opened with preloaded details		
Item selected on pending ListView and Delete Button clicked	Record deleted from database and ListView updated		
Item selected on completed ListView and Delete Button clicked	Record deleted from database and ListView updated		
Item selected on pending ListView and View or Edit Button clicked	Business Expense – View and Edit Form opened with loaded details		
Item selected on completed ListView and View or Edit Button clicked	Business Expense – View and Edit Form opened with loaded details		
Print All Data and Find Total Checkbox checked and Filter Button clicked	Print Dialog opened then Print Preview Dialog opened with correct data with total calculated		
Print All Data and Find Total Checkbox unchecked and Filter Button clicked	Print Dialog opened then Print Preview Dialog opened with correct data with no total calculated		
Filter Button clicked with filters applied	The correct data is filtered		
Print All Data Checkbox checked with other checkboxes checked	The other checkboxes are unchecked		
Clicking on the header of the columns of the ListView	Sorts the relevant columns		

Business Expenses – New Form

Test	Type of Data	Expected Outcome	Result	Evidence
Tracking Number that exists in the BEA table is entered into textbox	Normal	The relevant data from the BEA record is loaded into the controls		
Tracking Number that does not exist in the BEA table is entered into textbox	Invalid	The controls are set to blank		
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Save Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated		
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database and the ListView is updated		
Save Button is clicked with controls left blank	Invalid	Output error message to user		
Save Button is clicked but a record with the tracking number already exists	Invalid	Output error message to user		
User selects 'Company' as Payment Method	N/A	Reimbursement checkbox is unchecked and disabled		
The text in the Quantity or Unit Price textbox is changed	N/A	The Total Price is automatically calculated		

Business Expenses - View and Edit Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated		
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated		
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated		
Update Button with no data in controls	Invalid	Output error message to user		
The text in the Quantity or Unit Price textbox is changed	N/A	The Total Price is automatically calculated		

Business Expense Applications – Overview Form

Test	Expected Outcome	Result	Evidence
New Application Button clicked	Business Expense Application - New Form opened		
Search Button clicked and existing Tracking Number entered.	Business Expense Application - View and Edit Form opened with preloaded details		
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated		
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details		
Clicking on headers of the columns of the ListView	Sorts the relevant columns		

Business Expense Application – New Form

Test	Type of Data	Expected Outcome	Result	Evidence
Form loaded	N/A	Tracking Number automatically assigned		
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Save Button is clicked with valid data in the controls	Normal	Data is saved to database, email sent and ListView updated.		
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database, email sent and ListView updated.		
Save Button is clicked with controls left blank	Invalid	Output error message to user		
Save Button is clicked with valid data but no internet connection.	N/A	Output message telling user record was saved but email was not sent		
The text in the Quantity or Unit Price textbox is changed	N/A	The Total Price is automatically calculated		

Business Expense Application - View and Edit Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated		
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated		
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated		
Update Button with no data in controls	Invalid	Output error message to user		
Update Button with Approved checkbox checked from an initial unchecked state	Normal	The record is updated in the database, email sent to the applicant and ListView updated.		

Contact Support Form

Test	Type of Data	Expected Outcome	Result	Evidence
Send Button with relevant data in textboxes and richtextbox. Checkbox checked.	Normal	Sends email to the support email address		
Send Button with relevant data in textboxes and richtextbox. Checkbox unchecked.	Normal	Outputs error message		
Send button with relevant data in controls	Extreme	Sends email		
Send Button with email address in incorrect format	Invalid	Outputs error message		
Send Button with no data in controls	Invalid	Outputs error message		

Investment Form

Test	Type of Data	Expected Outcome	Result	Evidence
Add Button with data in controls	Normal	Adds record to database. ListView updated	Success	
Add Button with data in controls	Extreme	Adds record to database. ListView updated	Success	
Add Button with no data in controls	Invalid	Outputs error message	Success	
Delete Button with item in ListView selected	N/A	Deletes record from database. ListView updated	Success	
View and Edit Button with item in ListView selected	N/A	Opens the Investment – Edit form with relevant data	Success	
Clicking on the header of the columns of the ListView	N/A	Sorts the relevant columns	Success	

Investment Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button	N/A	Closes the form		
Update Button with data in controls	Normal	Updates the record in the database. ListView on overview form updated		
Update Button with data in controls	Extreme	Updates the record in the database. ListView on overview form updated		
Update Button with no data in controls	Invalid	Output error message to user		

Login Form

Test	Type of Data	Expected Outcome	Result	Evidence
Login Button with incorrect username and password	Invalid	Outputs error message. Increases attempts by 1		
Login Button with correct username and password	Normal	Outputs success message and closes Login.vb and opens the Main Menu		
Login Button with incorrect username and password when at attempt 5 out of 5	Invalid	Outputs error message and exits the program		
All the controls on the ToolStrip	N/A	Opens the relevant form or outputs a relevant message		

Main Menu Form

Test	Expected Outcome	Result	Evidence
All the controls on the ToolStrip	Opens the relevant forms		
Travel Application Button Clicked	Opens Travel Application - Overview form		
Business Application Button Clicked	Opens Business Expense Application - Overview form		
Revenues Clicked	Opens Revenues - Overview form		
Travel Expenses Button Clicked	Opens Travel Expense - Overview form		
Business Expenses Button Clicked	Opens Business Expense - Overview form		
Contact Support Button Clicked	Opens Contact Support form		
Investment Button Clicked	Opens Investment form		
Wages Button Clicked	Opens Wages - Overview form		
Account Manager Button Clicked	Opens Account Manager form		
Settings Button Clicked	Opens Settings form		
Help Button clicked	Shows message with instructions		
Logout Button clicked	Closes all open forms and then opens Login form		
Exit Button clicked and 'Yes' selected	Exits the application		

Revenues – Overview Form

Test	Expected Outcome	Result	Evidence
New Revenue Button clicked	Revenues - New Form opened		
Search Button clicked and existing PO Number entered	Revenues - View and Edit Form opened with preloaded details		
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated		
Item selected on ListView and View or Edit Button clicked	Revenues - View and Edit Form opened with loaded details		
Dates selected and Filter Button clicked	Correct data filtered and displayed on the ListView		
Clear Button clicked	The filter is cleared and ListView refreshed with all the data in the Revenues table.		
Clicking on headers of the columns of the ListView	Sorts the relevant columns		

Revenues – New Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Add Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated		
Add Button is clicked with extreme data in the controls	Extreme	Data is saved to database and the ListView is updated		
Add Button is clicked with controls left blank	Invalid	Output error message to user		
Add Button is clicked but a record with the PO Number already exists	Invalid	Output error message to user		
The text in the Subtotal or GST textbox is changed	N/A	The Total is automatically calculated		

Revenues - View and Edit Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Closes the form		
Delete Button is clicked and user selects 'Yes'	N/A	Deletes the record from the database and updates the ListView		
Update Button with data in controls	Normal	Updates the record in the database. ListView on overview form updated		
Update Button with data in controls	Extreme	Updates the record in the database. ListView on overview form updated		
Update Button with no data in controls	Invalid	Output error message to user		
The text in the Subtotal or GST textbox is changed	N/A	The Total is automatically calculated		

Search Box Form

Test	Expected Outcome	Result	Evidence
Search Button clicked with valid query in textbox	Opens relevant form with data loaded		
Cancel Button clicked	Form closes		

Settings Form

Test	Type of Data	Expected Outcome	Result	Evidence
Browse Button clicked and new valid file selected	Valid	Updates the setting for the database location		
Backup Button clicked and save selected	Valid	A copy of the database is copied into the specified directory		
Update Button clicked with valid email in textbox	Valid	Settings for application email address updated		
Update Button clicked with extreme email in textbox	Extreme	Settings for application email address updated		
Update Button clicked with invalid email in textbox	Invalid	Output error message		
View Variables Button clicked	N/A	Opens Variables Settings form		
Help Button clicked	N/A	Message box with information shown		
Close Button clicked	N/A	The form is closed		

Settings Variables Form

Test	Expected Outcome	Result	Evidence
Enter Button clicked with correct password in textbox	Form enabled and settings variable values loaded		
Enter Button clicked with incorrect password in textbox	Output error to user		
Close Button clicked	The form is closed		

Status Box Form

Test	Expected Outcome	Result	Evidence
OK Button is clicked	Form closes		

Travel Application – Overview Form

Test	Expected Outcome	Result	Evidence
New Application Button clicked	Business Expense Application - New Form opened		
Search Button clicked and existing Tracking Number entered.	Business Expense Application - View and Edit Form opened with preloaded details		
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated		
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details		
Clicking on headers of the columns of the ListView	Sorts the relevant columns		

Travel Application – New Form

Test	Type of Data	Expected Outcome	Result	Evidence
Form loaded	N/A	Tracking Number automatically assigned		
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Save Button is clicked with valid data in the controls	Normal	Data is saved to database, email sent and ListView updated.		
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database, email sent and ListView updated.		
Save Button is clicked with controls left blank	Invalid	Output error message to user		
Save Button is clicked with valid data but no internet connection.	N/A	Output message telling user record was saved but email was not sent		
The text in the Domestic, International or Accommodation costs textbox is changed	N/A	The Total costs is automatically calculated		

Travel Application - View and Edit Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated		
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated		
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated		
Update Button with no data in controls	Invalid	Output error message to user		
Update Button with Approved checkbox checked from an initial unchecked state	Normal	The record is updated in the database, email sent to the applicant and ListView updated.		
The text in the Domestic, International or Accommodation costs textbox is changed	N/A	The Total costs is automatically calculated		

Travel Expense – Overview Form

Test	Expected Outcome	Result	Evidence
New Expense Button clicked	Business Expense Application - New Form opened		
Search Button clicked and existing Tracking Number entered.	Business Expense Application - View and Edit Form opened with preloaded details		
Item selected on ListView and Print Button clicked	Brings up print dialog and then print preview dialog with all data on the relevant tracking number		
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated		
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details		

Travel Expense Form

Test	Type of Data	Expected Outcome	Result	Evidence
Tracking Number that exists in the TEA table is entered into textbox	Normal	The relevant data from the TEA record is loaded into the controls. ListView is updated		
Tracking Number that does not exist in the TEA table is entered into textbox	Invalid	The controls are set to blank		
Save Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated		
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database and the ListView is updated		
Save Button is clicked with controls left blank	Invalid	Output error message to user		
The Clear Button is clicked	N/A	Any text in controls is cleared		
New Expense Button clicked	N/A	The form is reset to accept a new record entry		
Item is selected in ListView and Delete Button (bottom) clicked. User selects 'Yes'	N/A	Record is deleted from database and ListView updated		
Item is selected in ListView and View or Edit Button clicked.	N/A	The relevant details for the expense is loaded onto the form		
Record loaded to View and Edit and Delete Button clicked	N/A	Record deleted from database and ListView updated		
Update Button clicked with normal data in controls	Normal	Record and ListView updated		

Test	Type of Data	Expected Outcome	Result	Evidence
Update Button clicked with extreme data in controls	Extreme	Record and Listview updated		
Update Button clicked with no data in controls	Invalid	Error message output		
All Reimbursed Button clicked and 'Yes' selected	N/A	All Records have their Reimbursed field set to true		
Print Button clicked	N/A	All records for the tracking expense are collected into a report		
Clicking on headers of the columns of the ListView	N/A	Sorts the relevant columns		

Wages - Overview Form

Test	Expected Outcome	Result	Evidence
New Record Button clicked	Wages - New Form opened		
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated		
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details		
Filter Button clicked with filter options selected	Data is filtered and the ListView is updated accordingly		
Clear Filter Button is clicked	The ListView is reset and all data is viewed		
Print All Data radio button is selected and Print Button is clicked	Print dialog and print preview dialog with all data from the Wages table is produced into a report		
Print Filtered Data radio button is selected and Print Button is clicked	Print dialog and print preview dialog with all data from the ListView produced into a report		
Clicking on headers of the columns of the ListView	Sorts the relevant columns		

Wages – New Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Save Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated		
Save Button is clicked with borderline data in the controls	Extreme	Data is saved to database and the ListView is updated		
Save Button is clicked with no data in controls	Invalid	Output error message		
The text in the Gross Pay, PAYE or KiwiSaver text box is changed	N/A	The Net Pay is automatically calculated		

Wages - View and Edit Form

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed		
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated		
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated		
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated		
Update Button with no data in controls	Invalid	Output error message to user		
The text in the Gross Pay, PAYE or KiwiSaver text box is changed	N/A	The Net Pay is automatically calculated		

User Agreement

As the potential user of this proposed solution, I agree to the following regarding the design of the system:

- The tables proposed hold all the required data.
- The system flowcharts outlining the process the new system will use is correct and above satisfaction.
- The design of the forms meet my requirements and hold all the data that is required.
- The design of the forms is aesthetically pleasing yet functional and will be efficient to use.
- There will be no major changes to the data structure and forms (e.g. new table, new section of system, etc.)
- The proposed system offers a large improvement over the current system.

USER COMMENTS

The design specification meets all the my requirements. Furthermore, I have been consulted throughout this process and am satisfied with the form design, data structures, etc.

Furthermore I agree to the design philosophy and the points listed above regarding the system.

Signed

Date: 23/04/15

Intended Benefits

The new system is a drastic improvement over the current, almost non-existent, unorganised system. There are many intended benefits that will help the client in creating a more sustainable business:

Automatic Emailing

In the cases of Travel Expense Applications and Business Expense Applications, users can easily and quickly fill in the fields required for the CEO or relevant person to review their application and efficiently accept it.

When the user or employee submits their application, an email is automatically sent by the program to the approval person allowing them to review the relevant details. He or she then goes on the proposed solution and can review them again in further detail if required and then approve it.

Once the application is approved, an email will be automatically sent to the employee telling them that their application has been successfully approved and they can go purchase or plan whatever they have earlier indicated.

Efficiency

The new system allows the users to easily locate past expenses, applications, etc. in comparison with the current system which consists of countless Excel spreadsheet files which are unorganised and uncollated.

Furthermore, the Excel files do not need to be created and formatted every single time a new record needs to be added but instead intuitively added into the database using a pre-designed form based interface.

This is a major improvement over the current system and would drastically improve efficiency and record-keeping procedures. The proposed solution is very organised and intuitive to use.

Advanced Data Viewing and Filtering System

Data which is to be viewed in a tablature format can be sorted via the columns and filtered through a filtering system based on employee name, tracking number, etc. Only the important fields will be shown at first and each record can be expanded to give a wider view.

This allows the user to view data that he or she wants to see and removes any redundant data for viewing. By doing this, the overall efficiency and productivity of the business is improved as less time is required to search through records.

Currently there is no way to filter data as everything is separated and scattered about in several Excel spreadsheets.

Automatic Calculation

In recording the travel expenses, the system will automatically cost the total cost of the business trip and the amount to be reimbursed to the employee. This can save time going through spreadsheets slowly adding together expenses and reimbursement amounts. This is also applied to business expenses, wages, etc.

Thus as shown, the new system would reduce the time significantly required to calculate relevant values as data is automatically taken from the relevant data tables and the relevant values are calculated.

Printing of Documents

The new system will allow for the printing of relevant data in an aesthetically pleasing manner. The user can easily select what he or she wants to print and then print it with the press of a button.

Limits of the Scope of the Solution

As with almost all solutions, there are limitations with the proposed system. Although these limitations have no significant effect on the overall performance of the system, they should be both understood by the client and the developer of the solution.

Outdated Forms

If any changes to the forms or databases are required, then this would require the forms to be redesigned and the database to be updated. This is relatively easy to do given whoever is updating the solution has a sufficient knowledge of using Access and Visual Basic .NET.

However, there is a very little chance for this to occur as the existing forms have been tested over time and tailored to what data is required. Thus, this is not a significant limit that could be easily solved.

Compatibility

The solution will be produced with Visual Basic .NET with an Access 2013 database. This means that the program will only be compatible with computers running the .NET Framework and an Access Database engine which can only be installed on Windows computer.

Although all the employees of Odin Health use computers with a Windows 7, 8 or 8.1 environment, this means that the program cannot be used on their phones, tablets, etc.

The environment the program has been chosen to run on reflects the main operating system the company uses, which is Windows.

Database Access

As the main database will be stored on Odin's network, this means that the 'real' database may only be accessed when a computer is connected to this network.

However, it has been agreed by the client and I that the solution will contain an option to change the location of the database and to back it up thus allowing users access when not connected to the network database. This is perhaps the most significant limitation of the proposed system but can be easily overcome as the system only needs to be accessed when in Odin's offices.

Storage

The .NET Framework requires approximately 250MB of storage space. However, this should already be installed in some employee's computers as some aspects of Odin's software are developed in a .NET language and thus would not be a major problem.

The proposed solution would also require a Microsoft ACE OLEDB Provider as the database engine. This would require approximately 50MB of storage space which would be negligible to the total storage space of the company computers.

As calculated earlier, the estimated size of the records in the database would be 0.915 Megabytes. However, table design and such must be also accounted for adding upon 1-2 Megabytes maximum. Nevertheless, this size is very small in comparison to today's storage standards.

The maximum size for Access databases is 2GB. If Odin Health were to use the program for a long time, then the database would slowly fill up but would reach nowhere near 2GB as shown by the earlier database size calculation. However, if the maximum were to be reached, then a fresh database would need to be used.

Given the current storage space options in computer systems, the size of the .NET Framework and database engine only accounts for a fraction of the total storage space on the computer. This would account for approximately 1% of a 256GB Solid State Drive or a mere 0.03% for a 1TB Hard Drive. The size of the proposed system is insignificant to the storage space available on modern computers and is thus trivial limitation.

Upgrading the System

If the system were to be upgraded then whoever is upgrading the system requires extensive knowledge of the Visual Basic .NET language and SQL Query formats. All form design and programming is completed in Visual Studio 2013. To update the database on the other hand, would require knowledge of the workings of Microsoft Access.

These requirements in order to update the system are somewhat specific and pose significant limitations if the system is to be updated. However, the project files will be provided to the client to allow the updating procedure, if it were to occur, run more smoothly and efficiently.

Development

Data Structures

Accounts Table

Accounts			
	Field Name	Data Type	Description (Optional)
Username	Short Text	The username which the user will login to the program with	
Password	Short Text	The password which the user will login to the program with	
FullName	Short Text	The first and last name of the user. Used to populate combination boxes	
Admin	Yes/No	A boolean value used to check if user is an admin to enable specific controls	

Business Expense Applications Table (BEA)

BEA			
	Field Name	Data Type	Description (Optional)
TrackingNumber	Short Text	Used for the tracking number to uniquely identify each business expense	
FullName	Short Text	The first and last name of user	
Email	Short Text	The email address of the user. Used to send email if application is approved	
ItemDescription	Short Text	Name of the good or service user wants to purchase	
Quantity	Number	The quantity of the good or service user wants to purchase	
UnitPrice	Currency	The price for one unit of the good or service	
TotalPrice	Currency	The total price for the goods or services. Quantity * UnitPrice	
Vendor	Short Text	The vendor who is selling the goods or services	
Reason	Short Text	Reason for purchasing the items specified	
Approved	Yes/No	Boolean used to check if application is approved or not	

Business Expenses Table

Business Expenses			
	Field Name	Data Type	Description (Optional)
TrackingNumber	Short Text	Previously assigned tracking number from application or manually	
FullName	Short Text	The first and last name of the user. Loaded from BEA	
DateIncurred	Date/Time	The date the user purchased the goods or services	
Description	Short Text	Description of the items. Loaded from BEA	
Quantity	Number	Number of items purchased. Loaded from BEA	
UnitPrice	Currency	The price for one unit. Loaded from BEA	
TotalPrice	Currency	The total price for all the units. Loaded from BEA	
Vendor	Short Text	The store or vendor where the items were purchased from. Loaded from BEA	
PaymentMethod	Short Text	The payment method used to purchase the goods (personal, company, other)	
ReimbursementRequired	Yes/No	Boolean to ask user if he/she needs to be reimbursed	
Reimbursed	Yes/No	Boolean to check if the business expense has been reimbursed yet	

Investments Table

Investments			
	Field Name	Data Type	Description (Optional)
ID	AutoNumber	Unique 'ID' to identify each Investment	
Investor	Short Text	The name of the investor	
DateInvested	Date/Time	The date the investment was received	
InjectionAmount	Currency	The amount the investor invested	

Revenues Table

Revenues			
	Field Name	Data Type	Description (Optional)
Y	PONumber	Short Text	The purchase order number of the order. Unique to each order
	InvoiceNumber	Short Text	The invoice number of the order
	InvoiceDate	Date/Time	The date the invoice was issued
	Customer	Short Text	The name of the customer
	Subtotal	Currency	The subtotal before GST is added
	Total	Currency	The total amount received
	GST	Currency	The goods and services tax on the order

Travel Expense Applications Table (TEA)

TEA			
	Field Name	Data Type	Description (Optional)
Y	TrackingNumber	Short Text	The uniquely assigned tracking number
	FullName	Short Text	First name and last name of applicant
	Email	Short Text	Email address of applicant. Used to send email if TEA is approved
	Destinations	Short Text	Destinations the user is planning to go to
	Duration	Number	How many days the user is planning to go for
	Reason	Short Text	The reason for the trip
	AccomodationCosts	Currency	Estimated costs for accomodation
	DomesticTravelCosts	Currency	Estimated costs for domestic travel (i.e. in the country)
	InternationalTravelCosts	Currency	Estimated costs for international travel
	TotalCost	Currency	The total estimated costs
	Approved	Yes/No	Boolean used to check if TEA was approved or not

Travel Expenses Table

Travel Expenses			
	Field Name	Data Type	Description (Optional)
Y	ID	AutoNumber	A unique 'ID' for each expense
	TrackingNumber	Short Text	The tracking number for which the ID relates to
	FullName	Short Text	First and last name of the user
	DateIncurred	Date/Time	The date that the expense was incurred on
	ExpenseType	Short Text	The type of expense
	Description	Short Text	Short description of the expense
	BillClient	Yes/No	Boolean for if the client is to be billed for the expense
	PaymentMethod	Short Text	How the user paid for the expense (personal, company, other)
	ForeignAmount	Currency	The amount for the expense
	ForeignCurrency	Short Text	The currency that the expense was paid with
	ExchangeRate	Currency	The exchange rate at the time of the expense
	NZDAmount	Currency	The NZD amount for the expense. ForeignAmount * ExchangeRate
	Reimburse	Yes/No	Boolean to check if user requires reimbursement or not
	Reimbursed	Yes/No	Boolean to check if user has been reimbursed. Set to Yes if Reimburse is 'No'

Wages Table

Wages			
	Field Name	Data Type	Description (Optional)
Y	ID	AutoNumber	Unique 'ID' for each wage payment
	StartDate	Date/Time	The date that the wage payment begins on
	EndDate	Date/Time	The date that the wage payment ends on
	FullName	Short Text	The first name and last name of who the wages are being paid to
	DatePaid	Date/Time	The date that the wage payment was actually paid
	GrossPay	Currency	The gross pay before any tax deductions
	PAYE	Currency	Amount for PAYE deductions
	KiwiSaver	Currency	Amount for KiwiSaver deductions
	NetPay	Currency	The overall net pay. GrossPay - PAYE - KiwiSaver

Points of Difference

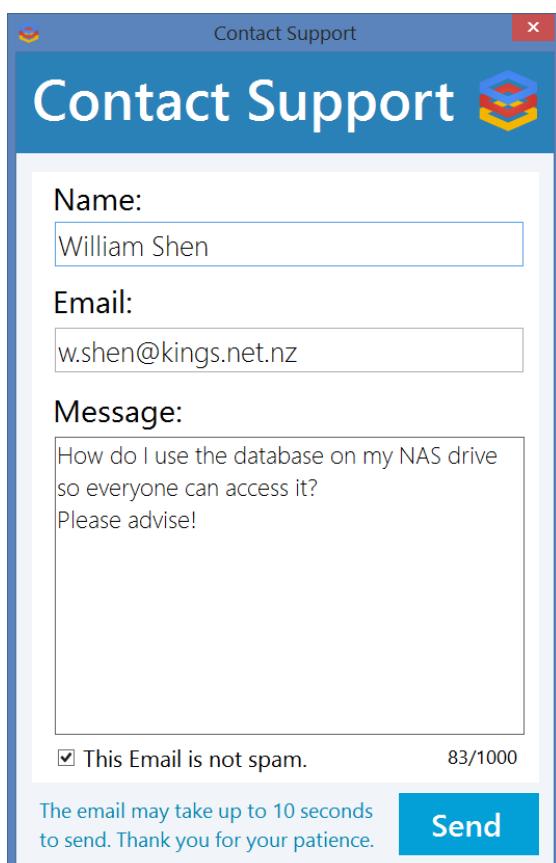
- Travel Expense and Business Expense Table – contains a new field called “Reimbursed” which is for checking if the expense has been reimbursed or not. It is different from “Reimburse” in that “Reimburse” means whether a reimbursement is required or not.
- Business Expense Table – the field “Reimburse” has been renamed to “ReimbursementRequired” for ease of use.
- Revenues Table – much of this table has been redesigned so it fits for the total purchases instead of each individual one. Instead of each individual purchase, the revenues table now stores overall numbers including “Subtotal”, “Total” and “GST”.

Program Form Layouts

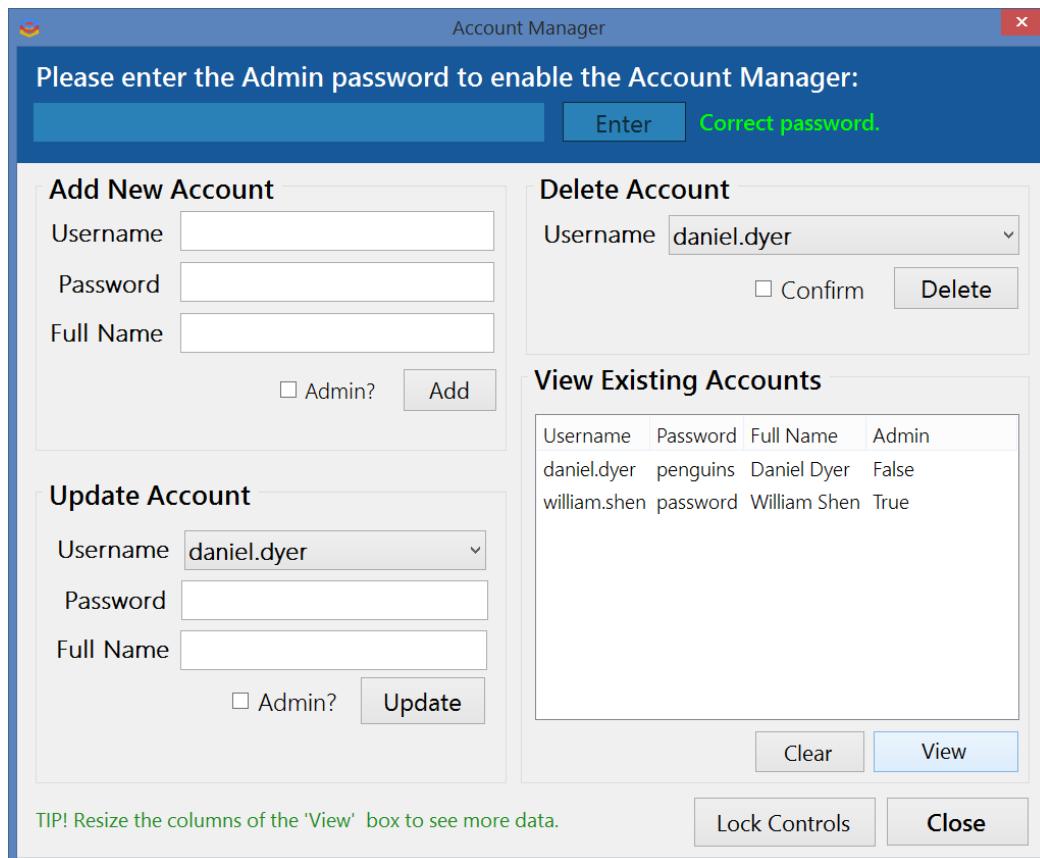
Login Form



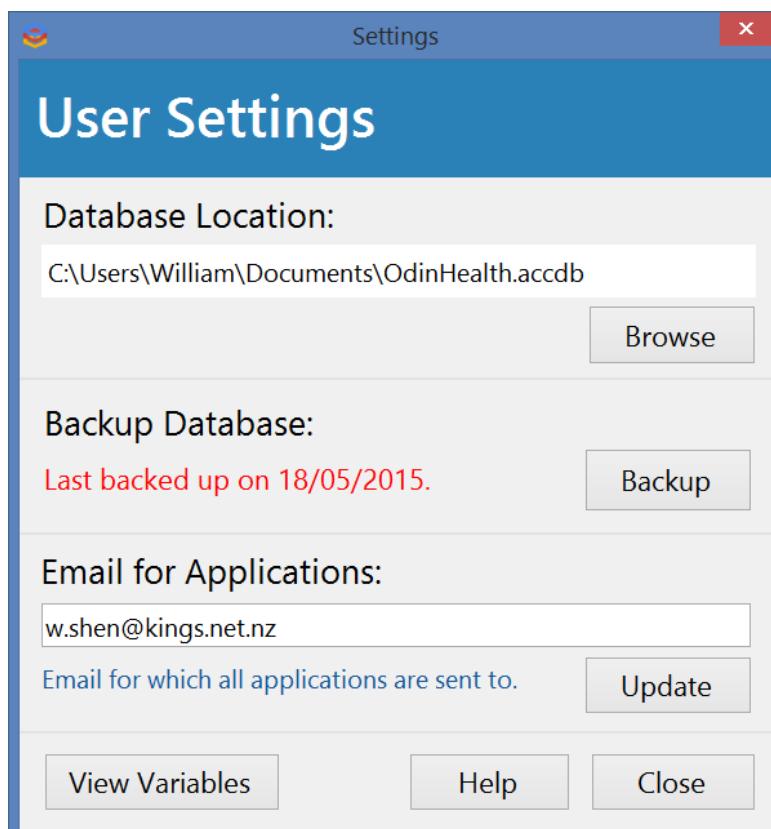
Contact Support



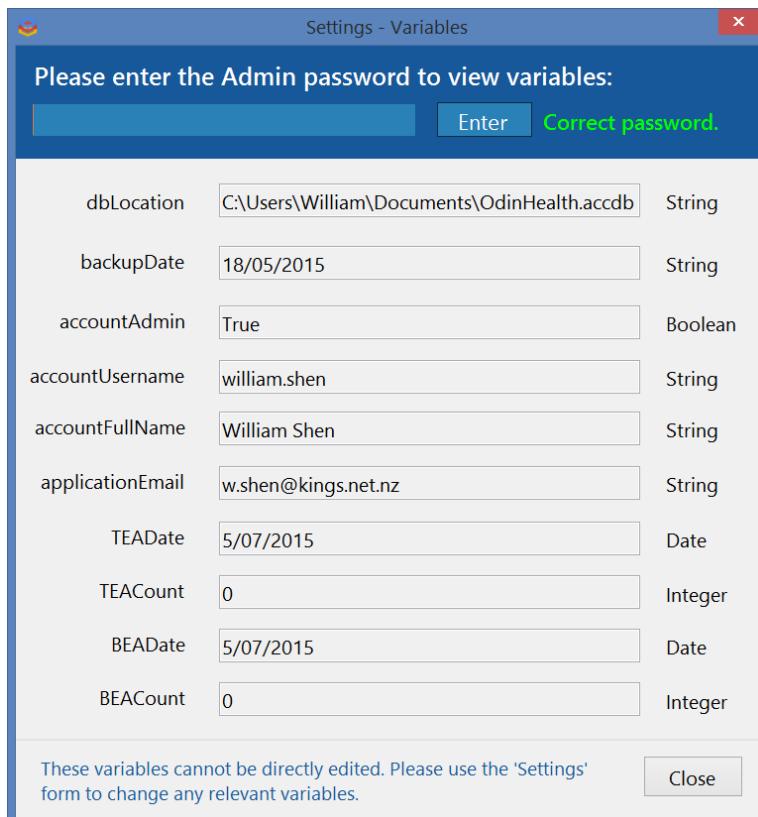
Account Manager



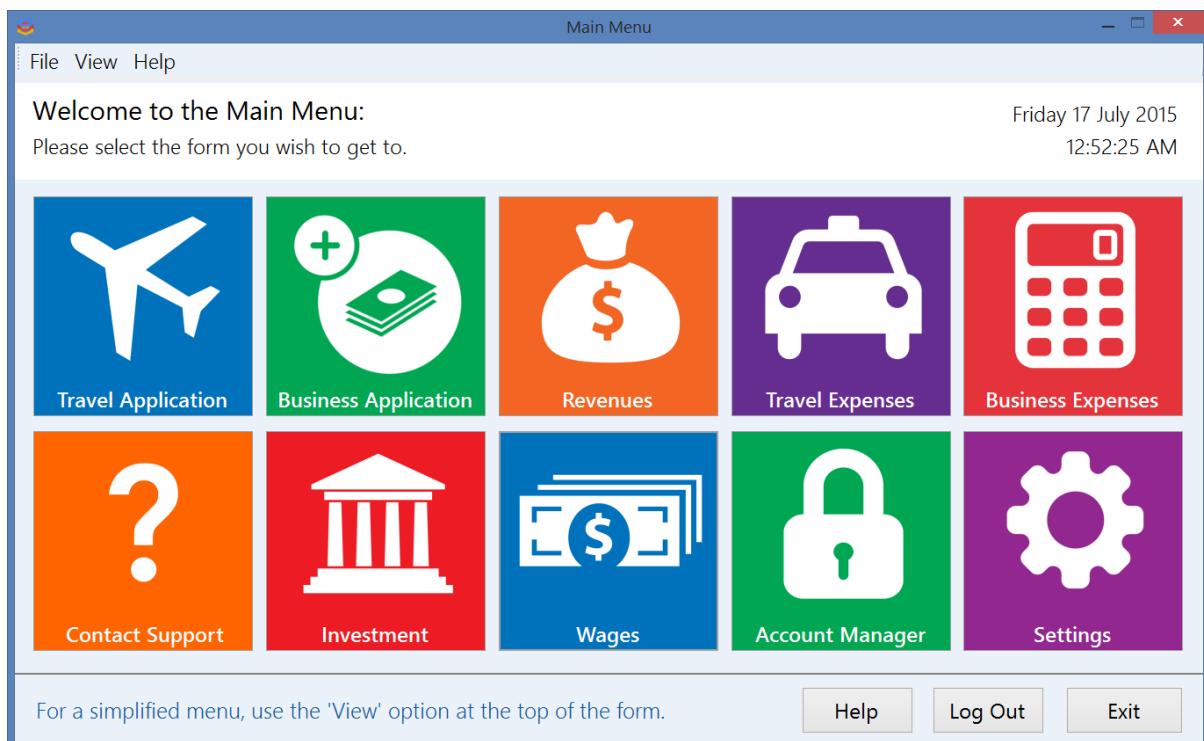
User Settings



Settings Variables



Main Menu



Travel Application – Overview

Travel Application - Overview

Pending Applications (Unapproved)				
Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0607150	Daniel Dyer	London UK	5	5602
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308

Past Applications (Approved)				
Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0706151	William Shen	Australia, USA, India	26	140000

Select a row from the tables above:

[View or Edit](#) [Delete](#) [Search](#) [New Application](#)

Travel Application – New

Travel Application - New

Employee Name	Email Address	Tracking Number		
William Shen	william.shen@odinhealth.co.nz	TV0607151		
Destinations (Countries and Cities)	Duration			
<input type="text"/>	<input type="text"/> Days			
Reason for Trip <input type="text"/>				
Please estimate the following costs:				
Domestic Travel Costs	<input type="text"/> NZD	Accommodation Costs	<input type="text"/> NZD	
International Travel Costs	<input type="text"/> NZD	Total Costs	<input type="text"/> NZD	
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.			Close	Submit

Travel Application – View and Edit

Travel Application - View and Edit

Approved?

Employee Name William Shen	Email Address w.shen@kings.net.nz	Tracking Number TV0706151
Destinations (Countries and Cities) Australia, USA, India		Duration 26 Days
Reason for Trip Learn about the different cultures in other countries. Make new business deals		
Please estimate the following costs:		
Domestic Travel Costs 10000 NZD	Accommodation Costs 10000 NZD	
International Travel Costs 120000 NZD	Total Costs 140000 NZD	
Update the relevant data and press 'Update'. Close Delete Update		

Business Expense Application – Overview

Business Expense Applications - Overview

Pending Applications (Unapproved)					
Tracking No.	Full Name	Description	Quantity	Total Price	
BS2105150	William Shen	LED Lamps	20	460	
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552	
Past Applications (Approved)					
Tracking No.	Full Name	Description	Quantity	Total Price	
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250	
BS1106151	William Shen	Dell Projector	1	1200	
Select a row from the tables above:					
View or Edit	Delete	Search	New Application		

Business Expense Application – New

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS0607151
Item Description	Quantity	
Unit Price	Total Price	Vendor
NZD	NZD	
Reason for Purchase		
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.		
<input type="button" value="Close"/>		<input type="button" value="Submit"/>

Business Expense Application – View and Edit

Business Expense Application - View and Edit

<input checked="" type="checkbox"/> Approved?		
Employee Name	Email Address	Tracking Number
Daniel Dyer	w.shen@kings.net.nz	BS0607150
Item Description	Quantity	
Nexus 9 64GB		5
Unit Price	Total Price	Vendor
850 NZD	4250 NZD	Google
Reason for Purchase		
To test the platform.		
Update the relevant data and press 'Update'.		
<input type="button" value="Close"/>		<input type="button" value="Delete"/>
		<input type="button" value="Update"/>

Travel Expenses – Overview

Travel Expense - Overview

Tracking No.	Name	Period	Amount (NZD)	Reimbursed
TV0706150	John Shen	29/06/2015 - 1/07/2015	5952.88	True
TV0706151	William Shen	2/07/2015 - 5/07/2015	3049.92	True

[View or Edit](#) [Delete](#) [Print](#) [Search](#) [New Expense](#)

Travel Expenses – New, View and Edit

Travel Expense

Travel Expense - New

Tracking Number	Employee Name	
TV0706151	William Shen	
Date Incurred	Expense Type	Payment Method
6/07/2015		
Item Description	<input type="checkbox"/> Reimburse? <input type="checkbox"/> Bill Client?	
Foreign Amount	Exchange Rate	NZD Amount
	RMB to NZD	0 NZD
Please enter the Tracking Number.		Clear Save

[View or Edit](#) [Delete](#) [Print](#) [New Expense](#)

ID	Date	Description	Type	NZD	Reimbursed
35	2/07/2015	Sheraton in Shanghai f...	Accomm...	2880	True
37	5/07/2015	Movie Tickets to watch ...	Entertai...	6	True
39	5/07/2015	High Speed rail to Han...	Public Tr...	56.4	True
40	5/07/2015	Hotel internet	Internet	78	True
41	5/07/2015	Cleaning suit	Dry Clea...	29.52	True

Travel Expense

Travel Expense - View and Edit

Reimbursed?

Tracking Number	Employee Name																																					
TV0706151	William Shen																																					
Date Incurred	Expense Type	Payment Method																																				
2/07/2015	Accommodation	Personal																																				
Item Description		<input checked="" type="checkbox"/> Reimburse?																																				
Sheraton in Shanghai for 12 nights		<input type="checkbox"/> Bill Client?																																				
Foreign Amount	Exchange Rate	NZD Amount																																				
12000 RMB	0.24	2880 NZD																																				
Record successfully loaded. Please make any desired changes and click 'Update'.																																						
<input type="button" value="Delete"/> <input type="button" value="Update"/>																																						
<table border="1"> <thead> <tr> <th>ID</th> <th>Date</th> <th>Description</th> <th>Type</th> <th>NZD</th> <th>Reimbursed</th> </tr> </thead> <tbody> <tr> <td>35</td> <td>2/07/2015</td> <td>Sheraton in Shanghai f...</td> <td>Accomm...</td> <td>2880</td> <td>True</td> </tr> <tr> <td>37</td> <td>5/07/2015</td> <td>Movie Tickets to watch ...</td> <td>Entertai...</td> <td>6</td> <td>True</td> </tr> <tr> <td>39</td> <td>5/07/2015</td> <td>High Speed rail to Han...</td> <td>Public Tr...</td> <td>56.4</td> <td>True</td> </tr> <tr> <td>40</td> <td>5/07/2015</td> <td>Hotel internet</td> <td>Internet</td> <td>78</td> <td>True</td> </tr> <tr> <td>41</td> <td>5/07/2015</td> <td>Cleaning suit</td> <td>Dry Clea...</td> <td>29.52</td> <td>True</td> </tr> </tbody> </table>			ID	Date	Description	Type	NZD	Reimbursed	35	2/07/2015	Sheraton in Shanghai f...	Accomm...	2880	True	37	5/07/2015	Movie Tickets to watch ...	Entertai...	6	True	39	5/07/2015	High Speed rail to Han...	Public Tr...	56.4	True	40	5/07/2015	Hotel internet	Internet	78	True	41	5/07/2015	Cleaning suit	Dry Clea...	29.52	True
ID	Date	Description	Type	NZD	Reimbursed																																	
35	2/07/2015	Sheraton in Shanghai f...	Accomm...	2880	True																																	
37	5/07/2015	Movie Tickets to watch ...	Entertai...	6	True																																	
39	5/07/2015	High Speed rail to Han...	Public Tr...	56.4	True																																	
40	5/07/2015	Hotel internet	Internet	78	True																																	
41	5/07/2015	Cleaning suit	Dry Clea...	29.52	True																																	
<input type="button" value="View or Edit"/> <input type="button" value="Delete"/> <input type="button" value="All Reimbursed"/> <input type="button" value="Print"/> <input type="button" value="New Expense"/>																																						

Business Expenses – Overview

Business Expense - Overview

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS1106150	William Shen	27/06/2015	Ubiquiti Unifi AP	53136
BS2105150	William Shen	6/07/2015	LED Lamps	460

Select a row from the tables above:

Print by:

Expenses Between and Completed Reimbursement
 Employee Name Pending Reimbursement
 Print all data Find Total

Business Expenses – New

Business Expense - New

Tracking Number	Employee Name	
BS1106151	William Shen	
Date Purchased	Vendor	Payment Method
6/07/2015	Dell.co.nz	
Item Description	Quantity	
Dell Projector	1	
Unit Price	Total Price	Reimbursement required?
1200 NZD	1200 NZD	<input type="checkbox"/>

Enter the Tracking Number and all relevant data will 'appear'.

Business Expenses – View and Edit

Business Expense - View and Edit

Tracking Number	Employee Name	<input checked="" type="checkbox"/> Reimbursed?
BS2105150	William Shen	
Date Purchased	Vendor	Payment Method
6/07/2015	PB Tech	Personal
Item Description	Quantity	
LED Lamps	20	
Unit Price	Total Price	Reimbursement required?
23 NZD	460 NZD	<input checked="" type="checkbox"/>

Update the relevant data and press 'Update'.

Revenues – Overview

Revenues - Overview

Revenue Records				
PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
234234	420Dustll	21/06/2015	Microsoft	23593.44
24356	1234	15/07/2015	56556	246

Select a row from the tables above:

Filter by:

Payments Between and

Revenues – New

Revenues - New

PO Number	Invoice Number	Invoice Date
<input type="text"/>	<input type="text"/>	<input type="text" value="6/07/2015"/>

Customer (Purchaser)

Subtotal	GST	Total
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/> NZD

Fill in all the relevant fields and then click 'Add'.

Revenues – View and Edit

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
15421354	20151001	10/01/2015 <input type="button" value="▼"/>
Customer (Purchaser)		
ShenZhen Government		
Subtotal	GST	Total
799235.26 NZD	0.0000 NZD	799235.26 NZD
Update in all relevant fields and then click 'Update'.		
<input type="button" value="Close"/> <input type="button" value="Delete"/> <input type="button" value="Update"/>		

Investment

Investment

ID	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421

Investor

Date Amount
6/07/2015 NZD

Enter all relevant data into the controls and press "Add".

Investment – Edit

Investment - Edit

Investment - Edit

ID	Investor
27	Shanghai Medical Investors
Date	Amount
3/03/2015	34843 NZD

Update all relevant data and press "Update".

Wages – Overview

Wages - Overview

Wages - Overview

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	68999090
22	21/06/2015	21/06/2015	William Shen	21/06/2015	9877
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34

Select a row from the tables above:

Record successfully updated.

Filter by:

Payments Between and

Employee Name

Printing Options:

Print all Data
 Print Filtered Data

Wages – New

Wages - New

Period for Payment		Date Paid			
From	6/07/2015	To	6/07/2015	Date Paid	6/07/2015
Employee Name			Gross Pay		
William Shen			NZD		
PAYE	KiwiSaver	Net Pay			
Fill in the relevant details and press "Add".					
			Close	Add	

Wages – View and Edit

Wages - View and Edit

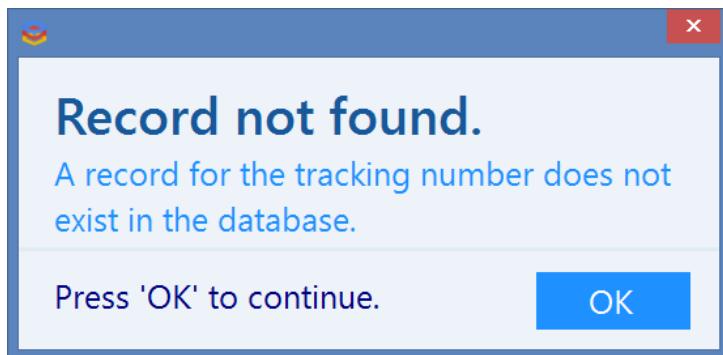
Period for Payment		Date Paid	ID				
From	21/06/2015	To	21/06/2015	Date Paid	21/06/2015	ID	22
Employee Name			Gross Pay				
William Shen			12344 NZD				
PAYE	KiwiSaver	Net Pay					
1234 NZD	1233 NZD	9877 NZD					
Press "Update" to update record.				Close	Delete	Update	

Search Box Form

Please enter the Tracking Number.

<input type="text"/>	
Cancel	Search

Status Box Form



About Form



Points of Difference

- Settings Form – a new part has been added for application emails.
- Settings Variables Form – a new form used to view the variables stored by the system for various functions (e.g. assigning tracking numbers).
- Close Buttons on “New” forms for ease of navigation.
- Business Expenses Form – a new segment has been added to allow for the filtering of data.
- Revenues – the form and data structures have been redesigned to account for an overview cost (subtotal, GST, total) instead of individual items.
- Investment Edit Form – a new form used to edit the investment records.
- Wages Overview Form – the print section has been moved and gives more options now.
- Search Box Form – a new form used to accept data used for searching records on the database.
- Status Box Form – a new form used to show output for the search box form.
- About Form – a new form used to show information about the software build.

Report Layouts (for Printing)

Business Expenses

Page 1

Business Expenses

Tracking No.	Name	Date	Description & Qty	Vendor	Total	Reimbursed
BS1106150	William Shen	27/06/15	Ubiquiti Unifi AP x123	PB Tech	\$53136	True
BS11061502	Daniel Dyer	28/06/15	Nexus 9 64GB x6	joiij	\$4680	False
BS2105150	William Shen	06/07/15	LED Lamps x20	PB Tech	\$460	True
				TOTAL =	\$58276	

Travel Expenses

Page 1

TV0706150 - John Shen

Date	Description	BC	Type	Amount	XRT	NZD	Repaid
28/06/15	NZ to HK (Air NZ)		Airfares	NZD \$3500	1	\$3500	Yes
29/06/15	Buffet dinner x5		Meals	RMB \$2000	0.23	\$460	No
29/06/15	Uber from Airport to Hotel		Taxi	RMB \$320	0.243	\$77.76	No
29/06/15	Shanghai Hotel for 12 nights		Accommodation	RMB \$6530	0.235	\$1534.55	No
29/06/15	Hotel Internet		Internet	RMB \$100	0.231	\$23.1	Yes
02/07/15	Shanghai to Hong Kong (Air China)		Airfares	RMB \$1200	0.231	\$277.2	No
04/07/15	Train to Hangzhou		Public Transport	RMB \$332	0.23	\$76.36	Yes
05/07/15	Cleaning business suit		Dry Cleaning	RMB \$500	0.24	\$120	No
07/07/15	Lunch for 2 weeks		Meals	RMB \$1563	0.24	\$375.12	No
07/07/15	Movie Tickets		Entertainment	RMB \$1220	0.24	\$292.8	No
08/07/15	Shanghai to Auckland		Airfares	NZD \$2100	1	\$2100	No
08/07/15	Hangzhou to Shanghai		Airfares	RMB \$900	0.24	\$216	Yes
09/07/15	Auckland Airport Taxi to Home		Taxi	NZD \$67	1	\$67	No
						\$9119.89	\$5304.43

Wages

Page 1

Wages - All

Dates	Name	Date Paid	Gross Pay	PAYE	KiwiSaver	Net Pay
14/01/15 - 19/06/15	William Shen	21/05/15	\$6000	\$5	\$763	\$5232
28/05/15 - 28/05/15	Max Mcguire	28/05/15	\$69000000	\$455	\$455	\$68999090
21/06/15 - 21/06/15	William Shen	21/06/15	\$12344	\$1234	\$1233	\$9877
10/06/15 - 28/06/15	John Shen	26/06/15	\$4535	\$65.33	\$0.33	\$4469.34
08/07/15 - 09/07/15	Daniel Dyer	06/07/15	\$562100	\$56223	\$123456	\$382421

Some changes have been made to these report layouts such as extra fields "BC = Bill Client" and addition of totals at the bottom of the report.

Programming

Problems Encountered

There were a few problems encountered during the programming of the solution. However, all these problems have been solved and are now fully working in the current build of the software.

Encountered problems included:

Problem	Solution
Sorting ListViews when their respective columns are clicked	Using the Microsoft Developers website to draw inspiration and study the properties behind the ListView. Furthermore, I did lots of research and looked at the code of other programmers to figure a working solution for the Odin Business Manager.
Filtering data and displaying it on the ListViews	Planning and designing logical queries so that the correct data is returned and then adding the data to the ListView one by one.
Printing data in a clear form by interacting with the database	Through research, I discovered that using a Data Grid View and then printing the data off this was the easiest and most efficient way to print the contents of a database. I then used a dynamic link library designed for printing Data Grid Views and passed the data as appropriate.

Listed in Alphabetical Order

About.vb

```
Public Class About
    Private Sub lblSupport_Click(sender As Object, e As EventArgs) Handles lblSupport.Click
        'Shows the form to contact support
        Contact.Show()
    End Sub

    Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
        'Closes this form
        Me.Close()
    End Sub
End Class
```

AccountManager.vb

```
Imports System.Data.OleDb

Public Class AccountManager
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

    Private Sub AccountManager_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Adds the columns and sizes for the listview box
        lstView.Columns.Add("Username", 130, HorizontalAlignment.Left)
        lstView.Columns.Add("Password", 130, HorizontalAlignment.Left)
        lstView.Columns.Add("Full Name", 130, HorizontalAlignment.Left)
        lstView.Columns.Add("Admin", 70, HorizontalAlignment.Left)
        lstView.View = View.Details
        'Runs subroutine to populate the combination boxes
        populateCBO()
        'Focuses form on the enable textbox
        txtEnable.Select()
    End Sub

    Private Sub btnEnable_Click(sender As Object, e As EventArgs) Handles btnEnable.Click
        'Checks if password is correct or not
        If txtEnable.Text = "password" Then
            'Disable Enable controls
            btnEnable.Enabled = False
            txtEnable.Enabled = False
            'Runs subroutine to enable controls
        End If
    End Sub
```

```

controlsEnable()
'Feedback for user to show success
lblAdmin.Text = "Correct password."
lblAdmin.ForeColor = Color.Lime
lblAdmin.Visible = True
txtEnable.Text = ""
Else
    'Feedback for incorrect password
    lblAdmin.Text = "Incorrect password."
    lblAdmin.ForeColor = Color.Red
    lblAdmin.Visible = True
    'Resets textbox contents
    txtEnable.Text = ""
End If
End Sub

Private Sub txtEnable_KeyDown(ByVal sender As Object, ByVal e As KeyEventArgs) Handles txtEnable.KeyDown
    'Calls subroutine of the enable button when enter key is pressed in the Password textbox
    If e.KeyCode = Keys.Enter Then
        Call btnEnable_Click(sender, e)
    End If
End Sub

Private Sub btnAddAccount_Click(sender As Object, e As EventArgs) Handles btnAddAccount.Click
    'Sets the variable which will be used in the Insert query to add the 'Admin' Yes/No field.
    Dim Admin As Integer
    If chkAdminAdd.Checked = True Then
        Admin = 1
    Else : Admin = 0
    End If
    'Converts Username into lowercase
    Dim Username As String = txtUsernameAdd.Text
    Username = Username.ToLower()
    'If checkbox is checked and textboxes contain text (validation)
    If txtUsernameAdd.Text <> "" AndAlso txtPasswordAdd.Text <> "" AndAlso txtNameAdd.Text <> "" Then
        Try
            'Declares OleDbCommand as gives the parameters and relevant fields to insert data from textboxes into
            Dim cmd As New OleDbCommand("INSERT INTO Accounts ([Username],[Password],[FullName],[Admin])" + "VALUES ('" & Username &
                "','" & txtPasswordAdd.Text & "','" & txtNameAdd.Text & "','" & Admin & "')", connection)
            'Opens connection, executes the OleDbCommand and closes the connection
            connection.Open()
            cmd.ExecuteNonQuery()
            connection.Close()
            'Feedback for user to show success
            lblAddConfirm.Text = "Account successfully added."
        End Try
    End If
End Sub

```

```

lblAddConfirm.ForeColor = Color.Green
lblAddConfirm.Visible = True
lstView.Items.Clear()
'Clears textboxes and checkbox and runs subroutine to populate combobox
txtUsernameAdd.Text = ""
txtPasswordAdd.Text = ""
txtNameAdd.Text = ""
'Resets checkbox and populates the combobox again
chkAdminAdd.Checked = False
populateCBO()
Catch ex As Exception
'Closes the connection
connection.Close()
'Output to user to show an error occurred
MsgBox("Something went wrong." & vbCrLf & "The Username you entered most probably already exists in the database.")
End Try
Else
'Feedback to show checkbox is not checked or textboxes are empty
lblAddConfirm.Text = "Please confirm the details."
lblAddConfirm.ForeColor = Color.Red
lblAddConfirm.Visible = True
End If
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
'Verify that the user wants to delete the account
If chkDelete.Checked = True Then
'Opens database connection
connection.Open()
'Declares a new OleDbCommand which deletes the record which contains the text from the 'Delete' combobox
Dim cmd As New OleDbCommand("DELETE FROM Accounts WHERE Username ='" & cboDelete.Text & "'", connection)
'Executes the command and closes the connection
cmd.ExecuteNonQuery()
connection.Close()
'Repopulates the comboboxes
populateCBO()
'Feedback for user to show success and resets relevant controls
lblDelete.Text = "Account has been deleted."
lblDelete.ForeColor = Color.Green
lblDelete.Visible = True
'Resets relevant parts of the form
chkDelete.Checked = False
lstView.Items.Clear()
Else
'Feedback to show checkbox is not checked

```

```

lblDelete.Text = "Please confirm your choice."
lblDelete.ForeColor = Color.Red
lblDelete.Visible = True
End If
End Sub

Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
    'Checks if the textboxes all contain text (validation)
    If txtPasswordUpdate.Text <> "" And txtNameUpdate.Text <> "" Then
        'Declares a new OleDbCommand which deletes the record which contains the text from the 'Delete' combobox
        Dim cmdDelete As New OleDbCommand("DELETE FROM Accounts WHERE Username ='" & cboUsernameUpdate.Text & "'", connection)
        'Opens connection, executes the command and closes the connection
        connection.Open()
        cmdDelete.ExecuteNonQuery()
        connection.Close()
        'Sets the variable which will be used in the Insert query to update the 'Admin' Yes/No field.
        Dim Admin As Integer
        If chkAdminUpdate.Checked = True Then
            Admin = 1
        Else : Admin = 0
        End If
        'Declares OleDbCommand as gives the parameters and relevant fields to insert data from textboxes into
        Dim cmdAdd As New OleDbCommand("INSERT INTO Accounts ([Username],[Password],[FullName],[Admin])" +
                                         "VALUES ('" & cboUsernameUpdate.Text & "','" & txtPasswordUpdate.Text & _
                                         "','" & txtNameUpdate.Text & "','" & Admin & "')", connection)
        'Opens connection, executes the command and closes the connection
        connection.Open()
        cmdAdd.ExecuteNonQuery()
        connection.Close()
        'Repopulates the comboboxes
        populateCBO()
        'Feedback for user to show success and resets relevant controls
        lblUpdate.Text = "Account has been updated."
        lblUpdate.ForeColor = Color.Green
        lblUpdate.Visible = True
        'Resets relevant parts of the form
        txtPasswordUpdate.Text = ""
        txtNameUpdate.Text = ""
        chkAdminUpdate.Checked = False
        lstView.Items.Clear()
    Else
        'Feedback to show validation was not successful
        lblUpdate.Text = "Please confirm the details."
        lblUpdate.ForeColor = Color.Red
        lblUpdate.Visible = True
    End If
End Sub

```

```

End If
End Sub

Private Sub btnView_Click(sender As Object, e As EventArgs) Handles btnView.Click
    'Enables the listview and clears any existing items
    lstView.Enabled = True
    lstView.Items.Clear()
    'Opens the connection and declares a new OleDbCommand and DataReader
    connection.Open()
    Dim cmd As New OleDbCommand("SELECT * From Accounts ORDER BY Username", connection)
    Dim dr As OleDbDataReader = cmd.ExecuteReader
    'Loops the Accounts table and adds each record to a new ListViewItem then displays it
    Do While dr.Read()
        Dim newitem As New ListViewItem(dr.Item("Username").ToString)
        newitem.SubItems.Add(dr.Item("Password").ToString)
        newitem.SubItems.Add(dr.Item("FullName").ToString)
        newitem.SubItems.Add(dr.Item("Admin"))
        lstView.Items.Add(newitem)
    Loop
    'Closes the connection
    connection.Close()
    'Resizes the ListView to fit the data.
    Dim width As Integer
    For i As Integer = 0 To lstView.Columns.Count - 1
        lstView.Columns(i).Width = -2
        width += lstView.Columns(i).Width
    Next i
End Sub

Private Sub btnViewClear_Click(sender As Object, e As EventArgs) Handles btnViewClear.Click
    'Clears the contents of the list view
    lstView.Enabled = False
    lstView.Items.Clear()
End Sub

Private Sub btnLock_Click(sender As Object, e As EventArgs) Handles btnLock.Click
    'Runs subroutine to disable the controls on the form
    controlsDisable()
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Closes the form completely
    Me.Dispose()
End Sub

```

```

Private Sub populateCBO()
    Try
        'Declares DataSet and fills it up with data from Username Column in database
        Dim ds As New DataSet()
        connection.Open()
        Using da As New OleDbDataAdapter("SELECT Username FROM Accounts", connection)
            da.Fill(ds, "Username")
        End Using
        connection.Close()
        'Loads DataSet data into a DataView, sorts it then writes it to a DataTable
        Dim view As New DataView(ds.Tables("Username"))
        view.Sort = "Username"
        Dim sorted As DataTable = view.ToTable()
        'Populates comboboxes with contents of the new Datatable
        With cboUsernameUpdate
            .DisplayMember = "Username"
            .DataSource = view
            .SelectedIndex = 0
        End With
        With cboDelete
            .DisplayMember = "Username"
            .DataSource = view
            .SelectedIndex = 0
        End With
    Catch ex As Exception
        'Runs the following code if an error occurs (for an empty table)
        cboUsernameUpdate.DataSource = Nothing
        cboDelete.DataSource = Nothing
        MsgBox("The table 'Accounts' is empty")
    End Try
End Sub

Private Sub AccountManager_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'When the form is closed, it is completely closed
    Me.Dispose()
End Sub

Private Sub controlsDisable()
    'Controls to disable and clear account management editing
    grpNew.Enabled = False
    grpDelete.Enabled = False
    grpChange.Enabled = False
    grpView.Enabled = False
    btnLock.Enabled = False
    btnView.Enabled = False

```

```

txtEnable.Enabled = True
btnEnable.Enabled = True
lblAdmin.Text = ""
txtEnable.Focus()
lstView.Items.Clear()
End Sub

Private Sub controlsEnable()
    'Controls to enable account management editing
    grpNew.Enabled = True
    grpDelete.Enabled = True
    grpChange.Enabled = True
    grpView.Enabled = True
    btnLock.Enabled = True
    btnView.Enabled = True
End Sub
End Class

```

Business Expense - Overview.vb

```

Imports System.Data.OleDb
Imports Odin.ListViewSort
Imports System.Drawing.Printing

Public Class Business_Expense__Overview
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Dim dgvPrinter As DataGridViewPrinter
    Private m_SortingColumn As ColumnHeader

    Private Sub Business_Expense__Overview_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Runs subroutines to populate listviews and combobox
        populateListViewPending()
        populateListViewCompleted()
        populateCBO()
        'Sets minimum value of the DateTo datepicker to the value of the DateFrom datepicker
        dtpDateTo.MinDate = dtpDateFrom.Value.Date
        'Sets font on the datagridview for printing
        DataGridView.Font = New Font("Segoe UI", 12)
        'Focuses on form
        PictureBox.Select()
    End Sub

    Public Sub populateListViewPending()
        'Clears and enables listview

```

```

lstViewPending.Items.Clear()
lstViewPending.Enabled = True
'Opens the connection and declares a new OleDbCommand and Reader
connection.Open()
Dim cmd As New OleDbCommand("SELECT * From [Business Expenses] WHERE Reimbursed = False ORDER BY TrackingNumber", connection)
Dim dr As OleDbDataReader = cmd.ExecuteReader
'Loops the Business Expenses table of the database and adds each record to a new ListViewItem
Do While dr.Read()
    Dim newitem As New ListViewItem(dr.Item("TrackingNumber").ToString)
    newitem.SubItems.Add(dr.Item("FullName").ToString)
    newitem.SubItems.Add(dr.Item("DateIncurred"))
    newitem.SubItems.Add(dr.Item("Description").ToString)
    newitem.SubItems.Add(dr.Item("TotalPrice"))
    lstViewPending.Items.Add(newitem)
Loop
'Closes the connection
connection.Close()
End Sub

Public Sub populateListViewCompleted()
'Clears and enables listview
lstViewCompleted.Items.Clear()
lstViewCompleted.Enabled = True
'Opens the connection and declares a new OleDbCommand and Reader
connection.Open()
Dim cmd As New OleDbCommand("SELECT * From [Business Expenses] WHERE Reimbursed = True ORDER BY TrackingNumber", connection)
Dim dr As OleDbDataReader = cmd.ExecuteReader
'Loops the Business Expenses table of the database and adds each record to a new ListViewItem
Do While dr.Read()
    Dim newitem As New ListViewItem(dr.Item("TrackingNumber").ToString)
    newitem.SubItems.Add(dr.Item("FullName").ToString)
    newitem.SubItems.Add(dr.Item("DateIncurred"))
    newitem.SubItems.Add(dr.Item("Description").ToString)
    newitem.SubItems.Add(dr.Item("TotalPrice"))
    lstViewCompleted.Items.Add(newitem)
Loop
'Closes the connection
connection.Close()
End Sub

Private Sub populateCBO()
'Declares DataSet and fills it up with data from Username Column in database
Dim ds As New DataSet()
Using da As New OleDbDataAdapter("SELECT FullName FROM Accounts", connection)
    da.Fill(ds, "FullName")

```

```

End Using
'Loads DataSet data into a DataView, sorts it then writes it to a DataTable
Dim view As New DataView(ds.Tables("FullName"))
view.Sort = "FullName"
Dim sorted As DataTable = view.ToTable()
'Populates comboboxes with contents of the new Datatable
With cboName
    .DisplayMember = "FullName"
    .DataSource = view
    .SelectedIndex = 0
End With
End Sub

Private Sub btnEdit_Click(sender As Object, e As EventArgs) Handles btnEdit.Click
    'Checks if an item is selected in the listview or not
    If lstViewPending.SelectedItems.Count > 0 Then
        'Passes value of ID to Edit form and opens it
        Dim TrackingNumber As String = lstViewPending.SelectedItems(0).Text
        Business_Expense__View_and_Edit.txtTrackingNumber.Text = TrackingNumber
        Business_Expense__View_and_Edit.ShowDialog()
    ElseIf lstViewCompleted.SelectedItems.Count > 0 Then
        'Passes value of ID to Edit form and opens it
        Dim TrackingNumber As String = lstViewCompleted.SelectedItems(0).Text
        Business_Expense__View_and_Edit.txtTrackingNumber.Text = TrackingNumber
        Business_Expense__View_and_Edit.ShowDialog()
        'Output to user if nothing has been selected
    Else : MsgBox("Please select a record from the table.")
    End If
End Sub

Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles btnSearch.Click
    'Opens search box form and passes relevant function name
    SearchBox.lblFunction.Text = "BusinessExpense"
    SearchBox.ShowDialog()
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles btnNew.Click
    'Opens the form for a new business expense
    Business_Expense__New.ShowDialog()
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Checks if either listviews has a line selected or not
    If lstViewPending.SelectedItems.Count > 0 Or lstViewCompleted.SelectedItems.Count > 0 Then
        'Displays a MsgBox asking user to confirm to delete the record
    End If
End Sub

```

```

Dim Response = MsgBox("Are you sure you want to delete this Business Expense Record?", MsgBoxStyle.YesNo,
                      "Delete Business Expense Record")
If Response = MsgBoxResult.Yes Then
    'Checks if an item is selected in the listview or not
    If lstViewPending.SelectedItems.Count > 0 Then
        'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
        Dim pending As String = lstViewPending.SelectedItems(0).Text
        Dim cmd As New OleDbCommand("DELETE FROM [Business Expenses] WHERE TrackingNumber = '" & pending & "'", connection)
        'Opens connection, executes the command and closes the connection
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Repopulates the listview
        populateListViewPending()
        'Output to show successful deletion
        lblDataStatus.Text = "Record successfully deleted."
        lblDataStatus.ForeColor = Color.Green
        lblDataStatus.Visible = True
    ElseIf lstViewCompleted.SelectedItems.Count > 0 Then
        'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
        Dim completed As String = lstViewCompleted.SelectedItems(0).Text
        Dim cmd As New OleDbCommand("DELETE FROM [Business Expenses] WHERE TrackingNumber = '" & completed & "'", connection)
        'Opens connection, executes the command and closes the connection
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Repopulates the listview
        populateListViewCompleted()
        'Output to show successful deletion
        lblDataStatus.Text = "Record successfully deleted."
        lblDataStatus.ForeColor = Color.Green
        lblDataStatus.Visible = True
    End If
End If
'Output if user has not selected a line from either of the listviews
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnPrint_Click(sender As Object, e As EventArgs) Handles btnPrint.Click
    If chkFilterDate.Checked = True Or chkFilterName.Checked = True Or chkCompleted.Checked = True Or
       chkPending.Checked = True Or chkAll.Checked = True Then
        'Declares cmd as OleDbCommand, x as a counter and if a checkbox is checked as a relevant 2^i value
        Dim cmd As OleDbCommand
        'Declares strings to be used in OleDbCommand

```

```

Dim query As String = "SELECT TrackingNumber, FullName, Format(DateIncurred, 'dd/mm/yy'), Description & ' x' & Quantity, Vendor, " &
    "'$' & TotalPrice AS Total, Reimbursed From [Business Expenses] WHERE "
Dim dateQuery As String = "(DateIncurred BETWEEN @DateFrom AND @DateTo) "
Dim name As String = "FullName = @FullName "
Dim reimbursed As String = "Reimbursed = @Reimbursed "
Dim order As String = "ORDER BY DateIncurred"
Dim x As Integer = 0
'Used for the different combinations
If chkFilterDate.Checked = True Then x = 1
If chkFilterName.Checked = True Then x = x + 2
If chkCompleted.Checked = True Then x = x + 4
If chkPending.Checked = True Then x = x + 8
If chkAll.Checked = True Then x = 16
>Selects the different cases. 15 possibilities + 1 possibility for chkAll
Select Case x
    'Declares relevant OleDbCommands and gives them the parameters
    Case 1 'Date only
        cmd = New OleDbCommand(query & dateQuery & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
    Case 2 'Name only
        cmd = New OleDbCommand(query & name & order, connection)
        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
    Case 3 'Date and Name
        cmd = New OleDbCommand(query & dateQuery & "AND " & name & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
    Case 4 'Completed only
        cmd = New OleDbCommand(query & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@Reimbursed", True)
    Case 5 'Date and Completed
        cmd = New OleDbCommand(query & dateQuery & "AND " & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
        cmd.Parameters.AddWithValue("@Reimbursed", True)
    Case 6 'Name and Completed
        cmd = New OleDbCommand(query & name & "AND " & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
        cmd.Parameters.AddWithValue("@Reimbursed", True)
    Case 7 'Date, Name and Completed
        cmd = New OleDbCommand(query & dateQuery & "AND " & name & "AND " & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)

```

```

        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
        cmd.Parameters.AddWithValue("@Reimbursed", True)
    Case 8 'Pending only
        cmd = New OleDbCommand(query & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@Reimbursed", False)
    Case 9 'Date and Pending
        cmd = New OleDbCommand(query & dateQuery & "AND " & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
        cmd.Parameters.AddWithValue("@Reimbursed", False)
    Case 10 'Name and Pending
        cmd = New OleDbCommand(query & name & "AND " & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
        cmd.Parameters.AddWithValue("@Reimbursed", False)
    Case 11 'Date, Name and Pending
        cmd = New OleDbCommand(query & dateQuery & "AND " & name & "AND " & reimbursed & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
        cmd.Parameters.AddWithValue("@Reimbursed", False)
    Case 12 'Completed and Pending (same as everything)
        cmd = New OleDbCommand(query.Substring(0, query.Length - 6) & order, connection)
    Case 13 'Date, Completed and Pending (same as Date only)
        cmd = New OleDbCommand(query & dateQuery & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
    Case 14 'Name, Completed and Pending (same as Name only)
        cmd = New OleDbCommand(query & name & order, connection)
        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
    Case 15 'Date, Name, Completed, Pending (same as Date and Name only)
        cmd = New OleDbCommand(query & dateQuery & "AND " & name & order, connection)
        cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
        cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
        cmd.Parameters.AddWithValue("@FullName", cboName.Text)
    Case 16 'All checkbox is checked
        cmd = New OleDbCommand(query.Substring(0, query.Length - 6) & order, connection)
End Select
'Declares data adapter. Declares datatable and fills it
Dim da As New OleDbDataAdapter(cmd)
Dim dt As New DataTable
'Changes datatype of Reimbursed column if print all data is checked
If chkAll.Checked = True Then
    da.FillSchema(dt, SchemaType.Source)
    dt.Columns(6).DataType = GetType(String)
    da.Fill(dt)

```

```

Else : da.Fill(dt)
End If
'Displays data on datagridview and sets the column names
DataGridView.DataSource = dt
With DataGridView
    .Columns(0).HeaderCell.Value = "Tracking No."
    .Columns(1).HeaderCell.Value = "Name"
    .Columns(2).HeaderCell.Value = "Date"
    .Columns(3).HeaderCell.Value = "Description & Qty"
    .Columns(5).HeaderCell.Value = "Total"
End With
'Checks if datagridview is empty (i.e. no data found)
If DataGridView.Rows.Count <> 1 Then
    'If user checked checkbox to calculate total
    If chkTotal.Checked = True Then
        'Calculates total for the Total column of the datagridview
        Dim totalDec As Decimal = 0
        For i = 0 To DataGridView.Rows.Count - 1
            totalDec = totalDec + DataGridView.Rows(i).Cells(5).Value
        Next i
        Dim totalStr As String = "$" & CStr(totalDec)
        'Adds new row to DataTable to update the datagridview. Sets values in rows then adds
        Dim newRow As DataRow = dt.NewRow
        For i = 0 To 3
            newRow.Item(i) = ""
        Next
        newRow.Item(4) = "TOTAL ="
        newRow.Item(5) = totalStr
        If chkAll.Checked = True Then
            newRow.Item(6) = ""
        Else
            'Checking if all the records have been reimbursed or not
            Dim rBool As Boolean = True
            'Loops through datagridview to find any False reimbursed values
            For i = 0 To DataGridView.Rows.Count - 2
                If DataGridView.Rows(i).Cells(6).Value = False Then
                    rBool = False
                End If
            Next i
            'Sets value for Reimbursed field
            newRow.Item(6) = rBool
        End If
        'Adds the row to the datatable and updates the datagridview
        dt.Rows.Add(newRow)
    End If
End If

```

```

'Runs function to set up for printing
If SetupThePrinting("Business Expenses") Then
    'Declares print preview dialog and opens it
    Dim ppd As PrintPreviewDialog = New PrintPreviewDialog()
    ppd.Document = pntDocument
    ppd.Height = Screen.PrimaryScreen.Bounds.Height / 1.5
    ppd.Width = Screen.PrimaryScreen.Bounds.Width / 2.5
    ppd.FindForm().StartPosition = FormStartPosition.CenterScreen
    ppd.Text = "Business Expenses"
    ppd.ShowDialog()
    'Output to user to show success
    lblDataStatus.ForeColor = Color.Green
    lblDataStatus.Text = "Successful print operation."
End If
Else
    'Output if no data was found
    lblFilterStatus.Visible = True
    lblFilterStatus.ForeColor = Color.Red
    lblFilterStatus.Text = "No data found."
End If
Else
    'Feedback to show none of the checkboxes were checked
    lblFilterStatus.Text = "Please select the filters."
    lblFilterStatus.ForeColor = Color.Red
    lblFilterStatus.Visible = True
End If
End Sub

Private Sub pntDocument_PrintPage(ByVal sender As System.Object, ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles pntDocument.PrintPage
    'Checks if there are any more pages
    Dim more As Boolean = dgvPrinter.DrawDataGridView(e.Graphics)
    If more = True Then
        e.HasMorePages = True
    End If
End Sub

Private Function SetupThePrinting(ByVal title As String) As Boolean
    'Declares print dialog and sets properties
    Dim pd As PrintDialog = New PrintDialog()
    pd.AllowCurrentPage = False
    pd.AllowPrintToFile = False
    pd.AllowSelection = False
    pd.AllowSomePages = False
    pd.PrintToFile = False

```

```

pd.ShowHelp = False
pd.ShowNetwork = False
'Function returns false if user cancels
If pd.ShowDialog() <> DialogResult.OK Then
    Return False
End If
'Sets properties for the print document and sets the data
pntDocument.DocumentName = "Business Expense"
pntDocument.PrinterSettings = pd.PrinterSettings
pntDocument.DefaultPageSettings = pd.PrinterSettings.DefaultPageSettings
pntDocument.DefaultPageSettings.Margins = New Margins(20, 20, 20, 20)
dgvPrinter = New DataGridViewPrinter(DataGridView, pntDocument, True, True, title, New Font("Segoe UI", 18, FontStyle.Bold,
GraphicsUnit.Point), Color.Black, True)
Return True
End Function

Private Sub chkAll_CheckedChanged(sender As Object, e As EventArgs) Handles chkAll.CheckedChanged
    'Unchecks other checkboxes if all checkbox is checked
    If chkAll.Checked = True Then
        chkFilterDate.Checked = False
        chkFilterName.Checked = False
        chkCompleted.Checked = False
        chkPending.Checked = False
        chkAll.Checked = True
    End If
End Sub

Private Sub chkOthers_CheckedChanged(sender As Object, e As EventArgs) Handles chkFilterDate.CheckedChanged, chkFilterName.CheckedChanged, _
    chkCompleted.CheckedChanged, chkPending.CheckedChanged
    'Sets the checkbox for All to false if any of the other checkboxes have their checks changed
    chkAll.Checked = False
End Sub

Private Sub dtpDateFrom_ValueChanged(sender As Object, e As EventArgs) Handles dtpDateFrom.ValueChanged
    'Sets minimum value of the DateTo datepicker to the value of the DateFrom datepicker
    dtpDateTo.MinDate = dtpDateFrom.Value.Date
End Sub

Private Sub Business_Expense__Overview_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Shows the main menu upon the closing of the form
    MainMenu.Show()
End Sub

Private Sub lstViewPending_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles
lstViewPending.ColumnClick

```

```

'Get the new sorting column to be sorted
Dim new_sorting_column As ColumnHeader = lstViewPending.Columns(e.Column)
'Find out the sorting order
Dim sort_order As System.Windows.Forms.SortOrder
If m_SortingColumn Is Nothing Then
    'Sort column by ascending order
    sort_order = SortOrder.Ascending
Else
    'Checks if sorting column is the same one
    If new_sorting_column.Equals(m_SortingColumn) Then
        'If it is the same column then the sorting order is switched around
        If m_SortingColumn.Text.StartsWith("> ") Then
            sort_order = SortOrder.Descending
        Else
            sort_order = SortOrder.Ascending
        End If
    Else
        'Sort by ascending value
        sort_order = SortOrder.Ascending
    End If
    'Removes the old sort column indicator
    m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
End If
'Displays the new sorting column indicator
m_SortingColumn = new_sorting_column
If sort_order = SortOrder.Ascending Then
    m_SortingColumn.Text = "> " & m_SortingColumn.Text
Else
    m_SortingColumn.Text = "< " & m_SortingColumn.Text
End If
'Creates the comparer and sorts the listview accordingly
lstViewPending.ListViewItemSorter = New clsListViewSorter(e.Column, sort_order)
lstViewPending.Sort()
End Sub

Private Sub lstViewApproved_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles
lstViewCompleted.ColumnClick
    'Get the new sorting column to be sorted
    Dim new_sorting_column As ColumnHeader = lstViewCompleted.Columns(e.Column)
    'Find out the sorting order
    Dim sort_order As System.Windows.Forms.SortOrder
    If m_SortingColumn Is Nothing Then
        'Sort column by ascending order
        sort_order = SortOrder.Ascending
    Else

```

```

'Checks if sorting column is the same one
If new_sorting_column.Equals(m_SortingColumn) Then
    'If it is the same column then the sorting order is switched around
    If m_SortingColumn.Text.StartsWith("> ") Then
        sort_order = SortOrder.Descending
    Else
        sort_order = SortOrder.Ascending
    End If
Else
    'Sort by ascending value
    sort_order = SortOrder.Ascending
End If
'Removes the old sort column indicator
m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
End If
'Displays the new sorting column indicator
m_SortingColumn = new_sorting_column
If sort_order = SortOrder.Ascending Then
    m_SortingColumn.Text = "> " & m_SortingColumn.Text
Else
    m_SortingColumn.Text = "< " & m_SortingColumn.Text
End If
'Creates the comparer and sorts the listview accordingly
lstViewCompleted.ListViewItemSorter = New clsListViewSorter(e.Column, sort_order)
lstViewCompleted.Sort()
End Sub
End Class

```

Business Expense - New.vb

```

Imports System.Data.OleDb

Public Class Business_Expense__New
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

    Private Sub Business_Expense__New_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Enable textbox for Name if user is an admin
        If My.Settings.accountAdmin = True Then
            txtName.ReadOnly = False
        End If
        'Focuses on Tracking Number textbox
        txtTrackingNumber.Select()
    End Sub

    Private Sub txtTrackingNumber_TextChanged(sender As Object, e As EventArgs) Handles txtTrackingNumber.TextChanged

```

```

'Clears the textboxes
txtName.Text = ""
txtDescription.Text = ""
txtQuantity.Text = ""
txtUnit.Text = ""
txtTotal.Text = ""
txtVendor.Text = ""

Try
    'Opens the connection and declares a new OleDbCommand and Reader
    Dim TrackingNumber As String = txtTrackingNumber.Text
    connection.Open()
    Dim dt As New DataTable()
    Using da As New OleDbDataAdapter("SELECT * FROM BEA WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
        da.Fill(dt)
    End Using
    'Closes the connection
    connection.Close()
    'Sets values in textboxes to data in datatable if it exists
    txtName.Text = dt.Rows(0)("FullName").ToString
    txtDescription.Text = dt.Rows(0)("ItemDescription").ToString
    txtQuantity.Text = dt.Rows(0)("Quantity").ToString
    txtUnit.Text = dt.Rows(0)("UnitPrice").ToString
    txtTotal.Text = dt.Rows(0)("TotalPrice").ToString
    txtVendor.Text = dt.Rows(0)("Vendor").ToString
Catch
End Try
End Sub

Private Sub btnSave_Click(sender As Object, e As EventArgs) Handles btnSave.Click
If Validation() Then
    Try
        'Declares the new OleDbCommand
        Dim cmd As New OleDbCommand("INSERT INTO [Business Expenses] (TrackingNumber, FullName, DateIncurred, Description, " & _
            "Quantity, UnitPrice, TotalPrice, ReimbursementRequired, PaymentMethod, Vendor, Reimbursed) " & _
            "VALUES (@TrackingNumber, @FullName, @DateIncurred, @Description, @Quantity, @UnitPrice, " & _
            "@TotalPrice, @ReimbursementRequired, @PaymentMethod, @Vendor, @Reimbursed)", connection)
        'Gives the parameters its values
        cmd.Parameters.AddWithValue("TrackingNumber", txtTrackingNumber.Text)
        cmd.Parameters.AddWithValue("FullName", txtName.Text)
        cmd.Parameters.AddWithValue("DateIncurred", dtpDate.Value.Date)
        cmd.Parameters.AddWithValue("Description", txtDescription.Text)
        cmd.Parameters.AddWithValue("Quantity", txtQuantity.Text)
        cmd.Parameters.AddWithValue("UnitPrice", txtUnit.Text)
        cmd.Parameters.AddWithValue("TotalPrice", txtTotal.Text)
        cmd.Parameters.AddWithValue("ReimbursementRequired", chkReimbursement.Checked)
    End Try
End If
End Sub

```

```

        cmd.Parameters.AddWithValue("PaymentMethod", cboMethod.Text)
        cmd.Parameters.AddWithValue("Vendor", txtVendor.Text)
        'If reimbursement is not required then reimbursed is set to true
        If chkReimbursement.Checked = False Then
            cmd.Parameters.AddWithValue("Reimbursed", True)
        Else : cmd.Parameters.AddWithValue("Reimbursed", False)
        End If
                'Opens database connection, executes query and closes the database
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Output to show successful updation on Overview form
        Business_Expense__Overview.lblDataStatus.Text = "Record successfully added."
        Business_Expense__Overview.lblDataStatus.ForeColor = Color.Green
        Business_Expense__Overview.lblDataStatus.Visible = True
        'Closes the form completely
        Me.Dispose()
    Catch
        'Closes the connection and shows output to user to show an error
        connection.Close()
        lblStatus.ForeColor = Color.Red
        lblStatus.Text = "A record for the tracking number already exists."
    End Try
End If
End Sub

Private Function Validation()
    'Checks if all the textboxes or combobox contain text
    If txtTrackingNumber.Text <> "" And txtName.Text <> "" And txtDescription.Text <> "" And cboMethod.Text <> "" And _
       txtQuantity.Text <> "" And txtUnit.Text <> "" And txtTotal.Text <> "" And txtVendor.Text <> "" Then
        Return True
    Else
        'Output to show that one or more of the textboxes have not been filled.
        lblStatus.ForeColor = Color.Red
        lblStatus.Text = "Please ensure all fields are filled out correctly."
        Return False
    End If
End Function

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Displays a MsgBox asking user to confirm to close the form
    Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
    'Completely closes form if user selects 'Yes'
    If Response = MsgBoxResult.Yes Then
        Me.Dispose()
    End If

```

```

End Sub

Private Sub Business_Expense__New_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Completely closes form and runs subroutines to refresh the listviews
    Me.Dispose()
    Business_Expense__Overview.populateListViewPending()
    Business_Expense__Overview.populateListViewCompleted()
End Sub

Private Sub TotalCost_TextChanged(sender As Object, e As EventArgs) Handles txtUnit.TextChanged, txtQuantity.TextChanged
    'Declares variables and set them equal to the value in their respective textboxes
    Dim quantity, unitCost, total As Double
    quantity = Val(txtQuantity.Text)
    unitCost = Val(txtUnit.Text)
    'Multiplies the value in the quantity textbox with the cost in the unit cost box to give total
    total = quantity * unitCost
    txtTotal.Text = total
End Sub

Private Sub txtQuantity_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtQuantity.KeyPress
    'Runs subroutine on textFormatting class to allow numbers only into textbox
    textFormatting.numbersOnly(sender, e)
End Sub

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtUnit.KeyPress, txtTotal.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

Private Sub cboMethod_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cboMethod.SelectedIndexChanged
    'Disables the checkbox when the payment method is company
    If cboMethod.Text = "Company" Then
        chkReimbursement.Checked = False
        chkReimbursement.Enabled = False
    ElseIf cboMethod.Text = "Personal" Then 'Checks the checkbox if payment method is personal
        chkReimbursement.Enabled = True
        chkReimbursement.Checked = True
    Else : chkReimbursement.Enabled = True
    End If
End Sub

```

Business Expense - View and Edit.vb

```
Imports System.Data.OleDb

Public Class Business_Expense__View_and_Edit
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

    Private Sub Business_Expense__View_and_Edit_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Opens the connection and declares a new OleDbCommand and Reader
        Dim TrackingNumber As String = txtTrackingNumber.Text
        connection.Open()
        Dim dt As New DataTable()
        'Fills the datatable with the appropriate data from the Business Expenses table
        Using da As New OleDbDataAdapter("SELECT * FROM [Business Expenses] WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
            da.Fill(dt)
        End Using
        'Closes the connection
        connection.Close()
        'Sets values in textboxes, comboboxes and checkboxes to data in datatable
        txtName.Text = dt.Rows(0)("FullName").ToString
        dtpDate.Value = dt.Rows(0)("DateIncurred")
        txtVendor.Text = dt.Rows(0)("Vendor").ToString
        cboMethod.Text = dt.Rows(0)("PaymentMethod").ToString
        txtDescription.Text = dt.Rows(0)("Description").ToString
        txtQuantity.Text = dt.Rows(0)("Quantity").ToString
        txtUnit.Text = dt.Rows(0)("UnitPrice").ToString
        txtTotal.Text = dt.Rows(0)("TotalPrice").ToString
        chkReimbursementRequired.Checked = dt.Rows(0)("ReimbursementRequired")
        chkReimbursed.Checked = dt.Rows(0)("Reimbursed")
        'Checks if the user is an admin or not
        If My.Settings.accountAdmin = True Then
            'Enables the 'name' textbox and the checkbox for reimbursed
            txtName.ReadOnly = False
            chkReimbursed.Enabled = True
        End If
        'Focuses on form
        PictureBox.Select()
    End Sub

    Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
        If Validation() Then
            'Declares the new OleDbCommand and gives it parameters
            Dim cmd As New OleDbCommand("UPDATE [Business Expenses] SET FullName = ?, DateIncurred = ?, Description = ?, Quantity = ?, " & _
                "UnitPrice = ?, TotalPrice = ?, ReimbursementRequired = ?, PaymentMethod = ?, Vendor = ?, " & _
                "Reimbursed = ? WHERE TrackingNumber = '" & txtTrackingNumber.Text & "'", connection)

```

```

cmd.Parameters.AddWithValue("@p1", txtName.Text)
cmd.Parameters.AddWithValue("@p2", dtpDate.Value.Date)
cmd.Parameters.AddWithValue("@p3", txtDescription.Text)
cmd.Parameters.AddWithValue("@p4", txtQuantity.Text)
cmd.Parameters.AddWithValue("@p5", txtUnit.Text)
cmd.Parameters.AddWithValue("@p6", txtTotal.Text)
cmd.Parameters.AddWithValue("@p7", chkReimbursementRequired.Checked)
cmd.Parameters.AddWithValue("@p8", cboMethod.Text)
cmd.Parameters.AddWithValue("@p9", txtVendor.Text)
cmd.Parameters.AddWithValue("@p10", chkReimbursed.Checked)
'Executes the command and closes form
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Output to show successful updation on Overview form and closes form
Business_Expense__Overview.lblDataStatus.Text = "Record successfully updated."
Business_Expense__Overview.lblDataStatus.ForeColor = Color.Green
Business_Expense__Overview.lblDataStatus.Visible = True
Me.Close()
End If
End Sub

Private Function Validation()
'First checks if all the textboxes and comboboxes contain text
If txtTrackingNumber.Text <> "" And txtName.Text <> "" And txtDescription.Text <> "" And cboMethod.Text <> "" And _
txtQuantity.Text <> "" And txtUnit.Text <> "" And txtTotal.Text <> "" And txtVendor.Text <> "" Then
    Return True
Else
    'Output to show that one or more of the textboxes have not been filled.
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "Please ensure all fields are filled out correctly."
    Return False
End If
End Function

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
'Opens a Yes/No messagebox asking user to confirm to delete the record
Dim Response = MsgBox("Are you sure you want to delete this Business Expense Application?", MsgBoxStyle.YesNo, "Delete Application")
'Code if user selects yes
If Response = MsgBoxResult.Yes Then
    'Declares a new OleDbCommand which deletes the record
    Dim TrackingNumber As String = txtTrackingNumber.Text
    Dim cmd As New OleDbCommand("DELETE FROM [Business Expenses] WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
    'Opens connection, executes the command and closes the connection
    connection.Open()

```

```

        cmd.ExecuteNonQuery()
        connection.Close()
        'Output to show successful updation on Overview form
        Business_Expense__Overview.lblDataStatus.Text = "Record successfully deleted."
        Business_Expense__Overview.lblDataStatus.ForeColor = Color.Green
        Business_Expense__Overview.lblDataStatus.Visible = True
        Me.Close()
    End If
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Displays a MsgBox asking user to confirm to close the form
    Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
    'Closes the form if user responds yes
    If Response = MsgBoxResult.Yes Then
        Me.Close()
    End If
End Sub

Private Sub Business_Expense__View_and_Edit_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'When this form closes, subroutines will be run to repopulate the listviews in the overview form
    Business_Expense__Overview.populateListViewPending()
    Business_Expense__Overview.populateListViewCompleted()
    'Closes form completely
    Me.Dispose()
End Sub

Private Sub TotalCost_TextChanged(sender As Object, e As EventArgs) Handles txtUnit.TextChanged, txtQuantity.TextChanged
    'Declares variables and initialises them to the values in the relevant textboxes
    Dim quantity, unitCost, total As Double
    quantity = Val(txtQuantity.Text)
    unitCost = Val(txtUnit.Text)
    'Multiplies the value in the quantity textbox with the cost in the unit cost box to give total
    total = quantity * unitCost
    txtTotal.Text = total
End Sub

Private Sub txtQuantity_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtQuantity.KeyPress
    'Runs subroutine on textFormatting class to allow numbers only into textbox
    textFormatting.numbersOnly(sender, e)
End Sub

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtUnit.KeyPress,
txtTotal.KeyPress

```

```

    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

Business Expense Application - Overview.vb

```

Imports System.Data.OleDb
Imports Odin.ListViewSort

Public Class Business_Expense_Application__Overview
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Private m_SortingColumn As ColumnHeader

    Private Sub Business_Expense_Application__Overview_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Populates the listviews and focuses on the form
        populateListViewApproved()
        populateListViewPending()
        PictureBox.Select()
    End Sub

    Public Sub populateListViewPending()
        'Clears and enables the listview
        lstViewPending.Items.Clear()
        lstViewPending.Enabled = True
        'Opens the connection and declares a new OleDbCommand and Reader
        connection.Open()
        Dim cmd As New OleDbCommand("SELECT * From BEA WHERE Approved = False ORDER BY TrackingNumber", connection)
        Dim dr As OleDbDataReader = cmd.ExecuteReader
        'Loops the Accounts table of the database and adds each record to a new ListViewItem
        Do While dr.Read()
            Dim newitem As New ListViewItem(dr.Item("TrackingNumber").ToString)
            newitem.SubItems.Add(dr.Item("FullName").ToString)
            newitem.SubItems.Add(dr.Item("ItemDescription").ToString)
            newitem.SubItems.Add(dr.Item("Quantity").ToString)
            newitem.SubItems.Add(dr.Item("TotalPrice"))
            lstViewPending.Items.Add(newitem)
        Loop
        'Closes the connection
        connection.Close()
    End Sub

    Public Sub populateListViewApproved()
        'Clears and enables the listview

```

```

lstViewApproved.Items.Clear()
lstViewApproved.Enabled = True
'Opens the connection and declares a new OleDbCommand and Reader
connection.Open()
Dim cmd As New OleDbCommand("SELECT * From BEA WHERE Approved = True ORDER BY TrackingNumber", connection)
Dim dr As OleDbDataReader = cmd.ExecuteReader
'Loops the Accounts table of the database and adds each record to a new ListViewItem
Do While dr.Read()
    Dim newitem As New ListViewItem(dr.Item("TrackingNumber").ToString)
    newitem.SubItems.Add(dr.Item("FullName").ToString)
    newitem.SubItems.Add(dr.Item("ItemDescription").ToString)
    newitem.SubItems.Add(dr.Item("Quantity").ToString)
    newitem.SubItems.Add(dr.Item("TotalPrice"))
    lstViewApproved.Items.Add(newitem)
Loop
'Closes the connection
connection.Close()
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles btnNew.Click
    'Opens the form for a new business expense application
    Business_Expensive_Application__New.ShowDialog()
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Checks that either of the listviews has an item selected
    If lstViewPending.SelectedItems.Count > 0 Or lstViewApproved.SelectedItems.Count > 0 Then
        'Displays a MsgBox asking user to confirm to delete the record
        Dim Response = MsgBox("Are you sure you want to delete this Business Expense Application?", MsgBoxStyle.YesNo,
                               "Delete Business Expense Application")
        If Response = MsgBoxResult.Yes Then
            'If response is yes then checks if an item is selected in the 'pending' listview or not
            If lstViewPending.SelectedItems.Count > 0 Then
                'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
                Dim pending As String = lstViewPending.SelectedItems(0).Text
                Dim cmd As New OleDbCommand("DELETE FROM BEA WHERE TrackingNumber = '" & pending & "'", connection)
                'Opens connection, executes the command and closes the connection
                connection.Open()
                cmd.ExecuteNonQuery()
                connection.Close()
                'Repopulates the listview
                populateListViewPending()
                'Updates label to show status
                lblDataStatus.Text = "Record successfully deleted."
                lblDataStatus.Visible = True
            End If
        End If
    End If
End Sub

```

```

    'Checks if selected item is in 'completed' listview
ElseIf lstViewApproved.SelectedItems.Count > 0 Then
    'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
    Dim approved As String = lstViewApproved.SelectedItems(0).Text
    Dim cmd As New OleDbCommand("DELETE FROM BEA WHERE TrackingNumber = '" & approved & "'", connection)
    'Opens connection, executes the command and closes the connection
    connection.Open()
    cmd.ExecuteNonQuery()
    connection.Close()
    'Repopulates the listview
    populateListViewApproved()
    'Updates label to show status
    lblDataStatus.Text = "Record successfully deleted."
    lblDataStatus.Visible = True
End If
End If
'If no item is selected then output error
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnEdit_Click(sender As Object, e As EventArgs) Handles btnEdit.Click
    'Checks if an item is selected in either listview or not
    If lstViewPending.SelectedItems.Count > 0 Then
        'Passes value of ID to Edit form and opens it
        Dim TrackingNumber As String = lstViewPending.SelectedItems(0).Text
        Business_Expense_Applications__View_and_Edit.lblTracking.Text = TrackingNumber
        Business_Expense_Applications__View_and_Edit.ShowDialog()
    ElseIf lstViewApproved.SelectedItems.Count > 0 Then
        'Passes value of ID to Edit form and opens it
        Dim TrackingNumber As String = lstViewApproved.SelectedItems(0).Text
        Business_Expense_Applications__View_and_Edit.lblTracking.Text = TrackingNumber
        Business_Expense_Applications__View_and_Edit.ShowDialog()
        'If neither listviews has an item selected then output error
    Else : MsgBox("Please select a record from the table.")
    End If
End Sub

Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles btnSearch.Click
    'Opens SearchBox form and passes appropriate function name
    SearchBox.lblFunction.Text = "BEA"
    SearchBox.ShowDialog()
End Sub

```

```

    Private Sub Business_Expense_Application__Overview_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
        'Opens Main Menu form upon closing of the overview form
        MainMenu.Show()
    End Sub

    Private Sub lstViewPending_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles lstViewPending.ColumnClick
        'Get the new sorting column to be sorted
        Dim new_sorting_column As ColumnHeader = lstViewPending.Columns(e.Column)
        'Find out the sorting order
        Dim sort_order As System.Windows.Forms.SortOrder
        If m_SortingColumn Is Nothing Then
            'Sort column by ascending order
            sort_order = SortOrder.Ascending
        Else
            'Checks if sorting column is the same one
            If new_sorting_column.Equals(m_SortingColumn) Then
                'If it is the same column then the sorting order is switched around
                If m_SortingColumn.Text.StartsWith("> ") Then
                    sort_order = SortOrder.Descending
                Else
                    sort_order = SortOrder.Ascending
                End If
            Else
                'Sort by ascending value
                sort_order = SortOrder.Ascending
            End If
            'Removes the old sort column indicator
            m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
        End If
        'Displays the new sorting column indicator
        m_SortingColumn = new_sorting_column
        If sort_order = SortOrder.Ascending Then
            m_SortingColumn.Text = "> " & m_SortingColumn.Text
        Else
            m_SortingColumn.Text = "< " & m_SortingColumn.Text
        End If
        'Creates the comparer and sorts the listview accordingly
        lstViewPending.ListViewItemSorter = New clsListViewSorter(e.Column, sort_order)
        lstViewPending.Sort()
    End Sub

    Private Sub lstViewApproved_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles lstViewApproved.ColumnClick

```

```

'Get the new sorting column to be sorted
Dim new_sorting_column As ColumnHeader = lstViewApproved.Columns(e.Column)
'Find out the sorting order
Dim sort_order As System.Windows.Forms.SortOrder
If m_SortingColumn Is Nothing Then
    'Sort column by ascending order
    sort_order = SortOrder.Ascending
Else
    'Checks if sorting column is the same one
    If new_sorting_column.Equals(m_SortingColumn) Then
        'If it is the same column then the sorting order is switched around
        If m_SortingColumn.Text.StartsWith("> ") Then
            sort_order = SortOrder.Descending
        Else
            sort_order = SortOrder.Ascending
        End If
    Else
        'Sort by ascending value
        sort_order = SortOrder.Ascending
    End If
    'Removes the old sort column indicator
    m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
End If
'Displays the new sorting column indicator
m_SortingColumn = new_sorting_column
If sort_order = SortOrder.Ascending Then
    m_SortingColumn.Text = "> " & m_SortingColumn.Text
Else
    m_SortingColumn.Text = "< " & m_SortingColumn.Text
End If
'Creates the comparer and sorts the listview accordingly
lstViewApproved.ListViewItemSorter = New clsListViewSorter(e.Column, sort_order)
lstViewApproved.Sort()
End Sub
End Class

```

Business Expense Application - New.vb

```

Imports System.Data.OleDb
Imports System.Net.Mail
Imports Odin.textFormatting

Public Class Business_Expensive_Application__New
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

```

```

Private Sub Business_Expensive_Application__New_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    'Runs subroutine to populate the combination box for Name and focuses the controls on the form
    populateCBO()
    PictureBox.Select()
    'Selects the users username in the combination box and sets email textbox
    cboName.Text = My.Settings.accountFullName
    txtEmail.Text = My.Settings.accountUsername & "@odinhealth.co.nz"
    'Sets the Tracking Number.
    If My.Settings.BEADate = DateTime.Today Then
        'Sets tracking number if the settings variable date is the same as todays date
        lblTracking.Text = "BS" & DateTime.Now.ToString("ddMMyy") & My.Settings.BEACount
    Else
        'Sets the settings variable to todays date if it does not match todays date
        My.Settings.BEADate = DateTime.Today
        'Sets the count for the number of applications to 0
        My.Settings.BEACount = 0
        lblTracking.Text = "BS" & DateTime.Now.ToString("ddMMyy") & My.Settings.BEACount
    End If
End Sub

Private Sub cboName_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cboName.SelectedIndexChanged
    'Update the textbox containing the email when combobox selection is altered
    Dim Name As String = cboName.Text
    Name = Name.ToLower
    Name = Name.Replace(" ", ".")
    txtEmail.Text = Name & "@odinhealth.co.nz"
End Sub

Private Sub btnSubmit_Click(sender As Object, e As EventArgs) Handles btnSubmit.Click
    'Checks if the function returned a true
    If Validation() Then
        'Declares the new OleDbCommand
        Dim cmd As New OleDbCommand("INSERT INTO BEA (TrackingNumber, FullName, Email, ItemDescription, Quantity, " & _
                                    "UnitPrice, TotalPrice, Vendor, Reason, Approved) " & _
                                    "VALUES (@TrackingNumber, @FullName, @Email, @ItemDescription, @Quantity, @UnitPrice, " & _
                                    "@TotalPrice, @Vendor, @Reason, @Approved)", connection)
        'Gives the parameters its values
        cmd.Parameters.AddWithValue("TrackingNumber", lblTracking.Text)
        cmd.Parameters.AddWithValue("FullName", cboName.Text)
        cmd.Parameters.AddWithValue("Email", txtEmail.Text)
        cmd.Parameters.AddWithValue("ItemDescription", txtDescription.Text)
        cmd.Parameters.AddWithValue("Quantity", txtQuantity.Text)
        cmd.Parameters.AddWithValue("UnitPrice", txtUnit.Text)
        cmd.Parameters.AddWithValue("TotalPrice", txtTotal.Text)
        cmd.Parameters.AddWithValue("Vendor", txtVendor.Text)
    End If
End Sub

```

```

cmd.Parameters.AddWithValue("Reason", rtbReason.Text)
cmd.Parameters.AddWithValue("Approved", False)
'Opens database connection, executes query and closes the database
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Adds 1 onto the count for making the application number
My.Settings.BEACount = My.Settings.BEACount + 1
'Output to show successful update on Overview form
Travel_Application__Overview.lblDataStatus.Text = "Record successfully added."
Travel_Application__Overview.lblDataStatus.Visible = True
'Tries to send an email
Try
    'Runs subroutine to send an email and sets feedback on the overview form
    SendEmail()
    Travel_Application__Overview.lblDataStatus.Text = "Record added and email sent."
    'Completely closes the form
    Me.Dispose()
Catch
    'Output to show that there was a problem sending the email but record was added
    lblStatus.ForeColor = Color.Green
    MsgBox("Application successfully added into database but email did not successfully send.")
    'Closes the form completely
    Me.Dispose()
End Try
End If
End Sub

Private Function Validation()
    'First checks if all the textboxes contain text
    If txtEmail.Text <> "" And txtDescription.Text <> "" And txtQuantity.Text <> "" And txtUnit.Text <> "" And txtTotal.Text <> "" And txtVendor.Text <> "" And rtbReason.Text <> "" Then
        'Validates the email address. Throws error if the address is in the wrong format
        Try
            'The function returns true
            Dim Email As New MailAddress(txtEmail.Text)
            Return True
        Catch
            'Output to show that email address format was incorrect.
            lblStatus.ForeColor = Color.Red
            lblStatus.Text = "Please check that the email address you entered was in a correct format."
            Return False
        End Try
    Else
        'Output to show that one or more of the textboxes have not been filled.

```

```

lblStatus.ForeColor = Color.Red
lblStatus.Text = "Please ensure all fields are filled out correctly."
Return False
End If
End Function

Private Sub populateCBO()
    'Declares DataSet and fills it up with data from Username Column in database
    Dim ds As New DataSet()
    Using da As New OleDbDataAdapter("SELECT FullName FROM Accounts", connection)
        da.Fill(ds, "FullName")
    End Using
    'Loads DataSet data into a DataView, sorts it then writes it to a DataTable
    Dim view As New DataView(ds.Tables("FullName"))
    view.Sort = "FullName"
    Dim sorted As DataTable = view.ToTable()
    'Populates combobox with contents of the new Datatable
    With cboName
        .DisplayMember = "FullName"
        .DataSource = view
        .SelectedIndex = 0
    End With
End Sub

Private Sub Business_Expense_Application__New_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Closes form completely and runs subroutines to refresh the listviews on the overview form
    Me.Dispose()
    Business_Expense_Application__Overview.populateListViewPending()
    Business_Expense_Application__Overview.populateListViewApproved()
End Sub

Private Sub SendEmail()
    Me.Enabled = False
    'Output to show that validation was successful.
    lblStatus.ForeColor = Color.ForestGreen
    lblStatus.Text = "Validation successful - Adding application to database and sending email for approval. Please wait a moment..."
    'Creates new message and declares Name variable
    Dim Email As New MailMessage()
    Dim Name As String = cboName.Text
    'Sets the email preferences
    Email.From = New MailAddress(txtEmail.Text)
    Email.[To].Add(My.Settings.applicationEmail)
    'Sets the content of the email
    Email.Subject = "Business Expense Application #" & lblTracking.Text
    Email.Body = "<H2>Business Expense Application for " & Name & "</H2>" & "Date of Submission: " & DateTime.Today & _

```

```

"<BR>Return Email: " & txtEmail.Text & "<BR>Item Description: " & txtDescription.Text & " (Quantity: " & txtQuantity.Text & _
")<BR>" & "Unit Price: $" & txtUnit.Text & " NZD<BR>Total Price: $" & txtTotal.Text & " NZD<BR>Vendor: " & _
txtVendor.Text & "<BR><BR>Reason for Expense: " & rtbReason.Text & "<BR><BR>" &
"To approve this application, open the Business Manager program and navigate to the Business Expense " & _
"Applications, select the application click the 'View and Edit' button and then tick the approved box and 'Update'."  

Email.IsBodyHtml = True  

'Sets the server to send the email  

Dim smtp As New SmtpClient("smtp.gmail.com")  

smtp.UseDefaultCredentials = False  

smtp.Credentials = New Net.NetworkCredential("shen.odinhealth@gmail.com", "c9dMy6RKPr")  

smtp.Port = 587  

smtp.EnableSsl = True  

smtp.Host = "smtp.gmail.com"  

'Sends the email and shows feedback  

smtp.Send(Email)  

lblStatus.ForeColor = Color.Green  

lblStatus.Text = "Application added into database and email has been successfully sent for approval. Thank You!"  

Me.Enabled = True  

End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
'Displays a MsgBox asking user to confirm to close the form  

Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
'Completely closes the form if user responds 'Yes'  

If Response = MsgBoxResult.Yes Then
    Me.Dispose()
End If
End Sub

Private Sub TotalCost_TextChanged(sender As Object, e As EventArgs) Handles txtUnit.TextChanged, txtQuantity.TextChanged
'Declares appropriate variables and sets them to the values in the relevant textboxes  

Dim quantity, unitCost, total As Double  

quantity = Val(txtQuantity.Text)  

unitCost = Val(txtUnit.Text)
'Multiplies the value in the quantity textbox with the cost in the unit cost box to give total  

total = quantity * unitCost  

txtTotal.Text = total
End Sub

Private Sub txtQuantity_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtQuantity.KeyPress
'Runs subroutine on textFormatting class to allow numbers only into textbox  

textFormatting.numbersOnly(sender, e)
End Sub

```

```

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtUnit.KeyPress,
txtTotal.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

Business Expense Application - View and Edit.vb

```

Imports System.Data.OleDb
Imports System.Net.Mail

Public Class Business_Expense_Applications__View_and_Edit
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Dim Approved As Boolean

    Private Sub Business_Expense_Applications__View_and_Edit_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Opens the connection and declares a new OleDbCommand and Reader and fills datatable with data from BEA table
        Dim TrackingNumber As String = lblTracking.Text
        connection.Open()
        Dim dt As New DataTable()
        Using da As New OleDbDataAdapter("SELECT * FROM BEA WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
            da.Fill(dt)
        End Using
        'Closes the connection
        connection.Close()
        'Sets values in textboxes to data in datatable
        txtName.Text = dt.Rows(0)("FullName").ToString
        txtEmail.Text = dt.Rows(0)("Email").ToString
        txtDescription.Text = dt.Rows(0)("ItemDescription").ToString
        txtQuantity.Text = dt.Rows(0)("Quantity").ToString
        txtUnit.Text = dt.Rows(0)("UnitPrice").ToString
        txtTotal.Text = dt.Rows(0)("TotalPrice").ToString
        txtVendor.Text = dt.Rows(0)("Vendor").ToString
        rtbReason.Text = dt.Rows(0)("Reason").ToString
        chkApproved.Checked = dt.Rows(0)("Approved")
        'Checks if user is an admin or not
        If My.Settings.accountAdmin = True Then
            'Enables textbox to change the name and checkbox to approve the application
            txtName.ReadOnly = False
            chkApproved.Enabled = True
        End If
        'Sets "Approved" variable to true or false which will be used in sending email to user.
        If chkApproved.Checked = False Then

```

```

    Approved = False
Else
    Approved = True
End If
'Focuses the form
PictureBox.Select()
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Displays a MsgBox asking user to confirm to close the form
    Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
    'Closes the form if user selects 'Yes'
    If Response = MsgBoxResult.Yes Then
        Me.Close()
    End If
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Displays a MsgBox asking user to confirm to delete the application
    Dim Response = MsgBox("Are you sure you want to delete this Business Expense Application?", MsgBoxStyle.YesNo,
                           "Delete Application")
    If Response = MsgBoxResult.Yes Then
        'Declares a new OleDbCommand which deletes the record if response is yes
        Dim TrackingNumber As String = lblTracking.Text
        Dim cmd As New OleDbCommand("DELETE FROM BEA WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
        'Opens connection, executes the command and closes the connection
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Closes the form
        Me.Close()
    End If
End Sub

Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
    If Validation() Then
        'Declares the new OleDbCommand and gives it parameters
        Dim cmd As New OleDbCommand("UPDATE BEA SET FullName = ?, Email = ?, ItemDescription = ?, Quantity = ?, " & _
                                   "UnitPrice = ?, TotalPrice = ?, Vendor = ?, Reason = ?, Approved = ?" & _
                                   "WHERE TrackingNumber = '" & lblTracking.Text & "'", connection)
        cmd.Parameters.AddWithValue("@p1", txtName.Text)
        cmd.Parameters.AddWithValue("@p2", txtEmail.Text)
        cmd.Parameters.AddWithValue("@p3", txtDescription.Text)
        cmd.Parameters.AddWithValue("@p4", txtQuantity.Text)
        cmd.Parameters.AddWithValue("@p5", txtUnit.Text)
    End If
End Sub

```

```

cmd.Parameters.AddWithValue("@p6", txtTotal.Text)
cmd.Parameters.AddWithValue("@p7", txtVendor.Text)
cmd.Parameters.AddWithValue("@p8", rtbReason.Text)
cmd.Parameters.AddWithValue("@p9", chkApproved.Checked)
'Executes the command and closes form
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Output to show successful update on Overview form
Business_Expense_Application__Overview.lblDataStatus.Text = "Record successfully updated."
Business_Expense_Application__Overview.lblDataStatus.ForeColor = Color.Green
Business_Expense_Application__Overview.lblDataStatus.Visible = True
'If the checkbox was initially unchecked and is now checked then try send email
If Approved = False And chkApproved.Checked = True Then
    Try
        'Runs subroutine to send email
        SendEmail()
        'Output to show successful update and email sent on Overview form and closes form
        Travel_Application__Overview.lblDataStatus.Text = "Record updated and email sent."
        Me.Close()
    Catch
        'Output in case email doesn't send and closes form
        MsgBox("Email could not be successfully sent but record was updated.")
        Me.Close()
    End Try
End If
'Closes form once record has been updated
Me.Close()
End If
End Sub

Private Function Validation()
    'First checks if all the textboxes and the richtextbox contain text
    If txtName.Text <> "" And txtEmail.Text <> "" And txtDescription.Text <> "" And txtQuantity.Text <> "" And txtUnit.Text <> "" _ 
    And txtTotal.Text <> "" And txtVendor.Text <> "" And rtbReason.Text <> "" Then
        'Validates the email address. Throws error if the address is in the wrong format
        Try
            'The function returns true
            Dim Email As New MailAddress(txtEmail.Text)
            Return True
        Catch
            'Output to show that email address format was incorrect.
            lblStatus.ForeColor = Color.Red
            lblStatus.Text = "Please check that the email address you entered was in a correct format."
            Return False
        End Try
    End If
End Function

```

```

    End Try
Else
    'Output to show that one or more of the textboxes have not been filled.
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "Please ensure all fields are filled out correctly."
    Return False
End If
End Function

Private Sub Business_Expense_Application__View_and_Edit_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'When the form closes, subroutines will be run to repopulate the listviews on the overview form
    Business_Expense_Application__Overview.populateListViewPending()
    Business_Expense_Application__Overview.populateListViewApproved()
    'Closes form completely
    Me.Dispose()
End Sub

Private Sub SendEmail()
    'Disables the form
    Me.Enabled = False
    'Output to show that validation was successful.
    lblStatus.ForeColor = Color.ForestGreen
    lblStatus.Text = "Validation successful - Adding application to database and sending email for approval. Please wait a moment..."
    'Creates new message and declares Name variable
    Dim Email As New MailMessage()
    Dim Name As String = txtName.Text
    'Sets the email preferences
    Email.From = New MailAddress("shen.odinhealth@gmail.com")
    Email.[To].Add(txtEmail.Text)
    'Sets the content of the email and formats it
    Email.Subject = "Approval - Business Expense Application #" & lblTracking.Text
    Email.Body = "<H2>Business Expense Application for " & Name & " has been approved.</H2>" & "Date of Approval: " & DateTime.Today & _
        "<BR>Details for " & lblTracking.Text & "<BR><BR>Item Description: " & txtDescription.Text & " (Quantity: " & txtQuantity.Text & _
        ")<BR>Unit Price: $" & txtUnit.Text & " NZD<BR>Total Price: $" & txtTotal.Text & " NZD<BR>Vendor: " & _
        txtVendor.Text & "<BR><BR>Reason for Expense: " & rtbReason.Text & "<BR><BR>" & _
        "You have been approved for this business expense. You may now go ahead and purchase the relevant expense!"
    Email.IsBodyHtml = True
    'Sets the server to send the email
    Dim smtp As New SmtpClient("smtp.gmail.com")
    smtp.UseDefaultCredentials = False
    smtp.Credentials = New Net.NetworkCredential("shen.odinhealth@gmail.com", "c9dMy6RKPr")
    smtp.Port = 587
    smtp.EnableSsl = True
    smtp.Host = "smtp.gmail.com"

```

```

'Sends the email and shows feedback
smtp.Send(Email)
lblStatus.ForeColor = Color.Green
lblStatus.Text = "Application updated and email has been successfully sent."
'Enables form
Me.Enabled = True
End Sub

Private Sub TotalCost_TextChanged(sender As Object, e As EventArgs) Handles txtUnit.TextChanged, txtQuantity.TextChanged
    'Declares variables and sets them to the values in the relevant textboxes
    Dim quantity, unitCost, total As Double
    quantity = Val(txtQuantity.Text)
    unitCost = Val(txtUnit.Text)
    'Multiplies the value in the quantity textbox with the cost in the unit cost box to give total
    total = quantity * unitCost
    txtTotal.Text = total
End Sub

Private Sub txtQuantity_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtQuantity.KeyPress
    'Runs subroutine on textFormatting class to allow numbers only into textbox
    textFormatting.numbersOnly(sender, e)
End Sub

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtUnit.KeyPress, txtTotal.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

Contact.vb

```

Imports System.Net.Mail

Public Class Contact
    Private Sub rtbMessage_TextChanged(sender As Object, e As EventArgs) Handles rtbMessage.TextChanged
        'Retrieves the length of the characters inside the RichTextBox and updates the label
        Dim number As Integer = rtbMessage.TextLength
        lblCounter.Text = number & "/1000"
    End Sub

    Private Sub btnSend_Click(sender As Object, e As EventArgs) Handles btnSend.Click
        'Runs the validation subroutine and sees if it returns true
        If Validation() Then

```

```

'Tries sending an email
Try
    'Runs subroutine to send an email
    SendEmail()
Catch ex As Exception
    'Output to show an error if there is an error sending the email
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "Something went wrong. The email could not be sent, please try again later."
End Try
End If
End Sub

Private Function Validation()
    'Checks that the textboxes and richtextbox contain text and the checkbox is checked
    If txtName.Text <> "" And txtEmail.Text <> "" And rtbMessage.Text <> "" And chkSpam.Checked = True Then
        'Validates the email address. Throws error if the address is in the wrong format
        Try
            'The function returns true if email address is valid
            Dim Email As New MailAddress(txtEmail.Text)
            Return True
        Catch
            'Output to show that email address format was incorrect.
            lblStatus.ForeColor = Color.Red
            lblStatus.Text = "Please check that the email address you entered was in a correct format."
            Return False
        End Try
    Else
        'Outputs to user that they have not filled in all necessary fields
        lblStatus.ForeColor = Color.Red
        lblStatus.Text = "Please ensure that all fields are filled out and the 'checkbox' is checked."
        Return False
    End If
End Function

Private Sub SendEmail()
    'Creates new message
    Dim Email As New MailMessage()
    'Sets the email preferences
    Email.From = New MailAddress(txtEmail.Text)
    Email.[To].Add("w.shen+odin@kings.net.nz")
    'Sets the content of the email
    Email.Subject = "Odin Support Request from " & txtName.Text
    Email.Body = "<H2>Support Request for 'Odin Health Business Manager'</H2>" & "Time Submitted: " & _
    DateTime.Now.ToString("dd/MM/yyyy HH:mm:ss") & "<BR>Name: " & txtName.Text & "<BR>Email Address: " & _
    txtEmail.Text & "<BR><BR><u>Problem Description:</u><BR>" & rtbMessage.Text

```

```

Email.IsBodyHtml = True
'Sets the server to send the email
Dim smtp As New SmtpClient("smtp.gmail.com")
smtp.UseDefaultCredentials = False
smtp.Credentials = New Net.NetworkCredential("shen.odinhealth@gmail.com", "c9dMy6RKPr")
smtp.Port = 587
smtp.EnableSsl = True
smtp.Host = "smtp.gmail.com"
'Sends the email and shows feedback
disableControls()
smtp.Send(Email)
lblStatus.ForeColor = Color.Green
lblStatus.Text = "Your message has been successfully sent. Thank You."
'Resets controls
txtName.Focus()
resetControls()
End Sub

Private Sub disableControls()
'Disables controls on form.
txtName.Enabled = False
txtEmail.Enabled = False
rtbMessage.Enabled = False
chkSpam.Enabled = False
btnSend.Enabled = False
End Sub

Private Sub resetControls()
'Enables and clears controls and textboxes
txtName.Enabled = True
txtName.Text = ""
txtEmail.Enabled = True
txtEmail.Text = ""
rtbMessage.Enabled = True
rtbMessage.Text = ""
chkSpam.Checked = False
chkSpam.Enabled = True
btnSend.Enabled = True
End Sub

Private Sub Contact_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
'Closes form completely upon closing
Me.Dispose()
End Sub
End Class

```

Investment.vb

```
Imports System.Data.OleDb
Imports Odin.ListViewSort

Public Class Investment
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Private m_SortingColumn As ColumnHeader

    Private Sub Investment_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Runs subroutine to populate the listview and focuses the form appropriately
        populateListView()
        PictureBox.Select()
    End Sub

    Public Sub populateListView()
        'Clears and enables the listview
        lstView.Items.Clear()
        lstView.Enabled = True
        'Opens the connection and declares a new OleDbCommand and Reader
        connection.Open()
        Dim cmd As New OleDbCommand("SELECT * From Investments ORDER BY ID", connection)
        Dim dr As OleDbDataReader = cmd.ExecuteReader
        'Loops the Accounts table of the database and adds each record to a new ListViewItem
        Do While dr.Read()
            Dim newitem As New ListViewItem(dr.Item("ID").ToString)
            newitem.SubItems.Add(dr.Item("Investor").ToString)
            newitem.SubItems.Add(dr.Item("DateInvested"))
            newitem.SubItems.Add(dr.Item("InjectionAmount").ToString)
            lstView.Items.Add(newitem)
        Loop
        'Closes the connection
        connection.Close()
    End Sub

    Private Sub btnEdit_Click(sender As Object, e As EventArgs) Handles btnEdit.Click
        'Checks if an item is selected in the listview or not
        If lstView.SelectedItems.Count > 0 Then
            'Passes value of ID to Edit form and opens it
            Dim ID As String = lstView.SelectedItems(0).Text
            Investment__Edit.txtID.Text = ID
            Investment__Edit.ShowDialog()
            'Output error if no item is selected
        Else : MsgBox("Please select a record from the table.")
        End If
    End Sub
```

```

End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Checks if an item has been selected or not
    If lstView.SelectedItems.Count > 0 Then
        'Displays a MsgBox asking user to confirm to delete the record
        Dim Response = MsgBox("Are you sure you want to delete this Investment record?", MsgBoxStyle.YesNo,
            "Delete Record")
        If Response = MsgBoxResult.Yes Then
            'Checks if an item is selected in the listview or not if user response is 'Yes'
            If lstView.SelectedItems.Count > 0 Then
                'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
                Dim ID As String = lstView.SelectedItems(0).Text
                Dim cmd As New OleDbCommand("DELETE FROM Investments WHERE ID = " & ID, connection)
                'Opens connection, executes the command and closes the connection
                connection.Open()
                cmd.ExecuteNonQuery()
                connection.Close()
                'Repopulates the listview
                populateListView()
                'Updates label to show status
                lblDataStatus.Text = "Record successfully deleted."
                lblDataStatus.ForeColor = Color.Green
                lblDataStatus.Visible = True
            End If
        End If
        'Output error if no record has been selected
    Else : MsgBox("Please select a record from the table.")
    End If
End Sub

Private Sub btnAdd_Click(sender As Object, e As EventArgs) Handles btnAdd.Click
    'Runs validation function and checks if it returns true
    If Validation() Then
        'Declares the new OleDbCommand
        Dim cmd As New OleDbCommand("INSERT INTO Investments (Investor, DateInvested, InjectionAmount) " & _
            "VALUES (@Investor, @DateInvested, @InjectionAmount)", connection)
        'Gives the parameters its values
        cmd.Parameters.AddWithValue("Investor", txtInvestor.Text)
        cmd.Parameters.AddWithValue("DateInvested", dtpDate.Value.Date)
        cmd.Parameters.AddWithValue("InjectionAmount", txtAmount.Text)
        'Opens database connection, executes query and closes the database
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
    End If
End Sub

```

```

'Output to show record has been added and repopulates listview
lblAddStatus.ForeColor = Color.Green
lblAddStatus.Text = "Investment information added into database."
populateListView()
'Resets controls
txtInvestor.Text = ""
dtpDate.Value = DateTime.Today
txtAmount.Text = ""
End If
End Sub

Private Function Validation()
'Checks if all the textboxes contain text
If txtInvestor.Text <> "" And txtAmount.Text <> "" Then
    Return True
Else
    'Output to show that one or more of the textboxes have not been filled.
    lblAddStatus.ForeColor = Color.Red
    lblAddStatus.Text = "Please ensure all fields are filled out correctly."
    Return False
End If
End Function

Private Sub Investment_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
'Opens the main menu form upon closing of the investment form
MainMenu.Show()
End Sub

Private Sub txtAmount_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtAmount.KeyPress
'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
textFormatting.twoDecimalPoints(sender, e)
End Sub

Private Sub lstView_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnClickEventArgs) Handles lstView.ColumnClick
'Get the new sorting column to be sorted
Dim new_sorting_column As ColumnHeader = lstView.Columns(e.Column)
'Find out the sorting order
Dim sort_order As System.Windows.Forms.SortOrder
If m_SortingColumn Is Nothing Then
    'Sort column by ascending order
    sort_order = SortOrder.Ascending
Else
    'Checks if sorting column is the same one
    If new_sorting_column.Equals(m_SortingColumn) Then
        'If it is the same column then the sorting order is switched around

```

```

    If m_SortingColumn.Text.StartsWith("> ") Then
        sort_order = SortOrder.Descending
    Else
        sort_order = SortOrder.Ascending
    End If
Else
    'Sort by ascending value
    sort_order = SortOrder.Ascending
End If
'Removes the old sort column indicator
m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
End If
'Displays the new sorting column indicator
m_SortingColumn = new_sorting_column
If sort_order = SortOrder.Ascending Then
    m_SortingColumn.Text = "> " & m_SortingColumn.Text
Else
    m_SortingColumn.Text = "< " & m_SortingColumn.Text
End If
'Creates the comparer and sorts the listview accordingly
lstView.ListViewItemSorter = New clsListViewSorter(e.Column, sort_order)
lstView.Sort()
End Sub
End Class

```

Investment - Edit.vb

```

Imports System.Data.OleDb

Public Class Investment__Edit
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

    Private Sub Investment__Edit_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Opens the connection and declares a new OleDbCommand and Reader
        Dim ID As String = txtID.Text
        connection.Open()
        Dim dt As New DataTable()
        'Fills datatable with data from the investments table
        Using da As New OleDbDataAdapter("SELECT * FROM Investments WHERE ID = " & ID, connection)
            da.Fill(dt)
        End Using
        'Closes the connection
        connection.Close()
        'Sets values in textboxes to data in datatable
        txtInvestor.Text = dt.Rows(0)("Investor").ToString()
    End Sub

```

```

dtpDate.Value = dt.Rows(0)("DateInvested")
txtAmount.Text = dt.Rows(0)("InjectionAmount").ToString
'Focuses the form
PictureBox.Select()
End Sub

Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
    'Runs Validation function and sees if it returns true
    If Validation() Then
        'Declares the new OleDbCommand and gives it parameters
        Dim cmd As New OleDbCommand("UPDATE Investments SET Investor = ?, DateInvested = ?, InjectionAmount = ? " & _
                                    "WHERE ID = " & txtID.Text, connection)
        cmd.Parameters.AddWithValue("@p1", txtInvestor.Text)
        cmd.Parameters.AddWithValue("@p2", dtpDate.Value.Date)
        cmd.Parameters.AddWithValue("@p3", txtAmount.Text)
        'Executes the command and closes form
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Output on the main investment form to show successful updating
        Investment.lblDataStatus.Text = "Record successfully updated."
        Investment.lblDataStatus.ForeColor = Color.Green
        Investment.lblDataStatus.Visible = True
        'Closes the form
        Me.Close()
    End If
End Sub

Private Function Validation()
    'First checks if all the textboxes contain text
    If txtInvestor.Text <> "" And txtAmount.Text <> "" Then
        Return True
    Else
        'Output to show that one or more of the textboxes have not been filled.
        lblStatus.ForeColor = Color.Red
        lblStatus.Text = "Please ensure all fields are filled out correctly."
        Return False
    End If
End Function

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Displays a MsgBox asking user to confirm to close the form
    Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
    'Closes form if user selects 'Yes'
    If Response = MsgBoxResult.Yes Then

```

```

        Me.Close()
    End If
End Sub

Private Sub Investment__Edit_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Runs subroutine to repopulate the listview on the main investment form upon closing of this form
    Investment.populateListView()
    'Closes form completely
    Me.Dispose()
End Sub

Private Sub txtAmount_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtAmount.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

ListView Sorting Class (ListViewSort.vb)

```

Imports System.Collections
Public Class ListViewSort
    Public Class clsListViewSorter
        'Implements a IComparer
        Implements System.Collections.IComparer
        Private m_ColumnNumber As Integer
        Private m_SortOrder As SortOrder
        Public Sub New(ByVal column_number As Integer, ByVal sort_order As SortOrder)
            m_ColumnNumber = column_number
            m_SortOrder = sort_order
        End Sub
        'Compares the items in the appropriate column
        Public Function Compare(ByVal x As Object, ByVal y As Object) As Integer Implements IComparer.Compare
            Dim item_x As ListViewItem = DirectCast(x, ListViewItem)
            Dim item_y As ListViewItem = DirectCast(y, ListViewItem)
            'Get the subitem values.
            Dim string_x As String
            If item_x.SubItems.Count <= m_ColumnNumber Then
                string_x = ""
            Else
                string_x = item_x.SubItems(m_ColumnNumber).Text
            End If
            Dim string_y As String
            If item_y.SubItems.Count <= m_ColumnNumber Then
                string_y = ""
            Else

```

```

    string_y = item_y.SubItems(m_ColumnNumber).Text
End If
'Compares the values according to their different types
If m_SortOrder = SortOrder.Ascending Then
    If IsNumeric(string_x) And IsNumeric(string_y) Then
        Return (Val(string_x).CompareTo(Val(string_y)))
    ElseIf IsDate(string_x) And IsDate(string_y) Then
        Return (DateTime.Parse(string_x).CompareTo(DateTime.Parse(string_y)))
    Else
        Return (String.Compare(string_x, string_y))
    End If
Else
    If IsNumeric(string_x) And IsNumeric(string_y) Then
        Return (Val(string_y).CompareTo(Val(string_x)))
    ElseIf IsDate(string_x) And IsDate(string_y) Then
        Return (DateTime.Parse(string_y).CompareTo(DateTime.Parse(string_x)))
    Else
        Return (String.Compare(string_y, string_x))
    End If
End If
End Function
End Class
End Class

```

Login.vb

```

Imports System.Data.OleDb

Public Class Login
'Declares number of Attempts and database connection
Dim Attempts As Integer = 1
Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

Private Sub Login_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    'Enables controls on the form
    enableControls()
    '//THIS CODE NEEDS TO BE REMOVED ON FINAL BUILD/
    txtUsername.Text = "william.shen"
    txtPassword.Text = "password"
End Sub

Private Sub btnLogin_Click(sender As Object, e As EventArgs) Handles btnLogin.Click
    'Sets database connection string again incase database location was changed
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    'Modifies the text in the username to all lowercase for checking in the database

```

```

Dim Username As String = txtUsername.Text
Username.ToLower()
'Opens connection
connection.Open()
'Selects Username field from the Accounts table
Dim cmdUser As New OleDbCommand("SELECT Username FROM Accounts WHERE Username = @Username;", connection)
cmdUser.Parameters.AddWithValue("@Username", Username)
'Declares and initializes data reader which retrieves data under the Username field
Dim readerUsername As OleDbDataReader
readerUsername = cmdUser.ExecuteReader
'Checks if the Username column contains the text in the Username textbox
If readerUsername.HasRows = False Then
    'Sets the value of Attempt one number higher
    Attempts = Attempts + 1
    'Sets the text in the label and resets the Password textbox
    lblStatus.Visible = True
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "The username or password you have entered is incorrect." & vbCrLf & "Attempt " & Attempts & " of 5"
    txtPassword.Text = ""
    'Closes data reader and command
    readerUsername.Close()
    cmdUser.Dispose()
Else
    'In the case if the Username exists in the database
    readerUsername.Close()
    'Declares and initializes command and reader
    Dim cmdPassword As New OleDbCommand("SELECT * FROM Accounts WHERE Username = @Username;", connection)
    cmdPassword.Parameters.AddWithValue("@Username", txtUsername.Text)
    Dim readerPassword As OleDbDataReader
    readerPassword = cmdPassword.ExecuteReader
    'Reads the database to see if the Username corresponds with the text in the Password textbox
    If readerPassword.Read() Then
        If readerPassword("Password").ToString = txtPassword.Text Then
            connection.Close()
            'Runs subroutine to check if user has 'Admin' permission
            adminCheck()
            'Resets form and starts timer to open main menu form
            disableControls()
            'Output to show correct username and password
            lblStatus.Text = "Correct password entered. The system will now login."
            lblStatus.ForeColor = Color.Green
            lblStatus.Visible = True
            Feedback.Start()
        Else
            'Sets the value of Attempt one number higher

```

```

Attempts = Attempts + 1
'Sets the text in the label and resets the Password textbox
lblStatus.Visible = True
lblStatus.ForeColor = Color.Red
lblStatus.Text = "The username or password you have entered is incorrect." & vbCrLf & "Attempt " & Attempts & " of 5"
txtPassword.Text = ""
End If
End If
'Closes reader and data reader
readerPassword.Close()
End If
'Closes the connection
connection.Close()
'Checks if number of attempts exceeds 5. If so then the program will close
If Attempts >= 6 Then
    MsgBox("Maximum number of attempts reached. Program will exit.")
    Application.Exit()
End If
End Sub

Private Sub Feedback_Tick(sender As Object, e As EventArgs) Handles Feedback.Tick
'Resets form
lblStatus.Text = ""
txtUsername.Text = ""
txtPassword.Text = ""
Attempts = 1
'Opens Menu form and closes login form with a 1 second delay
MainMenu.Show()
Me.Close()
End Sub

Private Sub txtPassword_KeyDown(ByVal sender As Object, ByVal e As KeyEventArgs) Handles txtPassword.KeyDown
'Calls subroutine of the login button when enter key is pressed in the Password textbox
If e.KeyCode = Keys.Enter Then
    Call btnLogin_Click(sender, e)
End If
End Sub

Private Sub txtUsername_KeyDown(ByVal sender As Object, ByVal e As KeyEventArgs) Handles txtUsername.KeyDown
'Calls subroutine of the login button when enter key is pressed in the Password textbox
If e.KeyCode = Keys.Enter Then
    Call btnLogin_Click(sender, e)
End If
End Sub

```

```

Private Sub adminCheck()
    'Sets database connection string again incase database location was changed
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    'Converts Username string to lowercase for database checking
    Dim Username As String = txtUsername.Text
    Username.ToLower()
    My.Settings.accountUsername = Username
    'Opens connection
    connection.Open()
    'Selects Admin field from the Accounts table
    Dim cmdAdmin As New OleDbCommand("SELECT FullName, Admin FROM Accounts WHERE Username = @Username;", connection)
    cmdAdmin.Parameters.AddWithValue("@Username", Username)
    'Declares data reader and executes it
    Dim dr As OleDbDataReader = cmdAdmin.ExecuteReader
    dr.Read()
    'Sets settings variable for account FullName
    My.Settings.accountFullName = dr.Item("FullName")
    'Checks if the Admin field is checked or not
    If dr.Item("Admin") = True Then
        My.Settings.accountAdmin = True
    Else : My.Settings.accountAdmin = False
    End If
    connection.Close()
End Sub

Private Sub disableControls()
    'Disables controls
    txtUsername.Enabled = False
    txtPassword.Enabled = False
    btnLogin.Enabled = False
    ToolStrip.Enabled = False
End Sub

Private Sub enableControls()
    'Enables controls
    txtUsername.Enabled = True
    txtPassword.Enabled = True
    btnLogin.Enabled = True
    ToolStrip.Enabled = True
End Sub

Private Sub AccountManagerToolStripMenuItem_Click_1(sender As Object, e As EventArgs) Handles AccountManagerToolStripMenuItem.Click
    'Opens the Account Manager
    AccountManager.ShowDialog()
End Sub

```

```

Private Sub ExitToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ExitToolStripMenuItem.Click
    'Exits the program
    Application.Exit()
End Sub
Private Sub ForgotToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ForgotToolStripMenuItem.Click
    'Output to display help to user
    MsgBox("Please contact your Administrator who will be able to give you your Username and Password if necessary." & vbCrLf &
           vbCrLf & "Use the Account Manager if you know the Master Password to view account information. (File -> Account Manager)")
End Sub
Private Sub ContactToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ContactToolStripMenuItem.Click
    'Opens and brings contact form to the front
    Contact.Show()
    Contact.BringToFront()
End Sub
Private Sub AboutToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles AboutToolStripMenuItem.Click
    'Opens the About form
    About.ShowDialog()
End Sub
Private Sub SettingsToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles SettingsToolStripMenuItem.Click
    'Opens the settings form
    Settings.ShowDialog()
End Sub
End Class

```

MainMenu.vb

```

Public Class MainMenu
    Private Sub MainMenu_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Shows the date and time on the labels
        lblTime.Text = DateTime.Now.ToString("dddd dd MMMM yyyy")
        lblDate.Text = DateTime.Now.ToString("hh:mm:ss tt")
    End Sub

    Private Sub DateTimer_Tick(sender As Object, e As EventArgs) Handles DateTimer.Tick
        'Code to update the date and time shown on labels
        lblTime.Text = DateTime.Now.ToString("dddd dd MMMM yyyy")
        lblDate.Text = DateTime.Now.ToString("hh:mm:ss tt")
    End Sub

    Private Sub btnExit_Click(sender As Object, e As EventArgs) Handles btnExit.Click
        'Displays a MsgBox asking user to confirm to exit the application
        Dim Response = MsgBox("Are you sure you want to exit?", MsgBoxStyle.YesNo, "Exit")
        If Response = MsgBoxResult.Yes Then
            Application.Exit()
        End If
    End Sub

```

```

End Sub

Private Sub btnLogOut_Click(sender As Object, e As EventArgs) Handles btnLogOut.Click
    'Closes all the forms if they are open
    Travel_Application__Overview.Close()
    Business_Expense_Application__Overview.Close()
    Revenues__Overview.Close()
    Travel_Expense__Overview.Close()
    Business_Expense__Overview.Close()
    Investment.Close()
    Wages__Overview.Close()
    Settings.Close()
    Contact.Close()
    'Shows the Login form
    Login.Show()
    'Closes Main Menu Form
    Me.Close()
End Sub

Private Sub btnHelp_Click(sender As Object, e As EventArgs) Handles btnHelp.Click
    'Opens and brings to front the Contact form
    Contact.Show()
    Contact.BringToFront()
End Sub

'Code for the buttons on the main menu form to open the relevant forms and bring them to the front/ show as dialog
Private Sub btnTravelApplication_Click(sender As Object, e As EventArgs) Handles btnTravelApplication.Click
    Travel_Application__Overview.Show()
    Travel_Application__Overview.BringToFront()
End Sub
Private Sub btnBusinessApplication_Click(sender As Object, e As EventArgs) Handles btnBusinessApplication.Click
    Business_Expense_Application__Overview.Show()
    Business_Expense_Application__Overview.BringToFront()
End Sub
Private Sub btnRevenues_Click(sender As Object, e As EventArgs) Handles btnRevenues.Click
    Revenues__Overview.Show()
    Revenues__Overview.BringToFront()
End Sub
Private Sub btnTravelExpenses_Click(sender As Object, e As EventArgs) Handles btnTravelExpenses.Click
    Travel_Expense__Overview.Show()
End Sub
Private Sub btnBusinessExpenses_Click(sender As Object, e As EventArgs) Handles btnBusinessExpenses.Click
    Business_Expense__Overview.Show()
    Business_Expense__Overview.BringToFront()
End Sub

```

```

Private Sub btnInvestment_Click(sender As Object, e As EventArgs) Handles btnInvestment.Click
    Investment.Show()
    Investment.BringToFront()
End Sub
Private Sub btnWages_Click(sender As Object, e As EventArgs) Handles btnWages.Click
    Wages_Overview.Show()
    Wages_Overview.BringToFront()
End Sub
Private Sub btnAccountManager_Click(sender As Object, e As EventArgs) Handles btnAccountManager.Click
    AccountManager.ShowDialog()
End Sub
Private Sub btnSettings_Click(sender As Object, e As EventArgs) Handles btnSettings.Click
    Settings.ShowDialog()
End Sub

'Opens relevant forms and brings them to the front or shows them as a dialog as appropriate
Private Sub AccountManagerToolStripMenuItem_Click_1(sender As Object, e As EventArgs) Handles AccountManagerToolStripMenuItem.Click
    AccountManager.ShowDialog()
End Sub
Private Sub LogoutToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles LogoutToolStripMenuItem.Click
    Login.Show()
    Me.Close()
End Sub
Private Sub ExitToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ExitToolStripMenuItem.Click
    'Displays a MsgBox asking user to confirm to exit the application
    Dim Response = MsgBox("Are you sure you want to exit?", MsgBoxStyle.YesNo, "Exit")
    If Response = MsgBoxResult.Yes Then
        Application.Exit()
    End If
End Sub
Private Sub ContactToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ContactToolStripMenuItem.Click
    Contact.Show()
    Contact.BringToFront()
End Sub
Private Sub AboutToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles AboutToolStripMenuItem.Click
    About.ShowDialog()
End Sub
Private Sub SettingsToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles SettingsToolStripMenuItem.Click
    Settings.ShowDialog()
End Sub
Private Sub MainMenuToolStripMenuItem1_Click(sender As Object, e As EventArgs) Handles MainMenuToolStripMenuItem1.Click
    Me.Show()
End Sub
Private Sub AccountManagerToolStripMenuItem1_Click(sender As Object, e As EventArgs) Handles AccountManagerToolStripMenuItem1.Click
    AccountManager.ShowDialog()

```

```

End Sub
Private Sub SettingsToolStripMenuItem1_Click(sender As Object, e As EventArgs) Handles SettingsToolStripMenuItem1.Click
    Settings.Show()
    Settings.BringToFront()
End Sub
Private Sub ContactSupportToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ContactSupportToolStripMenuItem.Click
    Contact.Show()
    Contact.BringToFront()
End Sub
Private Sub TravelApplicationToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles TravelApplicationToolStripMenuItem.Click
    Travel_Application__Overview.Show()
    Travel_Application__Overview.BringToFront()
End Sub
Private Sub BusinessExpenseApplicationToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles BusinessExpenseApplicationToolStripMenuItem.Click
    Business_Expense_Application__Overview.Show()
    Business_Expense_Application__Overview.BringToFront()
End Sub
Private Sub TravelExpensesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles TravelExpensesToolStripMenuItem.Click
    Travel_Expense__Overview.Show()
    Travel_Expense__Overview.BringToFront()
End Sub
Private Sub BusinessExpensesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles BusinessExpensesToolStripMenuItem.Click
    Business_Expense__Overview.Show()
    Business_Expense__Overview.BringToFront()
End Sub
Private Sub RevenuesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles RevenuesToolStripMenuItem.Click
    Revenues__Overview.Show()
    Revenues__Overview.BringToFront()
End Sub
Private Sub InvestmentToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles InvestmentToolStripMenuItem.Click
    Investment.Show()
    Investment.BringToFront()
End Sub
Private Sub WagesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles WagesToolStripMenuItem.Click
    Wages__Overview.Show()
    Wages__Overview.BringToFront()
End Sub
End Class

```

Revenues - Overview.vb

```

Imports System.Data.OleDb
Imports Odin.ListViewSort

```

```

Public Class Revenues__Overview
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Private m_SortingColumn As ColumnHeader

    Private Sub Revenues__Overview_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Runs subroutine to populate the listview
        populateListView()
        'Focuses the form
        PictureBox.Select()
        'Sets the minimum date of the DateTo date picker to the value of the DateFrom date picker
        dtpDateTo.MinDate = dtpDateFrom.Value.Date
    End Sub

    Public Sub populateListView()
        'Clears and enables the listview
        lstView.Items.Clear()
        lstView.Enabled = True
        'Opens the connection and declares a new OleDbCommand and Reader
        connection.Open()
        Dim cmd As New OleDbCommand("SELECT * From Revenues ORDER BY InvoiceDate", connection)
        Dim dr As OleDbDataReader = cmd.ExecuteReader
        'Loops the Accounts table of the database and adds each record to a new ListViewItem
        Do While dr.Read()
            Dim newitem As New ListViewItem(dr.Item("PONumber").ToString)
            newitem.SubItems.Add(dr.Item("InvoiceNumber").ToString)
            newitem.SubItems.Add(dr.Item("InvoiceDate"))
            newitem.SubItems.Add(dr.Item("Customer").ToString)
            newitem.SubItems.Add(dr.Item("Total"))
            lstView.Items.Add(newitem)
        Loop
        'Closes the connection
        connection.Close()
    End Sub

    Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
        'Checks if an item is selected in the listview or not
        If lstView.SelectedItems.Count > 0 Then
            'Displays a MsgBox asking user to confirm to delete the record
            Dim Response = MsgBox("Are you sure you want to delete this Revenue record?", MsgBoxStyle.YesNo,
                "Delete Record")
            If Response = MsgBoxResult.Yes Then
                'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
                Dim PONumber As String = lstView.SelectedItems(0).Text
                Dim cmd As New OleDbCommand("DELETE FROM Revenues WHERE PONumber = '" & PONumber & "'", connection)
                'Opens connection, executes the command and closes the connection
            End If
        End If
    End Sub

```

```

connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Repopulates the listview
populateListView()
End If
'Outputs error if user has not selected an item in the listview
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles btnNew.Click
'Shows the form for a new revenue record
Revenues__New.ShowDialog()
End Sub

Private Sub btnEdit_Click(sender As Object, e As EventArgs) Handles btnEdit.Click
'Checks if an item is selected in the listview or not
If lstView.SelectedItems.Count > 0 Then
    'Passes value of ID to Edit form
    Dim PONumber As String = lstView.SelectedItems(0).Text
    Revenues__View_and_Edit.txtPONumber.Text = PONumber
    Revenues__View_and_Edit.ShowDialog()
    'Output error if no item is selected
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub Revenues__Overview_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
'Opens the main menu form upon closing of this form
MainMenu.Show()
End Sub

Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles btnSearch.Click
'Opens SearchBox form and passes appropriate text in textboxes
SearchBox.lblStatus.Text = "Please enter the PO Number."
SearchBox.lblFunction.Text = "Revenues"
SearchBox.ShowDialog()
End Sub

Private Sub btnFilter_Click(sender As Object, e As EventArgs) Handles btnFilter.Click
'Checks if the filterdate checkbox is checked or not
If chkFilterDate.Checked = True Then
    'Clears the listview
    lstView.Items.Clear()

```

```

'Opens connection, declares OleDbCommand and OleDbReader and executes
connection.Open()
Dim cmd As New OleDbCommand("SELECT * From Revenues WHERE InvoiceDate BETWEEN @DateFrom AND @DateTo " & _
                           "ORDER BY InvoiceDate", connection)
cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
Dim dr As OleDbDataReader = cmd.ExecuteReader
'Loops the Revenues table of the database and adds each record to a new ListViewItem
Do While dr.Read()
    Dim newitem As New ListViewItem(dr.Item("PONumber").ToString)
    newitem.SubItems.Add(dr.Item("InvoiceNumber").ToString)
    newitem.SubItems.Add(dr.Item("InvoiceDate"))
    newitem.SubItems.Add(dr.Item("Customer").ToString)
    newitem.SubItems.Add(dr.Item("Total"))
    lstView.Items.Add(newitem)
Loop
'Closes the connection
connection.Close()
'Checks if any data was found
If lstView.Items.Count = 0 Then
    'Output to show that no data was found
    lblFilterStatus.Text = "No data found."
    lblFilterStatus.ForeColor = Color.Red
    lblFilterStatus.Visible = True
    populateListView()
ElseIf lstView.Items.Count <> 0 Then
    'Output to show that data was successfully filtered
    lblFilterStatus.Text = "Data successfully filtered."
    lblFilterStatus.ForeColor = Color.Green
    lblFilterStatus.Visible = True
End If
Else
    'Feedback to show that the checkbox was not checked
    lblFilterStatus.Text = "Please select the filter."
    lblFilterStatus.ForeColor = Color.Red
    lblFilterStatus.Visible = True
End If
End Sub

Private Sub btnFilterClear_Click(sender As Object, e As EventArgs) Handles btnFilterClear.Click
    'Runs subroutine to clear and populate the listview
    populateListView()
    chkFilterDate.Checked = False
    dtpDateFrom.Value = DateTime.Today
    dtpDateTo.Value = DateTime.Today

```

```

lblFilterStatus.Visible = False
End Sub

Private Sub dtpDateFrom_ValueChanged(sender As Object, e As EventArgs) Handles dtpDateFrom.ValueChanged
    'Sets the minimum DateTo datepicker to the value of the DateFrom datepicker
    dtpDateTo.MinDate = dtpDateFrom.Value.Date
End Sub

Private Sub lstView_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnClickEventArgs) Handles lstView.ColumnClick
    'Get the new sorting column to be sorted
    Dim new_sorting_column As ColumnHeader = lstView.Columns(e.Column)
    'Find out the sorting order
    Dim sort_order As System.Windows.Forms.SortOrder
    If m_SortingColumn Is Nothing Then
        'Sort column by ascending order
        sort_order = SortOrder.Ascending
    Else
        'Checks if sorting column is the same one
        If new_sorting_column.Equals(m_SortingColumn) Then
            'If it is the same column then the sorting order is switched around
            If m_SortingColumn.Text.StartsWith("> ") Then
                sort_order = SortOrder.Descending
            Else
                sort_order = SortOrder.Ascending
            End If
        Else
            'Sort by ascending value
            sort_order = SortOrder.Ascending
        End If
        'Removes the old sort column indicator
        m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
    End If
    'Displays the new sorting column indicator
    m_SortingColumn = new_sorting_column
    If sort_order = SortOrder.Ascending Then
        m_SortingColumn.Text = "> " & m_SortingColumn.Text
    Else
        m_SortingColumn.Text = "< " & m_SortingColumn.Text
    End If
    'Creates the comparer and sorts the listview accordingly
    lstView.ListViewItemSorter = New clsListviewSorter(e.Column, sort_order)
    lstView.Sort()
End Sub
End Class

```

Revenues - New.vb

```
Imports System.Data.OleDb

Public Class Revenues__New
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

    Private Sub Revenues__New_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Focuses the PO Number textbox
        txtPONumber.Select()
    End Sub

    Private Sub btnAdd_Click(sender As Object, e As EventArgs) Handles btnAdd.Click
        'Runs Validation function
        If Validation() Then
            'Try to add the new record into the Revenues table
            Try
                'Declares the new OleDbCommand
                Dim cmd As New OleDbCommand("INSERT INTO Revenues (PONumber, InvoiceNumber, InvoiceDate, Customer, Subtotal, " & _
                    "GST, Total) VALUES (@PONumber, @InvoiceNumber, @InvoiceDate, @Customer, " & _
                    "@Subtotal, @GST, @Total)", connection)
                'Gives the parameters its values
                cmd.Parameters.AddWithValue("PONumber", txtPONumber.Text)
                cmd.Parameters.AddWithValue("InvoiceNumber", txtInvoiceNumber.Text)
                cmd.Parameters.AddWithValue("InvoiceDate", dtpDate.Value.Date)
                cmd.Parameters.AddWithValue("Customer", txtCustomer.Text)
                cmd.Parameters.AddWithValue("Subtotal", txtSubtotal.Text)
                cmd.Parameters.AddWithValue("GST", txtGST.Text)
                cmd.Parameters.AddWithValue("Total", txtTotal.Text)
                'Opens database connection, executes query and closes the database
                connection.Open()
                cmd.ExecuteNonQuery()
                connection.Close()
                'Feedback to show successful insertion
                lblStatus.ForeColor = Color.Green
                lblStatus.Text = "Revenue successfully saved."
                'Output to show successful insert on Overview form and closes form completely
                Revenues__Overview.lblDataStatus.Text = "Record added."
                Revenues__Overview.lblDataStatus.Visible = True
                Me.Dispose()
            Catch
                'In the case where the OleDbCommand could not be executed, closes the connection
                connection.Close()
                'Output to show there is an error
                lblStatus.ForeColor = Color.Red
            End Try
        End If
    End Sub
```

```

        lblStatus.Text = "A record for the PO Number already exists."
    End Try
End If
End Sub

Private Function Validation()
    'Checks if textboxes all contain text
    If txtPONumber.Text <> "" And txtInvoiceNumber.Text <> "" And txtCustomer.Text <> "" And _
        txtSubtotal.Text <> "" And txtGST.Text <> "" And txtTotal.Text <> "" Then
        Return True
    Else
        'Output to show that one or more of the textboxes have not been filled.
        lblStatus.ForeColor = Color.Red
        lblStatus.Text = "Please ensure all fields are filled out correctly."
        Return False
    End If
End Function

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Displays a MsgBox asking user to confirm to close the form
    Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
    'Closes form completely if user clicks 'Yes'
    If Response = MsgBoxResult.Yes Then
        Me.Dispose()
    End If
End Sub

Private Sub Revenues__New_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Closes form completely and runs subroutine to populate the listview on the overview form
    Me.Dispose()
    Revenues__Overview.populateListView()
End Sub

Private Sub Calculation_TextChanged(sender As Object, e As EventArgs) Handles txtSubtotal.TextChanged, txtGST.TextChanged
    'Declares and gives values to the variables from their relevant textboxes
    Dim subtotal, GST, total As Double
    subtotal = Val(txtSubtotal.Text)
    GST = Val(txtGST.Text)
    'Adds value in Subtotal textbox and GST to give the total
    total = subtotal + GST
    txtTotal.Text = total
End Sub

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtSubtotal.KeyPress,
    txtGST.KeyPress, txtTotal.KeyPress

```

```

'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

Revenues - View and Edit.vb

```

Imports System.Data.OleDb

Public Class Revenues__View_and_Edit
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

    Private Sub Revenues__View_and_Edit_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Opens the connection and declares a new OleDbCommand and Reader
        Dim PONumber As String = txtPONumber.Text
        connection.Open()
        Dim dt As New DataTable()
        'Fills datatable with data from the Revenues table
        Using da As New OleDbDataAdapter("SELECT * FROM Revenues WHERE PONumber = '" & PONumber & "'", connection)
            da.Fill(dt)
        End Using
        'Closes the connection
        connection.Close()
        'Sets values in textboxes and date time pickers to data in datatable
        txtInvoiceNumber.Text = dt.Rows(0)("InvoiceNumber").ToString
        dtpDate.Value = dt.Rows(0)("InvoiceDate")
        txtCustomer.Text = dt.Rows(0)("Customer").ToString
        txtSubtotal.Text = dt.Rows(0)("Subtotal")
        txtGST.Text = dt.Rows(0)("GST")
        txtTotal.Text = dt.Rows(0)("Total")
        PictureBox.Select()
    End Sub

    Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
        'Runs Validation function
        If Validation() Then
            'Declares the new OleDbCommand and gives it parameters
            Dim cmd As New OleDbCommand("UPDATE Revenues SET InvoiceNumber = ?, InvoiceDate = ?, Customer = ?, Subtotal = ?, " & _
                "GST = ?, Total = ? WHERE PONumber = '" & txtPONumber.Text & "'", connection)
            cmd.Parameters.AddWithValue("@p1", txtInvoiceNumber.Text)
            cmd.Parameters.AddWithValue("@p2", dtpDate.Value.Date)
            cmd.Parameters.AddWithValue("@p3", txtCustomer.Text)
            cmd.Parameters.AddWithValue("@p4", txtSubtotal.Text)
            cmd.Parameters.AddWithValue("@p5", txtGST.Text)
            cmd.Parameters.AddWithValue("@p6", txtTotal.Text)
        End If
    End Sub

```

```

'Executes the command and closes form
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Output to show successful insert on Overview form
Revenues__Overview.lblDataStatus.Text = "Record updated."
Revenues__Overview.lblDataStatus.Visible = True
'Closes form
Me.Close()
End If
End Sub

Private Function Validation()
'Check if textboxes all contain text
If txtPONumber.Text <> "" And txtInvoiceNumber.Text <> "" And txtCustomer.Text <> "" And _
txtSubtotal.Text <> "" And txtGST.Text <> "" And txtTotal.Text <> "" Then
    Return True
Else
    'Output to show that one or more of the textboxes have not been filled.
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "Please ensure all fields are filled out correctly."
    Return False
End If
End Function

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
'Displays MsgBox asking user to select Yes or No
Dim Response = MsgBox("Are you sure you want to delete this Business Expense Application?", MsgBoxStyle.YesNo, "Delete Application")
If Response = MsgBoxResult.Yes Then
    'Declares a new OleDbCommand which deletes the record if response is 'Yes'
    Dim PONumber As String = txtPONumber.Text
    Dim cmd As New OleDbCommand("DELETE FROM Revenues WHERE PONumber = '" & PONumber & "'", connection)
    'Opens connection, executes the command and closes the connection
    connection.Open()
    cmd.ExecuteNonQuery()
    connection.Close()
    'Output to show successful insert on Overview form
    Revenues__Overview.lblDataStatus.Text = "Record deleted."
    Revenues__Overview.lblDataStatus.Visible = True
    'Closes form
    Me.Close()
End If
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click

```

```

'Displays a MsgBox asking user to confirm to close the form
Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
'Closes form if user responds 'Yes'
If Response = MsgBoxResult.Yes Then
    Me.Close()
End If
End Sub

Private Sub Revenues__View_and_Edit_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Runs subroutine on overview form to repopulate the listview
    Revenues__Overview.populateListView()
    'Closes form completely
    Me.Dispose()
End Sub

Private Sub Calculation_TextChanged(sender As Object, e As EventArgs) Handles txtSubtotal.TextChanged, txtGST.TextChanged
    'Declares variables and assigns them values based on their relevant textboxes
    Dim subtotal, GST, total As Double
    subtotal = Val(txtSubtotal.Text)
    GST = Val(txtGST.Text)
    'Adds value in Subtotal textbox and GST to give the total
    total = subtotal + GST
    txtTotal.Text = total
End Sub

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtSubtotal.KeyPress,
    txtGST.KeyPress, txtTotal.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

Search Box Form (SearchBox.vb)

```

Imports System.Data.OleDb

Public Class SearchBox
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

    Private Sub SearchBox_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Focuses on textbox control
    End Sub

```

```

txtSearch.Select()
End Sub

Private Sub txtSearch_KeyDown(ByVal sender As Object, ByVal e As KeyEventArgs) Handles txtSearch.KeyDown
    'Calls subroutine of the search button when enter key is pressed in the search textbox
    If e.KeyCode = Keys.Enter Then
        Call btnSearch_Click(sender, e)
    End If
End Sub

Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles btnSearch.Click
    'Checks which search function has been passed over by the overview form and runs appropriate subroutine
    If lblFunction.Text = "TEA" Then
        TEA()
    ElseIf lblFunction.Text = "BEA" Then
        BEA()
    ElseIf lblFunction.Text = "TravelExpense" Then
        TravelExpense()
    ElseIf lblFunction.Text = "BusinessExpense" Then
        BusinessExpense()
    ElseIf lblFunction.Text = "Revenues" Then
        Revenues()
    End If
End Sub

Private Sub btnCancel_Click(sender As Object, e As EventArgs) Handles btnCancel.Click
    'Completely closes the form
    Me.Dispose()
End Sub

Private Sub TEA()
    Dim Search As String = txtSearch.Text
    'Selects TrackingNumber field from the Accounts table
    Dim cmd As New OleDbCommand("SELECT TrackingNumber FROM TEA WHERE TrackingNumber = @TrackingNumber;", connection)
    cmd.Parameters.AddWithValue("@TrackingNumber", Search)
    'Declares and initializes data reader which retrieves data under the TrackingNumber field
    Dim dr As OleDbDataReader
    connection.Open()
    dr = cmd.ExecuteReader
    'Checks if the TrackingNumber column contains the text in the Search textbox
    If dr.HasRows = False Then
        'Sets labels in StatusBox form for output to show no tracking number exists
        StatusBox.lblStatus.Text = "Record not found."
        StatusBox.lblDescription.Text = "A record for the tracking number does not exist in the database."
        StatusBox.ShowDialog()
    End If
End Sub

```

```

'Closes data reader and command
dr.Close()
cmd.Dispose()
'Closes the connection and form
connection.Close()
Me.Dispose()

Else
    'In the case if the TrackingNumber exists in the database. Closes data reader
    dr.Close()
    'Sets the tracking number on the view and edit form and opens it
    Travel_Application__View_and_Edit.lblTracking.Text = Search
    Travel_Application__View_and_Edit.ShowDialog()
    'Closes the connection and form
    connection.Close()
    Me.Dispose()
End If
End Sub

Private Sub BEA()
    Dim Search As String = txtSearch.Text
    'Selects TrackingNumber field from the Accounts table
    Dim cmd As New OleDbCommand("SELECT TrackingNumber FROM BEA WHERE TrackingNumber = @TrackingNumber;", connection)
    cmd.Parameters.AddWithValue("@TrackingNumber", Search)
    'Declares and initializes data reader which retrieves data under the TrackingNumber field
    Dim dr As OleDbDataReader
    connection.Open()
    dr = cmd.ExecuteReader()
    'Checks if the TrackingNumber column contains the text in the Search textbox
    If dr.HasRows = False Then
        'Sets labels in StatusBox form for output to show no tracking number exists
        StatusBox.lblStatus.Text = "Record not found."
        StatusBox.lblDescription.Text = "A record for the tracking number does not exist in the database."
        StatusBox.ShowDialog()
        'Closes data reader, command, connection and form
        dr.Close()
        cmd.Dispose()
        connection.Close()
        Me.Dispose()
    Else
        'In the case if the TrackingNumber existed in the database closes datareader
        dr.Close()
        'Sets tracking number in the view and edit form and opens it
        Business_Expense_Applications__View_and_Edit.lblTracking.Text = Search
        Business_Expense_Applications__View_and_Edit.ShowDialog()
        'Closes connection and form
    End If
End Sub

```

```

        connection.Close()
        Me.Dispose()
    End If
End Sub

Private Sub TravelExpense()
    Dim Search As String = txtSearch.Text
    'Selects TrackingNumber field from the Accounts table
    Dim cmd As New OleDbCommand("SELECT TrackingNumber FROM [Travel Expenses] WHERE TrackingNumber = @TrackingNumber;", connection)
    cmd.Parameters.AddWithValue("@TrackingNumber", Search)
    'Declares and initializes data reader which retrieves data under the TrackingNumber field
    Dim dr As OleDbDataReader
    connection.Open()
    dr = cmd.ExecuteReader
    'Checks if the TrackingNumber column contains the text in the Search textbox
    If dr.HasRows = False Then
        'Sets labels in StatusBox form for output to show no tracking number exists
        StatusBox.lblStatus.Text = "Record not found."
        StatusBox.lblDescription.Text = "A record for the tracking number does not exist in the database."
        StatusBox.ShowDialog()
        'Closes data reader, command, connection and form
        dr.Close()
        cmd.Dispose()
        connection.Close()
        Me.Dispose()
    Else
        'In the case if the TrackingNumber exists in the database closes datareader
        dr.Close()
        'Sets tracking number in view and edit form and opens it
        Travel_Expense.txtTrackingNumber.Text = Search
        Travel_Expense.txtTrackingNumber.ReadOnly = True
        Travel_Expense.ShowDialog()
        'Closes connection and form
        connection.Close()
        Me.Dispose()
    End If
End Sub

Private Sub BusinessExpense()
    Dim Search As String = txtSearch.Text
    'Selects TrackingNumber field from the Accounts table
    Dim cmd As New OleDbCommand("SELECT TrackingNumber FROM [Business Expenses] WHERE TrackingNumber = @TrackingNumber;", connection)
    cmd.Parameters.AddWithValue("@TrackingNumber", Search)
    'Declares and initializes data reader which retrieves data under the TrackingNumber field
    Dim dr As OleDbDataReader

```

```

connection.Open()
dr = cmd.ExecuteReader
'Checks if the TrackingNumber column contains the text in the Search textbox
If dr.HasRows = False Then
    'Sets labels in StatusBox form for output to show no tracking number exists
    StatusBox.lblStatus.Text = "Record not found."
    StatusBox.lblDescription.Text = "A record for the tracking number does not exist in the database."
    StatusBox.ShowDialog()
    'Closes data reader, command, connection and form
    dr.Close()
    cmd.Dispose()
    connection.Close()
    Me.Dispose()
Else
    'In the case if the TrackingNumber exists in the database closes datareader
    dr.Close()
    'Sets tracking number in view and edit form and opens it
    Business_Expense__View_and_Edit.txtTrackingNumber.Text = Search
    Business_Expense__View_and_Edit.ShowDialog()
    'Closes connection and form
    connection.Close()
    Me.Dispose()
End If
End Sub

Private Sub Revenues()
Dim Search As String = txtSearch.Text
'Selects TrackingNumber field from the Accounts table
Dim cmd As New OleDbCommand("SELECT PONumber FROM Revenues WHERE PONumber = @PONumber;", connection)
cmd.Parameters.AddWithValue("@PONumber", Search)
'Declares and initializes data reader which retrieves data under the TrackingNumber field
Dim dr As OleDbDataReader
connection.Open()
dr = cmd.ExecuteReader
'Checks if the PONumber column contains the text in the Search textbox
If dr.HasRows = False Then
    'Sets labels in StatusBox form for output to show no tracking number exists
    StatusBox.lblStatus.Text = "Record not found."
    StatusBox.lblDescription.Text = "A record for PO number does not exist in the database."
    StatusBox.ShowDialog()
    'Closes data reader, command, connection and form
    dr.Close()
    cmd.Dispose()
    connection.Close()
    Me.Dispose()

```

```

Else
    'In the case if the PONumber exists in the database closes datareader
    dr.Close()
    'Sets PONumber in view and edit form and opens it
    Revenues__View_and_Edit.txtPONumber.Text = Search
    Revenues__View_and_Edit.ShowDialog()
    'Closes connection and form
    connection.Close()
    Me.Dispose()
End If
End Sub
End Class

```

User Settings (Settings.vb)

```

Imports System.Net.Mail

Public Class Settings

    Private Sub Settings_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Sets the text in the labels and textbox
        lblLocationDirectory.Text = My.Settings.dbLocation
        lblBackup.Text = "Last backed up on " & My.Settings.backupDate & "."
        txtApplicationEmail.Text = My.Settings.applicationEmail
        'Focuses form
        PictureBox.Select()
    End Sub

    Private Sub btnLocation_Click(sender As Object, e As EventArgs) Handles btnLocation.Click
        'Declares an OpenFileDialog and assigns it a title
        Dim OpenFD As OpenFileDialog = New OpenFileDialog
        OpenFD.Title = "Select your Database File"
        'Opens file dialog and checks if OK button was pressed or not
        If OpenFD.ShowDialog = Windows.Forms.DialogResult.OK Then
            'Sets the settings variable for the database location to the file the user selected
            My.Settings.dbLocation = OpenFD.FileName
            'Output to user showing success
            lblLocationDirectory.Text = My.Settings.dbLocation
            lblLocationStatus.Visible = True
            lblLocationStatus.ForeColor = Color.ForestGreen
            lblLocationStatus.Text = "Database Location successfully updated."
            '///DO THIS///'
            'Displays a MsgBox asking user whether to restart the program
            Dim Response = MsgBox("The program must restart for changes to take full effect. Restart now?", MsgBoxStyle.YesNo, "Restart Odin")
        End If
    End Sub

```

```

If Response = MsgBoxResult.Yes Then
    'Restarts application if user selects yes
    Application.Restart()
Else : MsgBox("Please ensure you restart the program.")
End If
Else
    'Output to user showing the database was not updated
    lblLocationStatus.Visible = True
    lblLocationStatus.ForeColor = Color.Red
    lblLocationStatus.Text = "Database Location was not updated."
End If
End Sub

Private Sub btnBackup_Click(sender As Object, e As EventArgs) Handles btnBackup.Click
    'Declares FolderBrowserDialog and gives it a description.
    Dim FolderBD As FolderBrowserDialog = New FolderBrowserDialog
    FolderBD.Description = "Select the directory where you want the database backed up to."
    'Opens the folder browser dialog and checks if OK button was pressed or not
    If FolderBD.ShowDialog = Windows.Forms.DialogResult.OK Then
        'Declares the directory and file name for the new file
        Dim Directory As String = FolderBD.SelectedPath & "\OdinHealth Database Backup (" & DateTime.Now.ToString("dd-MM-yyyy") & ").accdb"
        'Copies the file from current database location to user selected directory
        My.Computer.FileSystem.CopyFile(My.Settings.dbLocation, Directory, _
                                        Microsoft.VisualBasic.FileIO.UIOption.AllDialogs, _
                                        Microsoft.VisualBasic.FileIO.UICancelOption.DoNothing)
        'Output to show success
        lblBackup.ForeColor = Color.ForestGreen
        lblBackup.Text = "Database backed up successfully."
        My.Settings.backupDate = DateTime.Today
    Else
        'Output to show database was not backed up
        lblBackup.Visible = True
        lblBackup.ForeColor = Color.Red
        lblBackup.Text = "Database was not backed up."
    End If
End Sub

Private Sub btnHelp_Click(sender As Object, e As EventArgs) Handles btnHelp.Click
    'Gives the user help instructions
    MsgBox("To change the directory of the database, click the 'Browse' button and then go to the " &
          "directory of the database, select it and press 'Open'." & vbCrLf & vbCrLf & "To backup the database " &
          "click the 'Backup' button and navigate to the directory where you want the backup to be saved.")
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click

```

```

'Closes the form completely
Me.Dispose()
End Sub

Private Sub btnApplicationEmail_Click(sender As Object, e As EventArgs) Handles btnApplicationEmail.Click
    'Checks if the email address in the textbox is in the correct format or not
    Try
        Dim Email As New MailAddress(txtApplicationEmail.Text)
        'Sets the settings variable to the email in the textbox
        My.Settings.applicationEmail = txtApplicationEmail.Text
        'Output to show email was updated
        lblApplicationEmail.ForeColor = Color.Green
        lblApplicationEmail.Text = "Email successfully updated."
    Catch
        'Output to show error in validating email address
        lblApplicationEmail.ForeColor = Color.Red
        lblApplicationEmail.Text = "Email address in incorrect format."
    End Try
End Sub

Private Sub Settings_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Closes the form completely
    Me.Dispose()
End Sub

Private Sub btnVariables_Click(sender As Object, e As EventArgs) Handles btnVariables.Click
    'Opens the Settings Variables form
    SettingsVariables.ShowDialog()
End Sub
End Class

```

Settings Variables (SettingsVariables.vb)

```

Public Class SettingsVariables
    Private Sub SettingsVariables_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Focuses on the enable textbox
        txtEnable.Select()
    End Sub

    Private Sub btnEnable_Click(sender As Object, e As EventArgs) Handles btnEnable.Click
        'Checks the password in the textbox
        If txtEnable.Text = "password" Then
            'Fills in the textboxes with the appropriate variables
            txtDbLocation.Text = My.Settings.dbLocation
        End If
    End Sub

```

```

txtbackupDate.Text = My.Settings.backupDate
txtaccountAdmin.Text = My.Settings.accountAdmin
txtaccountUsername.Text = My.Settings.accountUsername
txtaccountFullName.Text = My.Settings.accountFullName
txtapplicationEmail.Text = My.Settings.applicationEmail
txtTEADate.Text = My.Settings.TEADate
txtTEACount.Text = My.Settings.TEACount
txtBEADate.Text = My.Settings.BEADate
txtBEACount.Text = My.Settings.BEACount
'Feedback for user to show success
lblAdmin.Text = "Correct password."
lblAdmin.ForeColor = Color.Lime
lblAdmin.Visible = True
'Resets enable textbox contents
txtEnable.Text = ""
Else
    'Feedback for incorrect password
    lblAdmin.Text = "Incorrect password."
    lblAdmin.ForeColor = Color.Red
    lblAdmin.Visible = True
    'Resets textbox contents
    txtEnable.Text = ""
End If
End Sub

Private Sub txtEnable_KeyDown(ByVal sender As Object, ByVal e As KeyEventArgs) Handles txtEnable.KeyDown
    'Calls subroutine of the enable button when enter key is pressed in the Enable textbox
    If e.KeyCode = Keys.Enter Then
        Call btnEnable_Click(sender, e)
    End If
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Closes the form completely
    Me.Dispose()
End Sub

Private Sub SettingsVariables_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Closes the form completely
    Me.Dispose()
End Sub
End Class

```

Status Box Form (StatusBox.vb)

```

Public Class StatusBox
    Private Sub StatusBox_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Focuses on the OK button
        btnOK.Select()
    End Sub

    Private Sub btnOK_Click(sender As Object, e As EventArgs) Handles btnOK.Click
        'Closes the form
        Me.Close()
    End Sub
End Class

```

Text Formatting Class (textFormatting.vb)

```

Public Class textFormatting
    Shared Sub numbersOnly(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
        'Checks if keypress is not a backspace
        If e.KeyChar <> Chr(&H8) Then
            'Checks if keypress is not a digit
            If Not Char.IsDigit(e.KeyChar) Then
                e.Handled = True
            End If
        End If
    End Sub

    Shared Sub twoDecimalPoints(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
        'Handles the event if the key pressed is not a digit or a decimal point
        If Not Char.IsDigit(e.KeyChar) And Not e.KeyChar = "." Then
            e.Handled = True
        Else
            'Handles the keypress if it is a decimal and it currently does exist (does not allow two decimal points)
            If e.KeyChar = "." And sender.Text.IndexOf(".") <> -1 Then
                e.Handled = True
            ElseIf e.KeyChar = "." Then 'If the keypress is a decimal point and none exists then allows input
                e.Handled = False
            ElseIf Char.IsDigit(e.KeyChar) Then 'If the keypress is a digit then checks if there is a decimal point
                If sender.Text.IndexOf(".") <> -1 Then
                    'Handles if there are 2 numbers after the decimal point
                    If sender.Text.Length >= sender.Text.IndexOf(".") + 3 Then
                        e.Handled = True
                    End If
                End If
            End If
        End If
    End Sub

```

```

'Allow backspace in textbox
If e.KeyChar = Chr(&H8) Then
    e.Handled = False
End If
End Sub
End Class

```

Travel Application - Overview.vb

```

Imports System.Data.OleDb
Imports Odin.ListViewSort

Public Class Travel_Application__Overview
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Private m_SortingColumn As ColumnHeader

    Private Sub Travel_Application__Overview_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Runs subroutines to populate the listviews
        populateListViewPending()
        populateListViewApproved()
        'Focuses the form
        PictureBox.Select()
    End Sub

    Public Sub populateListViewPending()
        'Clears and enables the listview
        lstViewPending.Items.Clear()
        lstViewPending.Enabled = True
        'Opens the connection and declares a new OleDbCommand and Reader
        connection.Open()
        Dim cmd As New OleDbCommand("SELECT * From TEA WHERE Approved = False ORDER BY TrackingNumber", connection)
        Dim dr As OleDbDataReader = cmd.ExecuteReader
        'Loops the Accounts table of the database and adds each record to a new ListViewItem
        Do While dr.Read()
            Dim newitem As New ListViewItem(dr.Item("TrackingNumber").ToString)
            newitem.SubItems.Add(dr.Item("FullName").ToString)
            newitem.SubItems.Add(dr.Item("Destinations").ToString)
            newitem.SubItems.Add(dr.Item("Duration").ToString)
            newitem.SubItems.Add(dr.Item("TotalCost"))
            lstViewPending.Items.Add(newitem)
        Loop
        'Closes the connection
        connection.Close()
    End Sub

```

```

Public Sub populateListViewApproved()
    'Clears and enables the listviews
    lstViewApproved.Items.Clear()
    lstViewApproved.Enabled = True
    'Opens the connection and declares a new OleDbCommand and Reader
    connection.Open()
    Dim cmd As New OleDbCommand("SELECT * From TEA WHERE Approved = True ORDER BY TrackingNumber", connection)
    Dim dr As OleDbDataReader = cmd.ExecuteReader
    'Loops the Accounts table of the database and adds each record to a new ListViewItem
    Do While dr.Read()
        Dim newitem As New ListViewItem(dr.Item("TrackingNumber").ToString)
        newitem.SubItems.Add(dr.Item("FullName").ToString)
        newitem.SubItems.Add(dr.Item("Destinations").ToString)
        newitem.SubItems.Add(dr.Item("Duration").ToString)
        newitem.SubItems.Add(dr.Item("TotalCost"))
        lstViewApproved.Items.Add(newitem)
    Loop
    'Closes the connection
    connection.Close()
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles btnNew.Click
    'Opens the new travel application form
    Travel_Application__New.ShowDialog()
End Sub

Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles btnSearch.Click
    'Opens SearchBox form and passes the relevant function text
    SearchBox.lblFunction.Text = "TEA"
    SearchBox.ShowDialog()
End Sub

Private Sub Travel_Application__Overview_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Opens the main menu form if form is closed
    MainMenu.Show()
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Checks if either listviews has an item selected
    If lstViewPending.SelectedItems.Count > 0 Or lstViewApproved.SelectedItems.Count > 0 Then
        'Displays a MsgBox asking user to confirm to delete the record
        Dim Response = MsgBox("Are you sure you want to delete this Travel Expense Application?", MsgBoxStyle.YesNo,
            "Delete Travel Expense Application")
        If Response = MsgBoxResult.Yes Then
            'Checks if an item is selected in the pending listview or not

```

```

If lstViewPending.SelectedItems.Count > 0 Then
    'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
    Dim pending As String = lstViewPending.SelectedItems(0).Text
    Dim cmd As New OleDbCommand("DELETE FROM TEA WHERE TrackingNumber = '" & pending & "'", connection)
    'Opens connection, executes the command and closes the connection
    connection.Open()
    cmd.ExecuteNonQuery()
    connection.Close()
    'Repopulates the listview
    populateListViewPending()
    'Updates label to show status
    lblDataStatus.Text = "Record successfully deleted."
    lblDataStatus.Visible = True
    'Checks if item is selected in the approved listview
ElseIf lstViewApproved.SelectedItems.Count > 0 Then
    'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
    Dim approved As String = lstViewApproved.SelectedItems(0).Text
    Dim cmd As New OleDbCommand("DELETE FROM TEA WHERE TrackingNumber = '" & approved & "'", connection)
    'Opens connection, executes the command and closes the connection
    connection.Open()
    cmd.ExecuteNonQuery()
    connection.Close()
    'Repopulates the listview
    populateListViewApproved()
    'Updates label to show status
    lblDataStatus.Text = "Record successfully deleted."
    lblDataStatus.Visible = True
End If
End If
'If neither listviews has an item selected
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnEdit_Click(sender As Object, e As EventArgs) Handles btnEdit.Click
    'Checks if an item is selected in either listview or not
    If lstViewPending.SelectedItems.Count > 0 Then
        'Passes value of ID to Edit form and opens it
        Dim TrackingNumber As String = lstViewPending.SelectedItems(0).Text
        Travel_Application__View_and_Edit.lblTracking.Text = TrackingNumber
        Travel_Application__View_and_Edit.ShowDialog()
    ElseIf lstViewApproved.SelectedItems.Count > 0 Then
        'Passes value of ID to Edit form and opens it
        Dim TrackingNumber As String = lstViewApproved.SelectedItems(0).Text
        Travel_Application__View_and_Edit.lblTracking.Text = TrackingNumber

```

```

    Travel_Application__View_and_Edit.ShowDialog()
    'Output to tell user to select a record
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub lstViewPending_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles
lstViewPending.ColumnClick
    'Get the new sorting column to be sorted
    Dim new_sorting_column As ColumnHeader = lstViewPending.Columns(e.Column)
    'Find out the sorting order
    Dim sort_order As System.Windows.Forms.SortOrder
    If m_SortingColumn Is Nothing Then
        'Sort column by ascending order
        sort_order = SortOrder.Ascending
    Else
        'Checks if sorting column is the same one
        If new_sorting_column.Equals(m_SortingColumn) Then
            'If it is the same column then the sorting order is switched around
            If m_SortingColumn.Text.StartsWith("> ") Then
                sort_order = SortOrder.Descending
            Else
                sort_order = SortOrder.Ascending
            End If
        Else
            'Sort by ascending value
            sort_order = SortOrder.Ascending
        End If
        'Removes the old sort column indicator
        m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
    End If
    'Displays the new sorting column indicator
    m_SortingColumn = new_sorting_column
    If sort_order = SortOrder.Ascending Then
        m_SortingColumn.Text = "> " & m_SortingColumn.Text
    Else
        m_SortingColumn.Text = "< " & m_SortingColumn.Text
    End If
    'Creates the comparer and sorts the listview accordingly
    lstViewPending.ListViewItemSorter = New clsListViewSorter(e.Column, sort_order)
    lstViewPending.Sort()
End Sub

Private Sub lstViewApproved_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles
lstViewApproved.ColumnClick

```

```

'Get the new sorting column to be sorted
Dim new_sorting_column As ColumnHeader = lstViewApproved.Columns(e.Column)
'Find out the sorting order
Dim sort_order As System.Windows.Forms.SortOrder
If m_SortingColumn Is Nothing Then
    'Sort column by ascending order
    sort_order = SortOrder.Ascending
Else
    'Checks if sorting column is the same one
    If new_sorting_column.Equals(m_SortingColumn) Then
        'If it is the same column then the sorting order is switched around
        If m_SortingColumn.Text.StartsWith("> ") Then
            sort_order = SortOrder.Descending
        Else
            sort_order = SortOrder.Ascending
        End If
    Else
        'Sort by ascending value
        sort_order = SortOrder.Ascending
    End If
    'Removes the old sort column indicator
    m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
End If
'Displays the new sorting column indicator
m_SortingColumn = new_sorting_column
If sort_order = SortOrder.Ascending Then
    m_SortingColumn.Text = "> " & m_SortingColumn.Text
Else
    m_SortingColumn.Text = "< " & m_SortingColumn.Text
End If
'Creates the comparer and sorts the listview accordingly
lstViewApproved.ListViewItemSorter = New clsListViewSorter(e.Column, sort_order)
lstViewApproved.Sort()
End Sub
End Class

```

Travel Application - New.vb

```

Imports System.Data.OleDb
Imports System.Net.Mail

Public Class Travel_Application__New

```

```

Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)

Private Sub Travel_Application__New_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    'Runs subroutine to populate the comboboxes
    populateCBO()
    'Selects the users username in the combination box and sets email textbox
    cboName.Text = My.Settings.accountFullName
    txtEmail.Text = My.Settings.accountUsername & "@odinhealth.co.nz"
    'Sets the Tracking Number
    If My.Settings.TEADate = DateTime.Today Then
        'Sets tracking number if the settings variable date is the same as todays date
        lblTracking.Text = "TV" & DateTime.Now.ToString("ddMMyy") & My.Settings.TEACount
    Else
        'Sets the settings variable to todays date if it does not match todays date
        My.Settings.TEADate = DateTime.Today
        'Sets the count for the number of applications to 0
        My.Settings.TEACount = 0
        lblTracking.Text = "TV" & DateTime.Now.ToString("ddMMyy") & My.Settings.TEACount
    End If
    'Focuses on the form
    PictureBox.Select()
End Sub

Private Sub btnSubmit_Click(sender As Object, e As EventArgs) Handles btnSubmit.Click
    'Checks if the function returns a true
    If Validation() Then
        Try
            'Declares the new OleDbCommand
            Dim cmd As New OleDbCommand("INSERT INTO TEA (TrackingNumber, FullName, Email, Destinations, Duration, Reason, " & _
                "AccommodationCosts, DomesticTravelCosts, InternationalTravelCosts, TotalCost, Approved)" & _
                "VALUES (@TrackingNumber, @FullName, @Email, @Destinations, @Duration, @Reason, " & _
                "@AccommodationCosts, @DomesticTravelCosts, @InternationalTravelCosts, @TotalCost, " & _
                "@Approved)", connection)
            'Gives the parameters its values
            cmd.Parameters.AddWithValue("TrackingNumber", lblTracking.Text)
            cmd.Parameters.AddWithValue("FullName", cboName.Text)
            cmd.Parameters.AddWithValue("Email", txtEmail.Text)
            cmd.Parameters.AddWithValue("Destinations", txtDestinations.Text)
            cmd.Parameters.AddWithValue("Duration", txtDuration.Text)
            cmd.Parameters.AddWithValue("Reason", rtbReason.Text)
            cmd.Parameters.AddWithValue("AccommodationCosts", txtAccommodation.Text)
            cmd.Parameters.AddWithValue("DomesticTravelCosts", txtDomestic.Text)
            cmd.Parameters.AddWithValue("InternationalTravelCosts", txtInternational.Text)
            cmd.Parameters.AddWithValue("TotalCost", txtTotal.Text)
            cmd.Parameters.AddWithValue("Approved", False)
        End Try
    End If
End Sub

```

```

'Opens database connection, executes query and closes the database
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Adds 1 onto the count for making the application number
My.Settings.TEACount = My.Settings.TEACount + 1
'Try running the email sending subroutine
Try
    SendEmail()
    'Output to show successful updation on Overview form
    Travel_Application__Overview.lblDataStatus.Text = "Record added and email sent."
    Travel_Application__Overview.lblDataStatus.ForeColor = Color.Green
    Travel_Application__Overview.lblDataStatus.Visible = True
    'Closes the form completely
    Me.Dispose()
Catch
    'Output to show record added but email not sent
    MsgBox("Application successfully added into database but email did not successfully send.")
    'Output to show successful updation on Overview form
    Travel_Application__Overview.lblDataStatus.Text = "Record successfully added."
    Travel_Application__Overview.lblDataStatus.ForeColor = Color.Green
    Travel_Application__Overview.lblDataStatus.Visible = True
    'Closes the form completely
    Me.Dispose()
End Try
Catch
    'Shows error if record cannot be inserted
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "A record with the tracking number already exists."
End Try
End If
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Displays a MsgBox asking user to confirm to close the form
    Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
    'Closes form completely if user selects 'Yes'
    If Response = MsgBoxResult.Yes Then
        Me.Dispose()
    End If
End Sub

Private Sub cboName_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cboName.SelectedIndexChanged
    'Update the textbox containing the email when combobox selection is altered
    Dim Name As String = cboName.Text

```

```

Name = Name.ToLower
Name = Name.Replace(" ", ".")
txtEmail.Text = Name & "@odinhealth.co.nz"
End Sub

Private Function Validation()
    'First checks if all the textboxes contain text
    If txtEmail.Text <> "" And txtDestinations.Text <> "" And txtDuration.Text <> "" And rtbReason.Text <> "" And txtDomestic.Text <> "" _ 
    And txtInternational.Text <> "" And txtAccommodation.Text <> "" And txtTotal.Text <> "" Then
        'Validates the email address. Throws error if the address is in the wrong format
        Try
            'The function returns true
            Dim Email As New MailAddress(txtEmail.Text)
            Return True
        Catch
            'Output to show that email address format was incorrect.
            lblStatus.ForeColor = Color.Red
            lblStatus.Text = "Please check that the email address you entered was in a correct format."
            Return False
        End Try
    Else
        'Output to show that one or more of the textboxes have not been filled.
        lblStatus.ForeColor = Color.Red
        lblStatus.Text = "Please ensure all fields are filled out correctly."
        Return False
    End If
End Function

Private Sub Travel_Application__New_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Closes form completely and runs subroutines to repopulate the listviews
    Me.Dispose()
    Travel_Application__Overview.populateListViewPending()
    Travel_Application__Overview.populateListViewApproved()
End Sub

Private Sub populateCBO()
    'Declares DataSet and fills it up with data from Username Column in database
    Dim ds As New DataSet()
    Using da As New OleDbDataAdapter("SELECT FullName FROM Accounts", connection)
        da.Fill(ds, "FullName")
    End Using
    'Loads DataSet data into a DataView, sorts it then writes it to a DataTable
    Dim view As New DataView(ds.Tables("FullName"))
    view.Sort = "FullName"
    Dim sorted As DataTable = view.ToTable()

```

```

'Populates comboboxes with contents of the new Datatable
With cboName
    .DisplayMember = "FullName"
    .DataSource = view
    .SelectedIndex = 0
End With
End Sub

Private Sub SendEmail()
    'Disables the form
    Me.Enabled = False
    'Output to show that validation was successful.
    lblStatus.ForeColor = Color.ForestGreen
    lblStatus.Text = "Validation successful - Adding application to database and sending email for approval. Please wait a moment..."
    'Creates new message and declares Name variable
    Dim Email As New MailMessage()
    Dim Name As String = cboName.Text
    'Sets the email preferences
    Email.From = New MailAddress(txtEmail.Text)
    Email.[To].Add(My.Settings.applicationEmail)
    'Sets the content of the email
    Email.Subject = "Travel Application #" & lblTracking.Text
    Email.Body = "<H2>Travel Application for " & Name & "</H2>" & "Date of Submission: " & DateTime.Today & "<BR>Return Email: " & _
        txtEmail.Text & "<BR><BR>" & "Destinations: " & txtDestinations.Text & " (Duration: " & txtDuration.Text & " Days)<BR><BR>" & _
        "Reason for Trip: " & rtbReason.Text & "<BR><BR>" & "<u>Estimated Costs:</u>" & "<BR>" & _
        "Domestic Travel Costs: $" & txtDomestic.Text & " NZD<BR>" & "International Travel Costs: $" & _
        txtInternational.Text & " NZD<BR>" & "Accomodation Costs: $" & txtAccomodation.Text & " NZD<BR>" & _
        "Total Costs: $" & txtTotal.Text & " NZD<BR><BR>To approve this application, open the Business " & _
        "Manager program and navigate to the Travel Applications, select the application click the 'View and Edit' button and " & _
        "then tick the approved box and 'Update'"
    Email.IsBodyHtml = True
    'Sets the server to send the email
    Dim smtp As New SmtpClient("smtp.gmail.com")
    smtp.UseDefaultCredentials = False
    smtp.Credentials = New Net.NetworkCredential("shen.odinhealth@gmail.com", "c9dMy6RKPr")
    smtp.Port = 587
    smtp.EnableSsl = True
    smtp.Host = "smtp.gmail.com"
    'Sends the email and shows feedback
    smtp.Send(Email)
    lblStatus.ForeColor = Color.Green
    lblStatus.Text = "Application added into database and email has been successfully sent for approval. Thank You!"
    'Enables the form
    Me.Enabled = True
End Sub

```

```

    Private Sub txtDuration_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtDuration.KeyPress
        'Runs subroutine on textFormatting class to allow numbers only into textbox
        textFormatting.numbersOnly(sender, e)
    End Sub

    Private Sub CostAddition_TextChanged(sender As Object, e As EventArgs) Handles txtDomestic.TextChanged, txtInternational.TextChanged,
txtAccomodation.TextChanged
        'Declares variables and gives them values according to their relevant textboxes
        Dim d, i, a, t As Double
        d = Val(txtDomestic.Text)
        i = Val(txtInternational.Text)
        a = Val(txtAccomodation.Text)
        'Adds the values in the textboxes for estimated costs together to give total
        t = d + i + a
        txtTotal.Text = t
    End Sub

    Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtDomestic.KeyPress,
txtInternational.KeyPress,
        txtAccomodation.KeyPress, txtTotal.KeyPress
        'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
        textFormatting.twoDecimalPoints(sender, e)
    End Sub
End Class

```

Travel Application - View and Edit.vb

```

Imports System.Data.OleDb
Imports System.Net.Mail

Public Class Travel_Application__View_and_Edit
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Dim Approved As Boolean

    Private Sub Travel_Application__View_and_Edit_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Opens the connection and declares a new OleDbCommand and Reader
        Dim TrackingNumber As String = lblTracking.Text
        connection.Open()
        Dim dt As New DataTable()
        'Fills datatable with data from the TEA table
        Using da As New OleDbDataAdapter("SELECT * FROM TEA WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
            da.Fill(dt)
        End Using
    End Sub

```

```

End Using
'Closes the connection
connection.Close()
'Sets values in textboxes to data in datatable
txtName.Text = dt.Rows(0)("FullName").ToString
txtEmail.Text = dt.Rows(0)("Email").ToString
txtDestinations.Text = dt.Rows(0)("Destinations").ToString
txtDuration.Text = dt.Rows(0)("Duration").ToString
rtbReason.Text = dt.Rows(0)("Reason").ToString
txtAccomodation.Text = dt.Rows(0)("AccomodationCosts").ToString
txtDomestic.Text = dt.Rows(0)("DomesticTravelCosts").ToString
txtInternational.Text = dt.Rows(0)("InternationalTravelCosts").ToString
txtTotal.Text = dt.Rows(0)("TotalCost").ToString
chkApproved.Checked = dt.Rows(0)("Approved")
If My.Settings.accountAdmin = True Then
    txtName.ReadOnly = False
    chkApproved.Enabled = True
End If
'Sets "Approved" variable to true or false to be used in sending email to user.
If chkApproved.Checked = False Then
    Approved = False
Else
    Approved = True
End If
'Focuses form
PictureBox.Select()
End Sub

Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
    'Runs Validation function and checks if it is true
    If Validation() Then
        'Declares the new OleDbCommand and gives it parameters
        Dim cmd As New OleDbCommand("UPDATE TEA SET FullName = ?, Email = ?, Destinations = ?, Duration = ?, Reason = ?, " & _
                                    "AccomodationCosts = ?, DomesticTravelCosts = ?, InternationalTravelCosts = ?, " & _
                                    "TotalCost = ?, Approved = ? WHERE TrackingNumber = '" & lblTracking.Text & "'", connection)
        cmd.Parameters.AddWithValue("@p1", txtName.Text)
        cmd.Parameters.AddWithValue("@p2", txtEmail.Text)
        cmd.Parameters.AddWithValue("@p3", txtDestinations.Text)
        cmd.Parameters.AddWithValue("@p4", txtDuration.Text)
        cmd.Parameters.AddWithValue("@p5", rtbReason.Text)
        cmd.Parameters.AddWithValue("@p6", txtAccomodation.Text)
        cmd.Parameters.AddWithValue("@p7", txtDomestic.Text)
        cmd.Parameters.AddWithValue("@p8", txtInternational.Text)
        cmd.Parameters.AddWithValue("@p9", txtTotal.Text)
        cmd.Parameters.AddWithValue("@p10", chkApproved.Checked)
    End If
End Sub

```

```

'Executes the command and closes form
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Output to show successful update on Overview form
Travel_Application__Overview.lblDataStatus.Text = "Record successfully updated."
Travel_Application__Overview.lblDataStatus.ForeColor = Color.Green
Travel_Application__Overview.lblDataStatus.Visible = True
'If the checkbox was initially unchecked and is now checked then send email
If Approved = False And chkApproved.Checked = True Then
    Try
        'Runs subroutine to send email
        SendEmail()
        'Output to show successful update and email sent on Overview form and closes form
        Travel_Application__Overview.lblDataStatus.Text = "Record updated and email sent."
        Me.Close()
    Catch
        'Output in case email doesn't send and closes form
        MsgBox("Email could not be successfully sent but record was updated.")
        Me.Close()
    End Try
End If
'Closes form
Me.Close()
End If
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Displays a MsgBox asking user to confirm to delete the application
    Dim Response = MsgBox("Are you sure you want to delete this Travel Expense Application?", MsgBoxStyle.YesNo, "Delete Application")
    If Response = MsgBoxResult.Yes Then
        'Declares a new OleDbCommand which deletes the record if user selects 'Yes'
        Dim TrackingNumber As String = lblTracking.Text
        Dim cmd As New OleDbCommand("DELETE FROM TEA WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
        'Opens connection, executes the command and closes the connection
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Output to show successful updation on Overview form
        Travel_Application__Overview.lblDataStatus.Text = "Record successfully deleted."
        Travel_Application__Overview.lblDataStatus.ForeColor = Color.Green
        Travel_Application__Overview.lblDataStatus.Visible = True
        'Closes form
        Me.Close()
    End If

```

```

End Sub

Private Function Validation()
    'First checks if all the textboxes and richtextbox contain text
    If txtName.Text <> "" And txtEmail.Text <> "" And txtDestinations.Text <> "" And txtDuration.Text <> "" And rtbReason.Text <> "" _ 
        And txtDomestic.Text <> "" And txtInternational.Text <> "" And txtAccomodation.Text <> "" And txtTotal.Text <> "" Then
            'Validates the email address. Throws error if the address is in the wrong format
            Try
                'The function returns true
                Dim Email As New MailAddress(txtEmail.Text)
                Return True
            Catch
                'Output to show that email address format was incorrect.
                lblStatus.ForeColor = Color.Red
                lblStatus.Text = "Please check that the email address you entered was in a correct format."
                Return False
            End Try
        Else
            'Output to show that one or more of the textboxes have not been filled.
            lblStatus.ForeColor = Color.Red
            lblStatus.Text = "Please ensure all fields are filled out correctly."
            Return False
        End If
    End Function

Private Sub Travel_Application__View_and_Edit_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Run subroutines to repopulate the listviews on the overview form
    Travel_Application__Overview.populateListViewPending()
    Travel_Application__Overview.populateListViewApproved()
    'Closes form completely
    Me.Dispose()
End Sub

Private Sub SendEmail()
    'Disables form controls
    Me.Enabled = False
    'Output to show that validation was successful.
    lblStatus.ForeColor = Color.ForestGreen
    lblStatus.Text = "Updating application and sending email to user. Please wait a moment..."
    'Creates new message and declares Name variable
    Dim Email As New MailMessage()
    Dim Name As String = txtName.Text
    'Sets the email preferences
    Email.From = New MailAddress("shen.odinhealth@gmail.com")
    Email.[To].Add(txtEmail.Text)

```

```

'Sets the content of the email and formats it
Email.Subject = "Approval - Travel Application #" & lblTracking.Text
Email.Body = "<H2>Travel Application for " & Name & " has been approved.</H2>" & "Date of Approval: " & DateTime.Today & "<BR>Details
for #" & _
    lblTracking.Text & "<BR><BR>" & "Destinations: " & txtDestinations.Text & " (Duration: " & txtDuration.Text & " Days)<BR><BR>" & _
    "Reason for Trip: " & rtbReason.Text & "<BR><BR>" & "<u>Estimated Costs:</u>" & "<BR>" & _
    "Domestic Travel Costs: $" & txtDomestic.Text & " NZD<BR>" & "International Travel Costs: $" & _
    txtInternational.Text & " NZD<BR>" & "Accommodation Costs: $" & txtAccommodation.Text & " NZD<BR>" & _
    "Total Costs: $" & txtTotal.Text & " NZD<BR><BR>You have been approved for this travel application. You may now " & _
    "go ahead and book or purchase the relevant expenses. Have a safe journey!"
Email.IsBodyHtml = True
'Sets the server to send the email
Dim smtp As New SmtpClient("smtp.gmail.com")
smtp.UseDefaultCredentials = False
smtp.Credentials = New Net.NetworkCredential("shen.odinhealth@gmail.com", "c9dMy6RKPr")
smtp.Port = 587
smtp.EnableSsl = True
smtp.Host = "smtp.gmail.com"
'Sends the email and shows feedback
smtp.Send(Email)
lblStatus.ForeColor = Color.Green
lblStatus.Text = "Application updated and email has been successfully sent."
'Enables form controls
Me.Enabled = True
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
'Displays a MsgBox asking user to confirm to close the form
Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
If Response = MsgBoxResult.Yes Then
    Me.Close()
    MainMenu.Show()
End If
End Sub

Private Sub CostAddition_TextChanged(sender As Object, e As EventArgs) Handles txtDomestic.TextChanged, txtInternational.TextChanged,
txtAccommodation.TextChanged
'Declares variables and assigns them values based on their relevant textboxes
Dim d, i, a, t As Double
d = Val(txtDomestic.Text)
i = Val(txtInternational.Text)
a = Val(txtAccommodation.Text)
'Adds the values in the textboxes for estimated costs and adds them together
t = d + i + a
txtTotal.Text = t

```

```

End Sub

Private Sub txtDuration_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtDuration.KeyPress
    'Runs subroutine on textFormatting class to allow numbers only into textbox
    textFormatting.numbersOnly(sender, e)
End Sub

Private Sub txtDomestic_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtDomestic.KeyPress,
txtInternational.KeyPress,
    txtAccomodation.KeyPress, txtTotal.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

Travel Expense - Overview.vb

```

Imports System.Data.OleDb
Imports Odin.ListViewSort
Imports System.Drawing.Printing

Public Class Travel_Expense__Overview
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Dim dgvPrinter As DataGridViewPrinter
    Private m_SortingColumn As ColumnHeader

    Private Sub Travel_Expense__Overview_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Runs subroutine to populate the listview
        populateListView()
        'Sets font on the datagridview for printing
        DataGridView.Font = New Font("Segoe UI", 10.5)
        'Focuses on form
        PictureBox.Select()
    End Sub

    Private Sub btnView_Click(sender As Object, e As EventArgs) Handles btnView.Click
        'Checks if an item is selected in the listview or not
        If lstView.SelectedItems.Count > 0 Then
            'Passes value of TrackingNumber to expense form and opens it
            Dim TrackingNumber As String = lstView.SelectedItems(0).Text
            Travel_Expense.txtTrackingNumber.Text = TrackingNumber
            Travel_Expense.txtTrackingNumber.ReadOnly = True
            Travel_Expense.cboName.Enabled = False
            Travel_Expense.cboName.Text = lstView.SelectedItems(0).SubItems(1).Text
        End If
    End Sub

```

```

    Travel_Expense.ShowDialog()
    'Output error if no item is selected
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Checks if an item has been selected or not
    If lstView.SelectedItems.Count > 0 Then
        If My.Settings.accountAdmin = True Then
            'Displays a MsgBox asking user to confirm to delete the record
            Dim Response = MsgBox("Are you sure you want to delete the records for this Travel Expense?" & vbCrLf & "All data " & _
                "on the given tracking number WILL be deleted in the Travel Expense table.", MsgBoxStyle.YesNo, "Delete
Record")
        If Response = MsgBoxResult.Yes Then
            'Checks if an item is selected in the listview or not if user response is 'Yes'
            If lstView.SelectedItems.Count > 0 Then
                'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
                Dim TrackingNumber As String = lstView.SelectedItems(0).Text
                Dim cmd As New OleDbCommand("DELETE FROM [Travel Expenses] WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
                'Opens connection, executes the command and closes the connection
                connection.Open()
                cmd.ExecuteNonQuery()
                connection.Close()
                'Repopulates the listview
                populateListView()
                'Updates label to show status
                lblDataStatus.Text = "Records deleted."
                lblDataStatus.ForeColor = Color.Green
                lblDataStatus.Visible = True
            End If
        End If
    Else : MsgBox("You are not authorised to carry out this action. You must be an admin.")
    End If
    'Output error if no record has been selected
Else : MsgBox("Please select a record from the table.")
End If
End Sub

//PRINTING
Private Sub btnPrint_Click(sender As Object, e As EventArgs) Handles btnPrint.Click
    'Checks if an item has been selected or not
    If lstView.SelectedItems.Count > 0 Then
        'Declares variables to be used in query
        Dim TrackingNumber As String = lstView.SelectedItems(0).Text

```

```

Dim Name As String = lstView.SelectedItems(0).SubItems(1).Text
'Declares OleDbCommand, Data Adapter and Dataset
Dim cmd As New OleDbCommand("SELECT Format(DateIncurred, 'dd/mm/yy'), Description, BillClient, ExpenseType, ForeignCurrency " & _
    "& ' $" & ForeignAmount, ExchangeRate, '$' & NZDAmount, Reimbursed From [Travel Expenses] WHERE " & _
    "TrackingNumber = '" & TrackingNumber & "' ORDER BY DateIncurred", connection)
Dim da As New OleDbDataAdapter(cmd)
Dim dt As New DataTable
'Fills dataset with data from table and outputs it to datagridview
da.FillSchema(dt, SchemaType.Source)
dt.Columns(2).DataType = GetType(String)
dt.Columns(5).DataType = GetType(String)
dt.Columns(7).DataType = GetType(String)
da.Fill(dt)
'Displays data on datagridview and sets the column names
DataGridView.DataSource = dt
With DataGridView
    .Columns(0).HeaderCell.Value = "Date"
    .Columns(2).HeaderCell.Value = "BC"
    .Columns(3).HeaderCell.Value = "Type"
    .Columns(4).HeaderCell.Value = "Amount"
    .Columns(5).HeaderCell.Value = "XRT"
    .Columns(6).HeaderCell.Value = "NZD"
    .Columns(7).HeaderCell.Value = "Repaid"
End With
'Loops through datagridview and replaces True with Yes and False with Nothing for BillClient and
'False with No for Repaid. Calculates total for the NZD and Reimbursed column (if it is false) of the datagridview
Dim totalDec As Decimal = 0
Dim repaidDec As Decimal = 0
For i = 0 To DataGridView.Rows.Count - 1
    'Checks if cell for Bill Client contains True
    If DataGridView.Rows(i).Cells(2).Value = "True" Then
        DataGridView.Rows(i).Cells(2).Value = "Yes"
    Else : DataGridView.Rows(i).Cells(2).Value = ""
    End If
    'Checks if cell for Repaid contains True
    If DataGridView.Rows(i).Cells(7).Value = "True" Then
        DataGridView.Rows(i).Cells(7).Value = "Yes"
    Else
        DataGridView.Rows(i).Cells(7).Value = "No"
        repaidDec += DataGridView.Rows(i).Cells(6).Value
    End If
    'Sums the values in the NZD column
    totalDec = totalDec + DataGridView.Rows(i).Cells(6).Value
Next i
'Rounds the total variables to 2 decimal points and converts them to string with $ symbol

```

```

totalDec = Decimal.Round(totalDec, 2, MidpointRounding.AwayFromZero)
repaidDec = Decimal.Round(repaidDec, 2, MidpointRounding.AwayFromZero)
Dim totalStr As String = "$" & CStr(totalDec)
Dim repaidStr As String = "$" & CStr(repaidDec)
'Adds new row to DataTable to update the datagridview. Sets values in rows then adds
Dim newRow As DataRow = dt.NewRow
For i = 0 To 5
    newRow.Item(i) = ""
Next
newRow.Item(6) = totalStr
newRow.Item(7) = repaidStr
dt.Rows.Add(newRow)
'Runs function to set up printing
If SetupThePrinting(TrackingNumber & " - " & Name) Then
    'Declares and opens print preview dialog
    Dim ppd As PrintPreviewDialog = New PrintPreviewDialog()
    ppd.Document = pntDocument
    ppd.Height = Screen.PrimaryScreen.Bounds.Height / 1.5
    ppd.Width = Screen.PrimaryScreen.Bounds.Width / 2.5
    ppd.FindForm().StartPosition = FormStartPosition.CenterScreen
    ppd.Text = "Travel Expenses"
    ppd.ShowDialog()
End If
Else : MsgBox("Please select a record from the table.") 'If user did not select a listview item
End If

End Sub

Private Sub pntDocument_PrintPage(ByVal sender As System.Object, ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles pntDocument.PrintPage
    'Checks if there are any more pages
    Dim more As Boolean = dgvPrinter.DrawDataGridView(e.Graphics)
    If more = True Then
        e.HasMorePages = True
    End If
End Sub

Private Function SetupThePrinting(ByVal title As String) As Boolean
    'Declares print dialog and sets properties
    Dim pd As PrintDialog = New PrintDialog()
    pd.AllowCurrentPage = False
    pd.AllowPrintToFile = False
    pd.AllowSelection = False
    pd.AllowSomePages = False
    pd.PrintToFile = False

```

```

pd.ShowHelp = False
pd.ShowNetwork = False
'Function returns false if user cancels
If pd.ShowDialog() <> DialogResult.OK Then
    Return False
End If
'Sets properties for the print document and sets the data
pntDocument.DocumentName = "Travel Expense"
pntDocument.PrinterSettings = pd.PrinterSettings
pntDocument.DefaultPageSettings = pd.PrinterSettings.DefaultPageSettings
pntDocument.DefaultPageSettings.Margins = New Margins(20, 20, 20, 20)
dgvPrinter = New DataGridViewPrinter(DataGridView, pntDocument, True, True, title, New Font("Segoe UI", 18, FontStyle.Bold,
GraphicsUnit.Point), Color.Black, True)
Return True
End Function

Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles btnSearch.Click
    'Opens search box form and passes relevant function name
    SearchBox.lblFunction.Text = "TravelExpense"
    SearchBox.ShowDialog()
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles btnNew.Click
    'Opens the form to input or view travel expenses
    Travel_Expense.ShowDialog()
End Sub

Private Sub Travel_Expense_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Shows the main menu form upon closing of this form
    MainMenu.Show()
End Sub

Public Sub populateListView()
    'Clears and enables the listview
    lstView.Items.Clear()
    lstView.Enabled = True
    'Opens the connection and declares a new OleDbCommand and Reader
    connection.Open()
    Dim cmd As New OleDbCommand("SELECT * From [Travel Expenses] ORDER BY TrackingNumber", connection)
    Dim dr As OleDbDataReader = cmd.ExecuteReader
    'Loops the Accounts table of the database and adds each record to a new ListViewItem
    Do While dr.Read()
        Dim newitem As New ListViewItem(dr.Item("TrackingNumber").ToString)
        newitem.SubItems.Add(dr.Item("FullName").ToString)
        newitem.SubItems.Add(dr.Item("DateIncurred"))
    End While
End Sub

```

```

newitem.SubItems.Add(dr.Item("NZDAmount"))
newitem.SubItems.Add(dr.Item("Reimbursed"))
lstView.Items.Add(newitem)
Loop
'Closes the connection
connection.Close()
'Checks that the listview contains at least 2 items
If Not lstView.Items.Count <= 1 Then
    'Declares variables used for new listviewitem
    Dim TrackingNumber As String
    Dim Name As String
    Dim dateFrom As Date
    Dim dateTo As Date
    Dim NZDAmount As Decimal
    Dim Reimbursed As Boolean
    'Sets variables according to their items and subitems in the first item in the listview
    TrackingNumber = lstView.Items(0).Text
    Name = lstView.Items(0).SubItems(1).Text
    dateFrom = lstView.Items(0).SubItems(2).Text
    dateTo = lstView.Items(0).SubItems(2).Text
    NZDAmount = lstView.Items(0).SubItems(3).Text
    Reimbursed = lstView.Items(0).SubItems(4).Text
    'Declares variable and sets it to length of listview minus one
    Dim lstViewLength As Integer = lstView.Items.Count - 1
    'Loops from the second item in listview to the last currently present
    For x = 1 To lstViewLength
        'Checks if the TrackingNumber in this item is equal to the previous one
        If lstView.Items(x).Text = TrackingNumber Then
            'Checks if the date in this item is smaller than the dateFrom (minimum date)
            If dateFrom > CDate(lstView.Items(x).SubItems(2).Text) Then
                dateFrom = CDate(lstView.Items(x).SubItems(2).Text)
            End If
            'Checks if the date in this item is smaller than the dateTo (maximum date)
            If dateTo < CDate(lstView.Items(x).SubItems(2).Text) Then
                dateTo = CDate(lstView.Items(x).SubItems(2).Text)
            End If
            'Sets the NZDAmount as the current amount plus the value in the current listview item
            NZDAmount = NZDAmount + CDec(lstView.Items(x).SubItems(3).Text)
            'Checks if the expense has been reimbursed or not
            If lstView.Items(x).SubItems(4).Text = False Then
                'Sets the boolean for Reimbursed to false
                Reimbursed = False
            End If
            'Checks if the end of the initial listview has been reached or not
            If x = lstViewLength Then

```

```

' Rounds the NZDAmount accordingly (2 decimal places)
NZDAmount = Decimal.Round(NZDAmount, 2, MidpointRounding.AwayFromZero)
' Adds a new listview item and gives the subitems their values
Dim i As ListViewItem = lstView.Items.Add(TrackingNumber)
i.SubItems.Add(Name)
i.SubItems.Add(dateFrom & " - " & dateTo)
i.SubItems.Add(NZDAmount)
i.SubItems.Add(Reimbursed)
End If
' In the case that the current listview item's tracking number is not equal to the previous one
ElseIf lstView.Items(x).Text <> TrackingNumber Then
    ' Rounds the NZDAmount accordingly to 2 decimal places
    NZDAmount = Decimal.Round(NZDAmount, 2, MidpointRounding.AwayFromZero)
    ' Adds a new listview item and gives the subitems their values
    Dim i As ListViewItem = lstView.Items.Add(TrackingNumber)
    i.SubItems.Add(Name)
    i.SubItems.Add(dateFrom & " - " & dateTo)
    i.SubItems.Add(NZDAmount)
    i.SubItems.Add(Reimbursed)
    ' Sets the variables to the current listview items contents
    TrackingNumber = lstView.Items(x).Text
    Name = lstView.Items(x).SubItems(1).Text
    dateFrom = lstView.Items(x).SubItems(2).Text
    dateTo = lstView.Items(x).SubItems(3).Text
    NZDAmount = lstView.Items(x).SubItems(4).Text
    Reimbursed = lstView.Items(x).SubItems(5).Text
End If
Next x
'Reverse loop from the initial listview length to the first record to delete the items
For x = lstViewLength To 0 Step -1
    ' Deletes the initial listview items
    lstView.Items(x).Remove()
Next
End If
End Sub

Private Sub lstView_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles lstView.ColumnClick
    ' Get the new sorting column to be sorted
    Dim new_sorting_column As ColumnHeader = lstView.Columns(e.Column)
    ' Find out the sorting order
    Dim sort_order As System.Windows.Forms.SortOrder
    If m_SortingColumn Is Nothing Then
        ' Sort column by ascending order
        sort_order = SortOrder.Ascending
    Else

```

```

'Checks if sorting column is the same one
If new_sorting_column.Equals(m_SortingColumn) Then
    'If it is the same column then the sorting order is switched around
    If m_SortingColumn.Text.StartsWith("> ") Then
        sort_order = SortOrder.Descending
    Else
        sort_order = SortOrder.Ascending
    End If
Else
    'Sort by ascending value
    sort_order = SortOrder.Ascending
End If
'Removes the old sort column indicator
m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
End If
'Displays the new sorting column indicator
m_SortingColumn = new_sorting_column
If sort_order = SortOrder.Ascending Then
    m_SortingColumn.Text = "> " & m_SortingColumn.Text
Else
    m_SortingColumn.Text = "< " & m_SortingColumn.Text
End If
'Creates the comparer and sorts the listview accordingly
lstView.ListViewItemSorter = New clsListviewSorter(e.Column, sort_order)
lstView.Sort()
End Sub
End Class

```

Travel Expense – New, View and Edit (Travel Expense.vb)

```

Imports System.Data.OleDb
Imports Odin.ListviewSort
Imports System.Drawing.Printing

Public Class Travel_Expense
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Dim dgvPrinter As DataGridViewPrinter
    Private m_SortingColumn As ColumnHeader

    Private Sub Travel_Expense_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Runs subroutine to populate the combobox with names
        populateCBO()
        'Sets font on the datagridview for printing
        DataGridView.Font = New Font("Segoe UI", 11.5)
        'Calls relevant subroutine to accomodate with View and Edit form open to fill in name
    End Sub

```

```

Call txtTrackingNumber_TextChanged(sender, e)
End Sub

'//CODE FOR BUTTONS/
Private Sub btnSave_Click(sender As Object, e As EventArgs) Handles btnSave.Click
    'Runs function validation and continues if it returns true
    If Validation() Then
        'Sets Reimbursed as True if user doesn't require reimbursement
        Dim Reimbursed As Boolean
        If chkReimburse.Checked = False Then
            Reimbursed = True
        Else : Reimbursed = False
        End If
        'Declares the new OleDbCommand
        Dim cmd As New OleDbCommand("INSERT INTO [Travel Expenses] (TrackingNumber, FullName, DateIncurred, ExpenseType, Description, " & _
            "BillClient, Reimburse, PaymentMethod, ForeignAmount, ForeignCurrency, ExchangeRate, NZDAmount, " & _
            "Reimbursed) VALUES (@TrackingNumber, @FullName, @DateIncurred, @ExpenseType, @Description, " & _
            "@BillClient, @Reimburse, @PaymentMethod, @ForeignAmount, @ForeignCurrency, @ExchangeRate, " & _
            "@NZDAmount, @Reimbursed)", connection)
        'Gives the parameters its values
        cmd.Parameters.AddWithValue("TrackingNumber", txtTrackingNumber.Text)
        cmd.Parameters.AddWithValue("FullName", cboName.Text)
        cmd.Parameters.AddWithValue("DateIncurred", dtpDate.Value.Date)
        cmd.Parameters.AddWithValue("ExpenseType", cboExpenseType.Text)
        cmd.Parameters.AddWithValue("Description", txtDescription.Text)
        cmd.Parameters.AddWithValue("BillClient", chkBillClient.Checked)
        cmd.Parameters.AddWithValue("Reimburse", chkReimburse.Checked)
        cmd.Parameters.AddWithValue("PaymentMethod", cboPaymentMethod.Text)
        cmd.Parameters.AddWithValue("ForeignAmount", txtForeign.Text)
        cmd.Parameters.AddWithValue("ForeignCurrency", cboCurrency.Text)
        cmd.Parameters.AddWithValue("ExchangeRate", txtExchangeRate.Text)
        cmd.Parameters.AddWithValue("NZDAmount", lblNZDAmount.Text)
        cmd.Parameters.AddWithValue("Reimbursed", Reimbursed)
        'Opens database connection, executes query and closes the database
        Try
            connection.Open()
            cmd.ExecuteNonQuery()
            connection.Close()
            'Output to show success
            lblStatus.ForeColor = Color.Green
            lblStatus.Text = "Travel Expense successfully saved."
            'Locks the tracking number and name controls
            txtTrackingNumber.ReadOnly = True
            cboName.Enabled = False
            'Runs subroutine to repopulate the listview and clears existing data on form

```

```

    populateListView()
    clearForm()
  Catch
    'Output in case data could not be saved for any reason
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "An error occurred. The expense was not successfully saved."
  End Try
End If
End Sub

Private Sub btnView_Click(sender As Object, e As EventArgs) Handles btnView.Click
  'Checks if an item is selected in the listview or not
  If lstView.SelectedItems.Count > 0 Then
    'Brings the relevant buttons to the front, enables them and shows checkbox
    btnReimburseAll.Visible = True
    btnUpdate.BringToFront()
    btnUpdate.Enabled = True
    btnDelete1.BringToFront()
    btnDelete1.Enabled = True
    chkReimbursed.Visible = True
    'Disables the buttons for new expense
    btnSave.Enabled = False
    btnClear.Enabled = False
    'Sets description label for the form and sets the textbox for trackingnumber to read only
    lblDescription.Text = "Travel Expense - View and Edit"
    txtTrackingNumber.ReadOnly = True
    cboName.Enabled = False
    'Sets text in the label as the ID for loading data
    lblID.Text = lstView.SelectedItems(0).Text
    'Opens the connection and declares a new OleDbCommand and Reader
    connection.Open()
    Dim dt As New DataTable()
    'Fills datatable with data from the investments table
    Using da As New OleDbDataAdapter("SELECT * FROM [Travel Expenses] WHERE ID = " & lblID.Text, connection)
      da.Fill(dt)
    End Using
    'Closes the connection
    connection.Close()
    'Sets values in textboxes to data in datatable
    txtTrackingNumber.Text = dt.Rows(0)("TrackingNumber").ToString
    cboName.Text = dt.Rows(0)("FullName").ToString
    dtpDate.Value = dt.Rows(0)("DateIncurred")
    cboExpenseType.Text = dt.Rows(0)("ExpenseType").ToString
    txtDescription.Text = dt.Rows(0)("Description").ToString
    chkBillClient.Checked = dt.Rows(0)("BillClient")
  End If
End Sub

```

```

chkReimburse.Checked = dt.Rows(0)("Reimburse")
cboPaymentMethod.Text = dt.Rows(0)("PaymentMethod").ToString
txtForeign.Text = dt.Rows(0)("ForeignAmount")
cboCurrency.Text = dt.Rows(0)("ForeignCurrency").ToString
txtExchangeRate.Text = dt.Rows(0)("ExchangeRate")
lblNZDAmount.Text = dt.Rows(0)("NZDAmount")
chkReimbursed.Checked = dt.Rows(0)("Reimbursed")
'Updates label to show status
lblStatus.Text = "Record successfully loaded. Please make any desired changes and click 'Update'."
lblStatus.ForeColor = Color.Green
'Output error if no item is selected
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
'Runs Validation function and sees if it returns true
If Validation() Then
'Declares the new OleDbCommand and gives it parameters values
Dim cmd As New OleDbCommand("UPDATE [Travel Expenses] SET FullName = ?, DateIncurred = ?, ExpenseType = ?, Description = ?, " & _
                           "BillClient = ?, Reimburse = ?, PaymentMethod = ?, ForeignAmount = ?, ForeignCurrency = ?, " & _
                           "ExchangeRate = ?, NZDAmount = ?, Reimbursed = ? WHERE ID = " & lblID.Text, connection)
cmd.Parameters.AddWithValue("@p1", cboName.Text)
cmd.Parameters.AddWithValue("@p2", dtpDate.Value.Date)
cmd.Parameters.AddWithValue("@p3", cboExpenseType.Text)
cmd.Parameters.AddWithValue("@p4", txtDescription.Text)
cmd.Parameters.AddWithValue("@p5", chkBillClient.Checked)
cmd.Parameters.AddWithValue("@p6", chkReimburse.Checked)
cmd.Parameters.AddWithValue("@p7", cboPaymentMethod.Text)
cmd.Parameters.AddWithValue("@p8", txtForeign.Text)
cmd.Parameters.AddWithValue("@p9", cboCurrency.Text)
cmd.Parameters.AddWithValue("@p10", txtExchangeRate.Text)
cmd.Parameters.AddWithValue("@p11", lblNZDAmount.Text)
cmd.Parameters.AddWithValue("@p12", chkReimbursed.Checked)
'Executes the command and closes form
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Output to show success
lblStatus.Text = "Record successfully updated. Please enter a new expense or select another option."
lblStatus.ForeColor = Color.Green
'Runs subroutine to repopulate listview and calls subroutine to reset the form
populateListView()
Call btnNew_Click(sender, e)
End If

```

```

End Sub

Private Sub btnDelete1_Click(sender As Object, e As EventArgs) Handles btnDelete1.Click
    'Displays a MsgBox asking user to confirm to delete the record
    Dim Response = MsgBox("Are you sure you want to delete this Travel Expense record?", MsgBoxStyle.YesNo, "Delete Record")
    'If user responds 'Yes' to the MsgBox
    If Response = MsgBoxResult.Yes Then
        'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
        Dim cmd As New OleDbCommand("DELETE FROM [Travel Expenses] WHERE ID = " & lblID.Text, connection)
        'Opens connection, executes the command and closes the connection
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Repopulates the listview and resets the form
        populateListView()
        Call btnNew_Click(sender, e)
        'Updates label to show status
        lblDataStatus.Text = "Record deleted."
        lblDataStatus.ForeColor = Color.Green
        lblDataStatus.Visible = True
    End If
End Sub

Private Sub btnDelete2_Click(sender As Object, e As EventArgs) Handles btnDelete2.Click
    'Checks if an item has been selected or not
    If lstView.SelectedItems.Count > 0 Then
        'Displays a MsgBox asking user to confirm to delete the record
        Dim Response = MsgBox("Are you sure you want to delete this Travel Expense record?", MsgBoxStyle.YesNo, "Delete Record")
        If Response = MsgBoxResult.Yes Then
            'Checks if an item is selected in the listview or not if user response is 'Yes'
            If lstView.SelectedItems.Count > 0 Then
                'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
                Dim ID As String = lstView.SelectedItems(0).Text
                Dim cmd As New OleDbCommand("DELETE FROM [Travel Expenses] WHERE ID = " & ID, connection)
                'Opens connection, executes the command and closes the connection
                connection.Open()
                cmd.ExecuteNonQuery()
                connection.Close()
                'Repopulates the listview and resets the form
                populateListView()
                Call btnNew_Click(sender, e)
                'Updates label to show status
                lblDataStatus.Text = "Record deleted."
                lblDataStatus.ForeColor = Color.Green
                lblDataStatus.Visible = True
            End If
        End If
    End If
End Sub

```

```

        End If
    End If
    'Output error if no record has been selected
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles btnNew.Click
    'Brings relevant buttons to the front
    btnSave.BringToFront()
    btnSave.Enabled = True
    btnClear.BringToFront()
    btnClear.Enabled = True
    'Disables the buttons for updating expense and hides the reimbursed checkbox
    btnReimburseAll.Visible = False
    btnUpdate.Enabled = False
    btnDelete1.Enabled = False
    chkReimbursed.Visible = False
    'Runs subroutine to clear the form
    clearForm()
    'Updates the labels to show user what to do
    lblDescription.Text = "Travel Expense - New"
    lblStatus.Text = "Please enter relevant data and press save."
    lblStatus.ForeColor = Color.FromArgb(22, 88, 153)
End Sub

Private Sub btnReimburseAll_Click(sender As Object, e As EventArgs) Handles btnReimburseAll.Click
    'Displays a MsgBox asking user to confirm that all records are reimbursed
    Dim Response = MsgBox("Have all records for the Travel Expense #" & txtTrackingNumber.Text & " been reimbursed fully?", _
                           MsgBoxStyle.YesNo, "Confirmation")
    If Response = MsgBoxResult.Yes Then
        'Declares the new OleDbCommand and gives it parameters values if user selects 'Yes'
        Dim cmd As New OleDbCommand("UPDATE [Travel Expenses] SET Reimbursed = True WHERE TrackingNumber = '" & _
                                    txtTrackingNumber.Text & "'", connection)
        'Executes the command and closes form
        connection.Open()
        cmd.ExecuteNonQuery()
        connection.Close()
        'Output to show success
        lblStatus.Text = "Record successfully updated. Please enter a new expense or select another option."
        lblStatus.ForeColor = Color.Green
        populateListView()
        Call btnNew_Click(sender, e)
    End If
End Sub

```

```

Private Sub btnClear_Click(sender As Object, e As EventArgs) Handles btnClear.Click
    'Runs subroutine to clear the form
    clearForm()
End Sub

//PRINTING
Private Sub btnPrint_Click(sender As Object, e As EventArgs) Handles btnPrint.Click
    'Declares variables to be used in query
    Dim TrackingNumber As String = txtTrackingNumber.Text
    Dim Name As String = cboName.Text
    'Declares OleDbCommand, Data Adapter and Dataset
    Dim cmd As New OleDbCommand("SELECT Format(DateIncurred, 'dd/mm/yy'), Description, BillClient, ExpenseType, ForeignCurrency " & _
                                "& '$' & ForeignAmount, ExchangeRate, '$' & NZDAmount, Reimbursed From [Travel Expenses] WHERE " & _
                                "TrackingNumber = '" & TrackingNumber & "' ORDER BY DateIncurred", connection)
    Dim da As New OleDbDataAdapter(cmd)
    Dim dt As New DataTable
    'Fills dataset with data from table and outputs it to datagridview
    da.FillSchema(dt, SchemaType.Source)
    dt.Columns(2).DataType = GetType(String)
    dt.Columns(5).DataType = GetType(String)
    dt.Columns(7).DataType = GetType(String)
    da.Fill(dt)
    'Displays data on datagridview and sets the column names
    DataGridView.DataSource = dt
    With DataGridView
        .Columns(0).HeaderCell.Value = "Date"
        .Columns(2).HeaderCell.Value = "BC"
        .Columns(3).HeaderCell.Value = "Type"
        .Columns(4).HeaderCell.Value = "Amount"
        .Columns(5).HeaderCell.Value = "XRT"
        .Columns(6).HeaderCell.Value = "NZD"
        .Columns(7).HeaderCell.Value = "Repaid"
    End With
    'Loops through datagridview and replaces True with Yes and False with Nothing for BillClient and
    'False with No for Repaid. Calculates total for the NZD and Reimbursed column (if it is false) of the datagridview
    Dim totalDec As Decimal = 0
    Dim repaidDec As Decimal = 0
    For i = 0 To DataGridView.Rows.Count - 1
        'Checks if cell for Bill Client contains True
        If DataGridView.Rows(i).Cells(2).Value = "True" Then
            DataGridView.Rows(i).Cells(2).Value = "Yes"
        Else : DataGridView.Rows(i).Cells(2).Value = ""
        End If
        'Checks if cell for Repaid contains True

```

```

If DataGridView.Rows(i).Cells(7).Value = "True" Then
    DataGridView.Rows(i).Cells(7).Value = "Yes"
Else
    DataGridView.Rows(i).Cells(7).Value = "No"
    repaidDec += DataGridView.Rows(i).Cells(6).Value
End If
'Sums the values in the NZD column
totalDec = totalDec + DataGridView.Rows(i).Cells(6).Value
Next i
'Rounds the total variables to 2 decimal points and converts them to string with $ symbol
totalDec = Decimal.Round(totalDec, 2, MidpointRounding.AwayFromZero)
repaidDec = Decimal.Round(repaidDec, 2, MidpointRounding.AwayFromZero)
Dim totalStr As String = "$" & CStr(totalDec)
Dim repaidStr As String = "$" & CStr(repaidDec)
'Adds new row to DataTable to update the datagridview. Sets values in rows then adds
Dim newRow As DataRow = dt.NewRow
For i = 0 To 5
    newRow.Item(i) = ""
Next
newRow.Item(6) = totalStr
newRow.Item(7) = repaidStr
dt.Rows.Add(newRow)
'Runs function to set up printing
If SetupThePrinting(TrackingNumber & " - " & Name) Then
    'Declares and opens print preview dialog
    Dim ppd As PrintPreviewDialog = New PrintPreviewDialog()
    ppd.Document = pntDocument
    ppd.Height = Screen.PrimaryScreen.Bounds.Height / 1.5
    ppd.Width = Screen.PrimaryScreen.Bounds.Width / 2.5
    ppd.FindForm().StartPosition = FormStartPosition.CenterScreen
    ppd.Text = "Travel Expenses"
    ppd.ShowDialog()
End If
End Sub

Private Sub pntDocument_PrintPage(ByVal sender As System.Object, ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles pntDocument.PrintPage
    'Checks if there are any more pages
    Dim more As Boolean = dgvPrinter.DrawDataGridView(e.Graphics)
    If more = True Then
        e.HasMorePages = True
    End If
End Sub

Private Function SetupThePrinting(ByVal title As String) As Boolean

```

```

'Declares print dialog and sets properties
Dim pd As PrintDialog = New PrintDialog()
pd.AllowCurrentPage = False
pd.AllowPrintToFile = False
pd.AllowSelection = False
pd.AllowSomePages = False
pd.PrintToFile = False
pd.ShowHelp = False
pd.ShowNetwork = False
'Function returns false if user cancels
If pd.ShowDialog() <> DialogResult.OK Then
    Return False
End If
'Sets properties for the print document and sets the data
pntDocument.DocumentName = "Travel Expense"
pntDocument.PrinterSettings = pd.PrinterSettings
pntDocument.DefaultPageSettings = pd.PrinterSettings.DefaultPageSettings
pntDocument.DefaultPageSettings.Margins = New Margins(20, 20, 20, 20)
dgvPrinter = New DataGridViewPrinter(DataGridView, pntDocument, True, True, title, New Font("Segoe UI", 18, FontStyle.Bold,
GraphicsUnit.Point), Color.Black, True)
Return True
End Function

//SUBROUTINES AND FUNCTIONS (NOT CONTROL BASED)
Private Function Validation()
    'Checks if the textboxes and combobox contain text
    If txtTrackingNumber.Text <> "" And cboName.Text <> "" And cboExpenseType.Text <> "" And cboPaymentMethod.Text <> "" And _
        txtDescription.Text <> "" And txtForeign.Text <> "" And txtExchangeRate.Text <> "" And lblNZDAmount.Text <> "" And _
        cboCurrency.Text.Length = 3 Then 'Checks that the length of the text in the currency combobox is 3
        Return True
    Else
        'Output to show validation was not successful
        lblStatus.ForeColor = Color.Red
        lblStatus.Text = "Please ensure all fields contain valid data."
        Return False
    End If
End Function

Private Sub populateCBO()
    'Declares DataSet and fills it up with data from Username Column in database
    Dim ds As New DataSet()
    Using da As New OleDbDataAdapter("SELECT FullName FROM Accounts", connection)
        da.Fill(ds, "FullName")
    End Using
    'Loads DataSet data into a DataView, sorts it then writes it to a DataTable

```

```

Dim view As New DataView(ds.Tables("FullName"))
view.Sort = "FullName"
Dim sorted As DataTable = view.ToTable()
'Populates comboboxes with contents of the new Datatable
With cboName
    .DisplayMember = "FullName"
    .DataSource = view
    .SelectedIndex = 0
End With
End Sub

Private Sub populateListView()
    'Clears and enables the listview
    lstView.Items.Clear()
    lstView.Enabled = True
    'Opens the connection and declares a new OleDbCommand and Reader
    connection.Open()
    Dim cmd As New OleDbCommand("SELECT * From [Travel Expenses] WHERE TrackingNumber = '" &
        txtTrackingNumber.Text & "' ORDER BY DateIncurred", connection)
    Dim dr As OleDbDataReader = cmd.ExecuteReader
    'Loops the Travel Expenses table of the database and adds each record to a new ListViewItem
    Do While dr.Read()
        Dim newitem As New ListViewItem(dr.Item("ID").ToString)
        newitem.SubItems.Add(dr.Item("DateIncurred"))
        newitem.SubItems.Add(dr.Item("Description").ToString)
        newitem.SubItems.Add(dr.Item("ExpenseType"))
        newitem.SubItems.Add(dr.Item("NZDAmount"))
        newitem.SubItems.Add(dr.Item("Reimbursed"))
        lstView.Items.Add(newitem)
    Loop
    'Closes the connection
    connection.Close()
End Sub

Private Sub clearForm()
    'Clears the controls that require resetting
    dtpDate.Value = DateTime.Today
    txtDescription.Text = ""
    chkBillClient.Checked = False
    chkReimburse.Checked = False
    txtForeign.Text = ""
    txtExchangeRate.Text = cboCurrency.Text & " to NZD"
    lblNZDAmount.Text = ""
    lblDataStatus.Visible = False
    'Sets the indexes of the comboboxes to show no text

```

```

cboExpenseType.SelectedIndex = -1
cboPaymentMethod.SelectedIndex = -1
End Sub

Private Sub Travel_Expense_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Runs subroutine to repopulate listview on the overview form
    Travel_Expense__Overview.populateListView()
    'Closes form completely
    Me.Dispose()
End Sub

//OTHER CONTROL BASED SUBROUTINES
Private Sub txtTrackingNumber_TextChanged(sender As Object, e As EventArgs) Handles txtTrackingNumber.TextChanged
    'Converts the text in textbox to uppercase and sets it
    Dim Tracking As String = txtTrackingNumber.Text
    txtTrackingNumber.Text = Tracking.ToUpper()
    txtTrackingNumber.Select(txtTrackingNumber.Text.Length, 0)
    Try
        'Opens the connection and declares a new OleDbCommand and datatable
        Dim TrackingNumber As String = txtTrackingNumber.Text
        connection.Open()
        Dim dt As New DataTable()
        'Uses data from the travel expense application table to fill in the FullName textbox
        Using da As New OleDbDataAdapter("SELECT FullName FROM TEA WHERE TrackingNumber = '" & TrackingNumber & "'", connection)
            da.Fill(dt)
        End Using
        'Closes the connection
        connection.Close()
        'Sets values in combobox to data in datatable
        cboName.Text = dt.Rows(0)("FullName").ToString
        cboName.Enabled = False
        populateListView()
    Catch
        'Clears the listview and enables the combobox if an error occurred
        lstView.Items.Clear()
        cboName.Enabled = True
    End Try
End Sub

Private Sub cboPaymentMethod_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cboPaymentMethod.SelectedIndexChanged
    'Checks which payment method the user has selected
    If cboPaymentMethod.Text = "Personal" Then
        'Sets the reimburse checkbox to true
        chkReimburse.Enabled = True
        chkReimburse.Checked = True
    End If
End Sub

```

```

ElseIf cboPaymentMethod.Text = "Company" Then
    'Disables and sets checkbox to unchecked
    chkReimburse.Checked = False
    chkReimburse.Enabled = False
Else : chkReimburse.Enabled = True
End If
End Sub

Private Sub Calculation_TextChanged(sender As Object, e As EventArgs) Handles txtForeign.TextChanged, txtExchangeRate.TextChanged
    'Adds the values in the textboxes for estimated costs and adds them together
    Dim foreign, exchange, nzd As Double
    foreign = Val(txtForeign.Text)
    exchange = Val(txtExchangeRate.Text)
    nzd = foreign * exchange
    lblNZDAmount.Text = nzd
End Sub

Private Sub cboCurrency_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles cboCurrency.KeyPress
    'Does not allow numbers, punctuation, etc. into the combobox
    If Not Char.IsLetter(e.KeyChar) Then
        e.Handled = True
    End If
    'Allow backspace in textbox
    If e.KeyChar = Chr(&H8) Then
        e.Handled = False
    End If
End Sub

Private Sub cboCurrency_TextChanged(sender As Object, e As EventArgs) Handles cboCurrency.TextChanged
    'Converts the text in the combobox to full capitals
    Dim currency As String = cboCurrency.Text
    currency = currency.ToUpper
    cboCurrency.Text = currency
    'Sets cursor to end of combobox
    cboCurrency.Select(cboCurrency.Text.Length, 0)
End Sub

Private Sub cboCurrency_SelectedIndexChanged(sender As Object, e As EventArgs) Handles cboCurrency.SelectedIndexChanged
    'Sets the text to the allow the user to know what to input (e.g. RMB to NZD)
    txtExchangeRate.Text = cboCurrency.Text & " to NZD"
End Sub

Private Sub txtExchangeRate_Click(sender As Object, e As EventArgs) Handles txtExchangeRate.Click
    If txtExchangeRate.Text.Contains("NZD") Then
        'Clears the textbox upon clicking if text contains NZD

```

```

        txtExchangeRate.Text = ""
    End If
End Sub

Private Sub txtExchangeRate_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtExchangeRate.KeyPress
    'Allows numbers and decimals to be input into the textbox
    Dim dot As Integer
    If Not e.KeyChar = "." And Not Char.IsNumber(e.KeyChar) Then
        e.Handled = True
    End If
    'Allow backspace in textbox
    If e.KeyChar = Chr(&H8) Then
        e.Handled = False
    End If
    'Checks if the text in the textbox already contains a dot or not
    If e.KeyChar = "." And Not txtExchangeRate.Text.IndexOf(".") = -1 Then
        e.Handled = False
        dot = txtExchangeRate.Text.IndexOf(".")
        If dot > -1 Then
            e.Handled = True
        End If
    End If
End Sub

Private Sub txtForeign_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtForeign.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub

Private Sub lblNZDAmount_TextChanged(sender As Object, e As EventArgs) Handles lblNZDAmount.TextChanged
    Try
        'Rounds the value in the label to 2 decimal points
        lblNZDAmount.Text = Decimal.Round(lblNZDAmount.Text, 2, MidpointRounding.AwayFromZero)
    Catch
    End Try
End Sub
End Class

```

Wages - Overview.vb

```

Imports System.Data.OleDb
Imports Odin.ListViewSort
Imports System.Drawing.Printing

```

```

Public Class Wages__Overview
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Dim dgvPrinter As DataGridViewPrinter
    Private m_SortingColumn As ColumnHeader

    Private Sub Wages__Overview_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Runs subroutines to populate the listview and the combobox
        populateListView()
        populateCBO()
        'Sets minimum value of the DateTo datepicker to the value of the DateFrom datepicker
        dtpDateTo.MinDate = dtpDateFrom.Value.Date
        'Sets font on the datagridview for printing
        DataGridView.Font = New Font("Segoe UI", 11.5)
        'Focuses form
        PictureBox.Select()
    End Sub

    Public Sub populateListView()
        'Clears and enables the listview
        lstView.Items.Clear()
        lstView.Enabled = True
        'Opens the connection and declares a new OleDbCommand and Reader
        connection.Open()
        Dim cmd As New OleDbCommand("SELECT * From Wages ORDER BY ID", connection)
        Dim dr As OleDbDataReader = cmd.ExecuteReader
        'Loops the Accounts table of the database and adds each record to a new ListViewItem
        Do While dr.Read()
            Dim newitem As New ListViewItem(dr.Item("ID").ToString)
            newitem.SubItems.Add(dr.Item("StartDate"))
            newitem.SubItems.Add(dr.Item("EndDate"))
            newitem.SubItems.Add(dr.Item("FullName").ToString)
            newitem.SubItems.Add(dr.Item("DatePaid"))
            newitem.SubItems.Add(dr.Item("NetPay").ToString)
            lstView.Items.Add(newitem)
        Loop
        'Closes the connection
        connection.Close()
    End Sub

    Private Sub populateCBO()
        'Declares DataSet and fills it up with data from Username Column in database
        Dim ds As New DataSet()
        Using da As New OleDbDataAdapter("SELECT FullName FROM Accounts", connection)
            da.Fill(ds, "FullName")
        End Using
    End Sub

```

```

'Loads DataSet data into a DataView, sorts it then writes it to a DataTable
Dim view As New DataView(ds.Tables("FullName"))
view.Sort = "FullName"
Dim sorted As DataTable = view.ToTable()
'Populates comboboxes with contents of the new Datatable
With cboName
    .DisplayMember = "FullName"
    .DataSource = view
    .SelectedIndex = 0
End With
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles btnNew.Click
    'Opens the form for a new wage payment
    Wages__New.ShowDialog()
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
    'Checks if an item is selected in the listview
    If lstView.SelectedItems.Count > 0 Then
        'Displays a MsgBox asking user to confirm to delete the record
        Dim Response = MsgBox("Are you sure you want to delete this Wage Payment?", MsgBoxStyle.YesNo, "Delete Wage Payment")
        If Response = MsgBoxResult.Yes Then
            'Checks if an item is selected in the listview or not
            'Declares a new OleDbCommand which deletes the record which contains the text from the listview item
            Dim ID As String = lstView.SelectedItems(0).Text
            Dim cmd As New OleDbCommand("DELETE FROM Wages WHERE ID = " & ID, connection)
            'Opens connection, executes the command and closes the connection
            connection.Open()
            cmd.ExecuteNonQuery()
            connection.Close()
            'Repopulates the listview
            populateListView()
        End If
        'Output to tell user to select a record
    Else : MsgBox("Please select a record from the table.")
    End If
End Sub

Private Sub btnEdit_Click(sender As Object, e As EventArgs) Handles btnEdit.Click
    'Checks if an item is selected in the listview or not
    If lstView.SelectedItems.Count > 0 Then
        'Passes value of ID to Edit form and opens it
        Dim ID As String = lstView.SelectedItems(0).Text
        Wages__View_and_Edit.txtID.Text = ID
    End If
End Sub

```

```

Wages__View_and_Edit.ShowDialog()
'Output to tell user to select a record
Else : MsgBox("Please select a record from the table.")
End If
End Sub

Private Sub btnFilterClear_Click(sender As Object, e As EventArgs) Handles btnFilterClear.Click
'Runs subroutine to clear and populate the listview
populateListView()
populateCBO()
chkFilterDate.Checked = False
chkFilterName.Checked = False
dtpDateFrom.Value = DateTime.Today
dtpDateTo.Value = DateTime.Today
lblFilterStatus.Visible = False
End Sub

Private Sub Wages__Overview_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
'Opens main menu form upon closing of this form
MainMenu.Show()
End Sub

Private Sub dtpDateFrom_ValueChanged(sender As Object, e As EventArgs) Handles dtpDateFrom.ValueChanged
'Sets minimum value of the DateTo datepicker to the value of the DateFrom datepicker
dtpDateTo.MinDate = dtpDateFrom.Value.Date
End Sub

Private Sub btnFilter_Click(sender As Object, e As EventArgs) Handles btnFilter.Click
'Checks which checkboxes have been checked and runs subroutines accordingly
If chkFilterDate.Checked = True And chkFilterName.Checked = True Then
    filterDateName()
ElseIf chkFilterDate.Checked = True Then
    filterDate()
ElseIf chkFilterName.Checked = True Then
    filterName()
Else
    'Feedback to show none of the checkboxes were checked
    lblFilterStatus.Text = "Please select the filters."
    lblFilterStatus.ForeColor = Color.Red
    lblFilterStatus.Visible = True
End If
End Sub

Private Sub filterDateName()
'Clears the listview

```

```

lstView.Items.Clear()
'Opens connection, declares OleDbCommand and OleDbReader and executes
connection.Open()
Dim cmd As New OleDbCommand("SELECT * From Wages WHERE (StartDate BETWEEN @DateFrom AND @DateTo " & _
                           "OR EndDate BETWEEN @DateFrom AND @DateTo) AND FullName = @FullName ORDER BY ID", connection)
cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
cmd.Parameters.AddWithValue("@FullName", cboName.Text)
Dim dr As OleDbDataReader = cmd.ExecuteReader
'Loops the Accounts table of the database and adds each record to a new ListViewItem
Do While dr.Read()
    Dim newitem As New ListViewItem(dr.Item("ID").ToString)
    newitem.SubItems.Add(dr.Item("StartDate"))
    newitem.SubItems.Add(dr.Item("EndDate"))
    newitem.SubItems.Add(dr.Item("FullName").ToString)
    newitem.SubItems.Add(dr.Item("DatePaid"))
    newitem.SubItems.Add(dr.Item("NetPay").ToString)
    lstView.Items.Add(newitem)
Loop
'Closes the connection and runs subroutine to check if data was found
connection.Close()
filterCheck()
End Sub

Private Sub filterDate()
'Clears the listview
lstView.Items.Clear()
'Opens connection, declares OleDbCommand and OleDbReader and executes
connection.Open()
Dim cmd As New OleDbCommand("SELECT * From Wages WHERE StartDate BETWEEN @DateFrom AND @DateTo " & _
                           "OR EndDate BETWEEN @DateFrom AND @DateTo ORDER BY ID", connection)
cmd.Parameters.AddWithValue("@DateFrom", dtpDateFrom.Value.Date)
cmd.Parameters.AddWithValue("@DateTo", dtpDateTo.Value.Date)
Dim dr As OleDbDataReader = cmd.ExecuteReader
'Loops the Accounts table of the database and adds each record to a new ListViewItem
Do While dr.Read()
    Dim newitem As New ListViewItem(dr.Item("ID").ToString)
    newitem.SubItems.Add(dr.Item("StartDate"))
    newitem.SubItems.Add(dr.Item("EndDate"))
    newitem.SubItems.Add(dr.Item("FullName").ToString)
    newitem.SubItems.Add(dr.Item("DatePaid"))
    newitem.SubItems.Add(dr.Item("NetPay").ToString)
    lstView.Items.Add(newitem)
Loop
'Closes the connection and runs subroutine to check if data was found

```

```

connection.Close()
filterCheck()
End Sub

Private Sub filterName()
    'Clears the listview
    lstView.Items.Clear()
    'Opens connection, declares OleDbCommand and OleDbReader and executes
    connection.Open()
    Dim cmd As New OleDbCommand("Select * From Wages WHERE FullName = '" & cboName.Text & "' ORDER BY ID", connection)
    Dim dr As OleDbDataReader = cmd.ExecuteReader
    'Loops the Accounts table of the database and adds each record to a new ListViewItem
    Do While dr.Read()
        Dim newitem As New ListViewItem(dr.Item("ID").ToString)
        newitem.SubItems.Add(dr.Item("StartDate"))
        newitem.SubItems.Add(dr.Item("EndDate"))
        newitem.SubItems.Add(dr.Item("FullName").ToString)
        newitem.SubItems.Add(dr.Item("DatePaid"))
        newitem.SubItems.Add(dr.Item("NetPay").ToString)
        lstView.Items.Add(newitem)
    Loop
    'Closes the connection and runs subroutine to check if data was found
    connection.Close()
    filterCheck()
End Sub

Private Sub filterCheck()
    'Checks if any data was found
    If lstView.Items.Count = 0 Then
        'Output if no data was found
        lblFilterStatus.Text = "No data found."
        lblFilterStatus.ForeColor = Color.Red
        lblFilterStatus.Visible = True
        populateListView()
    ElseIf lstView.Items.Count <> 0 Then
        'Output if data was found
        lblFilterStatus.Text = "Data successfully filtered."
        lblFilterStatus.ForeColor = Color.Green
        lblFilterStatus.Visible = True
    End If
End Sub

Private Sub lstView_ColumnClick(ByVal sender As Object, ByVal e As System.Windows.Forms.ColumnEventArgs) Handles lstView.ColumnClick
    'Get the new sorting column to be sorted
    Dim new_sorting_column As ColumnHeader = lstView.Columns(e.Column)

```

```

'Find out the sorting order
Dim sort_order As System.Windows.Forms.SortOrder
If m_SortingColumn Is Nothing Then
    'Sort column by ascending order
    sort_order = SortOrder.Ascending
Else
    'Checks if sorting column is the same one
    If new_sorting_column.Equals(m_SortingColumn) Then
        'If it is the same column then the sorting order is switched around
        If m_SortingColumn.Text.StartsWith("> ") Then
            sort_order = SortOrder.Descending
        Else
            sort_order = SortOrder.Ascending
        End If
    Else
        'Sort by ascending value
        sort_order = SortOrder.Ascending
    End If
    'Removes the old sort column indicator
    m_SortingColumn.Text = m_SortingColumn.Text.Substring(2)
End If
'Displays the new sorting column indicator
m_SortingColumn = new_sorting_column
If sort_order = SortOrder.Ascending Then
    m_SortingColumn.Text = "> " & m_SortingColumn.Text
Else
    m_SortingColumn.Text = "< " & m_SortingColumn.Text
End If
'Creates the comparer and sorts the listview accordingly
lstView.ListViewItemSorter = New clsListviewSorter(e.Column, sort_order)
lstView.Sort()
End Sub

Private Sub btnPrint_Click(sender As Object, e As EventArgs) Handles btnPrint.Click
    'Declares query for the OleDbCommand
    Dim query As String = "SELECT Format(StartDate, 'dd/mm/yy') & ' - ' & Format(EndDate, 'dd/mm/yy'), " &
        "FullName, Format(DatePaid, 'dd/mm/yy'), '$' & GrossPay, '$' & PAYE, '$' & KiwiSaver, '$' & NetPay FROM Wages"
    'Checks which radio button is checked
    If rdbAll.Checked = True Then
        'Opens connection, declares OleDbCommand and OleDbDataAdapter. Fills dataset
        connection.Open()
        Dim cmd As New OleDbCommand(query, connection)
        Dim da As OleDbDataAdapter = New OleDbDataAdapter(cmd)
        Dim ds As DataSet = New DataSet()

```

```

da.Fill(ds, "Wages")
'Sets datasource in the data grid view and runs subroutine to change header names
DataGridView.DataSource = ds.Tables("Wages").DefaultView
formatDGV()
connection.Close()
'Runs function to setup the printing
If SetupThePrinting("Wages - All") Then
    'Declares PrintPreviewDialog then shows it accordingly
    Dim ppd As PrintPreviewDialog = New PrintPreviewDialog()
    ppd.Document = pntDocument
    ppd.Height = Screen.PrimaryScreen.Bounds.Height / 1.5
    ppd.Width = Screen.PrimaryScreen.Bounds.Width / 2.5
    ppd.FindForm().StartPosition = FormStartPosition.CenterScreen
    ppd.Text = "Wages"
    ppd.ShowDialog()
End If
ElseIf rdbFiltered.Checked = True Then
    connection.Open()
    Dim ds As DataSet = New DataSet()
    'Loops through all items in the listview
    For i = 0 To lstView.Items.Count - 1
        'Declares ID, OleDbCommand and adapter. Fills dataset
        Dim ID As Integer = lstView.Items(i).Text
        Dim cmd As New OleDbCommand(query & " WHERE ID =" & ID, connection)
        Dim da As OleDbDataAdapter = New OleDbDataAdapter(cmd)
        da.Fill(ds, "Wages")
    Next i
    'Sets datasource in the data grid view , runs subroutine to change header names
    DataGridView.DataSource = ds.Tables("Wages").DefaultView
    formatDGV()
    connection.Close()
    'Runs function to setup the printing
    If SetupThePrinting("Wages - Filtered") Then
        'Declares PrintPreviewDialog then shows it accordingly
        Dim ppd As PrintPreviewDialog = New PrintPreviewDialog()
        ppd.Document = pntDocument
        ppd.Height = Screen.PrimaryScreen.Bounds.Height / 1.5
        ppd.Width = Screen.PrimaryScreen.Bounds.Width / 2.5
        ppd.FindForm().StartPosition = FormStartPosition.CenterScreen
        ppd.Text = "Wages"
        ppd.ShowDialog()
    End If
Else : MsgBox("Please select a radio button.") 'Output to user asking to select a radio button
End If
End Sub

```

```

Private Sub formatDGV()
    'Formatting the headercells for the datagridview
    With DataGridView
        .Columns(0).HeaderCell.Value = "Dates"
        .Columns(1).HeaderCell.Value = "Name"
        .Columns(2).HeaderCell.Value = "Date Paid"
        .Columns(3).HeaderCell.Value = "Gross Pay"
        .Columns(4).HeaderCell.Value = "PAYE"
        .Columns(5).HeaderCell.Value = "KiwiSaver"
        .Columns(6).HeaderCell.Value = "Net Pay"
    End With
End Sub

Private Sub pntDocument_PrintPage(ByVal sender As System.Object, ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles pntDocument.PrintPage
    'Checks if there are any more pages
    Dim more As Boolean = dgvPrinter.DrawDataGridView(e.Graphics)
    If more = True Then
        e.HasMorePages = True
    End If
End Sub

Private Function SetupThePrinting(ByVal title As String) As Boolean
    'Declares print dialog and sets properties
    Dim pd As PrintDialog = New PrintDialog()
    pd.AllowCurrentPage = False
    pd.AllowPrintToFile = False
    pd.AllowSelection = False
    pd.AllowSomePages = False
    pd.PrintToFile = False
    pd.ShowHelp = False
    pd.ShowNetwork = False
    'Function returns false if user cancels
    If pd.ShowDialog() <> DialogResult.OK Then
        Return False
    End If
    'Sets properties for the print document and sets the data
    pntDocument.DocumentName = "Wages"
    pntDocument.PrinterSettings = pd.PrinterSettings
    pntDocument.DefaultPageSettings = pd.PrinterSettings.DefaultPageSettings
    pntDocument.DefaultPageSettings.Margins = New Margins(20, 20, 20, 20)
    dgvPrinter = New DataGridViewPrinter(DataGridView, pntDocument, True, True, title, New Font("Segoe UI", 18, FontStyle.Bold,
    GraphicsUnit.Point), Color.Black, True)
    Return True
End Function

```

```
End Function  
End Class
```

Wages - New.vb

```
Imports System.Data.OleDb  
  
Public Class Wages__New  
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)  
  
    Private Sub Wages__New_Load(sender As Object, e As EventArgs) Handles MyBase.Load  
        'Runs subroutine to populate the combobox  
        populateCBO()  
        'Focuses the form  
        PictureBox.Select()  
        'Sets minimum value of the End datepicker to the value of the Start datepicker  
        dtpEnd.MinDate = dtpStart.Value.Date  
    End Sub  
  
    Private Sub btnAdd_Click(sender As Object, e As EventArgs) Handles btnAdd.Click  
        'Runs validation function  
        If Validation() Then  
            'Declares the new OleDbCommand  
            Dim cmd As New OleDbCommand("INSERT INTO Wages (StartDate,EndDate, FullName, DatePaid, GrossPay, " &  
                "PAYE, KiwiSaver, NetPay) " &  
                "VALUES (@StartDate, @EndDate, @FullName, @DatePaid, @GrossPay, @PAYE, " &  
                "@KiwiSaver, @NetPay)", connection)  
            'Gives the parameters its values  
            cmd.Parameters.AddWithValue("StartDate", dtpStart.Value.Date)  
            cmd.Parameters.AddWithValue("EndDate", dtpEnd.Value.Date)  
            cmd.Parameters.AddWithValue("FullName", cboName.Text)  
            cmd.Parameters.AddWithValue("DatePaid", dtpPaid.Value.Date)  
            cmd.Parameters.AddWithValue("GrossPay", txtGross.Text)  
            cmd.Parameters.AddWithValue("PAYE", txtPAYE.Text)  
            cmd.Parameters.AddWithValue("KiwiSaver", txtKiwiSaver.Text)  
            cmd.Parameters.AddWithValue("NetPay", txtNet.Text)  
            'Opens database connection, executes query and closes the database  
            connection.Open()  
            cmd.ExecuteNonQuery()  
            connection.Close()  
            'Output to show successful insertion of record  
            lblStatus.ForeColor = Color.Green  
            lblStatus.Text = "Wage payment successfully added into database."  
            'Sets the labels on the overview form to show success  
            Wages__Overview.lblDataStatus.Text = "Record successfully added."
```

```

Wages__Overview.lblDataStatus.Visible = True
'Runs subroutine to clear the form
clearForm()
End If
End Sub

Function Validation()
'Checks if textboxes contain text, if the values of the textboxes are appropriately 'sized' and the dates are correct
If txtGross.Text <> "" And txtPAYE.Text <> "" And txtKiwiSaver.Text <> "" And txtNet.Text <> "" And
    Val(txtNet.Text) > "0" And Val(txtGross.Text) > Val(txtPAYE.Text) And Val(txtGross.Text) > Val(txtKiwiSaver.Text) And _
    dtpEnd.Value.Date >= dtpStart.Value.Date And dtpPaid.Value.Date >= dtpStart.Value.Date Then
    Return True
Else
    'Output to show that one or more of the textboxes have not been filled.
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "Please ensure all fields are filled out correctly."
    Return False
End If
Return False
End Function

Private Sub populateCBO()
'Declares DataSet and fills it up with data from Username Column in database
Dim ds As New DataSet()
Using da As New OleDbDataAdapter("SELECT FullName FROM Accounts", connection)
    da.Fill(ds, "FullName")
End Using
'Loads DataSet data into a DataView, sorts it then writes it to a DataTable
Dim view As New DataView(ds.Tables("FullName"))
view.Sort = "FullName"
Dim sorted As DataTable = view.ToTable()
'Populates comboboxes with contents of the new Datatable
With cboName
    .DisplayMember = "FullName"
    .DataSource = view
    .SelectedIndex = 0
End With
End Sub

Private Sub PayAddition_TextChanged(sender As Object, e As EventArgs) Handles txtGross.TextChanged, txtPAYE.TextChanged,
txtKiwiSaver.TextChanged
'Declares variables and sets values according to the textboxes
Dim g, p, k, t As Double
g = Val(txtGross.Text)
p = Val(txtPAYE.Text)

```

```

k = Val(txtKiwiSaver.Text)
'Subtracts the values in the textboxes for wages
t = g - p - k
txtNet.Text = t
End Sub

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtGross.KeyPress,
    txtPAYE.KeyPress, txtKiwiSaver.KeyPress, txtNet.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub

Private Sub clearForm()
    'Resets controls
    dtpStart.Value = DateTime.Today
    dtpEnd.Value = DateTime.Today
    dtpPaid.Value = DateTime.Today
    txtGross.Text = ""
    txtPAYE.Text = ""
    txtKiwiSaver.Text = ""
    txtNet.Text = ""
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
    'Displays a MsgBox asking user to confirm to close the form
    Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
    If Response = MsgBoxResult.Yes Then
        Me.Dispose()
    End If
End Sub

Private Sub dtpStart_ValueChanged(sender As Object, e As EventArgs) Handles dtpStart.ValueChanged
    'Sets minimum value of the End datepicker to the value of the Start datepicker
    dtpEnd.MinDate = dtpStart.Value.Date
End Sub

Private Sub Wages__New_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Closes form completely and repopulates the listview on the overview form
    Me.Dispose()
    Wages__Overview.populateListView()
End Sub
End Class

```

Wages - View and Edit.vb

```
Imports System.Data.OleDb

Public Class Wages__View_and_Edit
    Dim connection As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & My.Settings.dbLocation)
    Private Sub Wages__View_and_Edit_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Opens the connection and declares a new OleDbCommand and Reader
        Dim ID As String = txtID.Text
        connection.Open()
        Dim dt As New DataTable()
        Using da As New OleDbDataAdapter("SELECT * FROM Wages WHERE ID = " & ID, connection)
            da.Fill(dt)
        End Using
        'Closes the connection
        connection.Close()
        'Sets values in textboxes and date time pickers to data in datatable
        dtpStart.Value = dt.Rows(0)("StartDate")
        dtpEnd.Value = dt.Rows(0)("EndDate")
        txtName.Text = dt.Rows(0)("FullName").ToString
        dtpPaid.Value = dt.Rows(0)("DatePaid")
        txtGross.Text = dt.Rows(0)("GrossPay").ToString
        txtPAYE.Text = dt.Rows(0)("PAYE").ToString
        txtKiwiSaver.Text = dt.Rows(0)("KiwiSaver").ToString
        txtNet.Text = dt.Rows(0)("NetPay").ToString
        'Checks if user is an admin and then enables Name textbox if he/she is
        If My.Settings.accountAdmin = True Then
            txtName.ReadOnly = False
        End If
        'Focuses the form
        PictureBox.Select()
        'Sets minimum value of the End datepicker to the value of the Start datepicker
        dtpEnd.MinDate = dtpStart.Value.Date
    End Sub

    Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
        'Runs Validation function
        If Validation() Then
            'Declares the new OleDbCommand and gives it parameters
            Dim cmd As New OleDbCommand("UPDATE Wages SET StartDate = ?, EndDate = ?, FullName = ?, DatePaid = ?, " &
                "GrossPay = ?, PAYE = ?, KiwiSaver = ?, NetPay = ? WHERE ID =" & txtID.Text, connection)
            cmd.Parameters.AddWithValue("@p1", dtpStart.Value.Date)
            cmd.Parameters.AddWithValue("@p2", dtpEnd.Value.Date)
            cmd.Parameters.AddWithValue("@p3", txtName.Text)
            cmd.Parameters.AddWithValue("@p4", dtpPaid.Value.Date)
        End If
    End Sub

```

```

cmd.Parameters.AddWithValue("@p5", txtGross.Text)
cmd.Parameters.AddWithValue("@p6", txtPAYE.Text)
cmd.Parameters.AddWithValue("@p7", txtKiwiSaver.Text)
cmd.Parameters.AddWithValue("@p8", txtNet.Text)
'Executes the command and closes form
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Output on the overview form for successful updating
Wages__Overview.lblDataStatus.Text = "Record successfully updated."
Wages__Overview.lblDataStatus.Visible = True
'Closes the form
Me.Close()
End If
End Sub

Private Sub btnClose_Click(sender As Object, e As EventArgs) Handles btnClose.Click
'Displays a MsgBox asking user to confirm to close the form
Dim Response = MsgBox("Are you sure you want to close this form?", MsgBoxStyle.YesNo, "Close")
'If user selects 'Yes' then the form is closed
If Response = MsgBoxResult.Yes Then
    Me.Close()
End If
End Sub

Function Validation()
'Checks if all the textboxes contain text, the values of the relevant textboxes and the dates
If txtGross.Text <> "" And txtPAYE.Text <> "" And txtKiwiSaver.Text <> "" And txtNet.Text <> "" And _
    Val(txtNet.Text) > "0" And Val(txtGross.Text) > Val(txtPAYE.Text) And Val(txtGross.Text) > Val(txtKiwiSaver.Text) And _ 
    dtpEnd.Value.Date >= dtpStart.Value.Date And dtpPaid.Value.Date >= dtpStart.Value.Date Then
    Return True
Else
    'Output to show that one or more of the textboxes have not been filled.
    lblStatus.ForeColor = Color.Red
    lblStatus.Text = "Please ensure all fields are filled out correctly."
    Return False
End If
Return False
End Function

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
'Displays a MsgBox asking user to confirm to delete the record
Dim Response = MsgBox("Are you sure you want to delete this Wage Payment?", MsgBoxStyle.YesNo, "Delete Wage Payment")
If Response = MsgBoxResult.Yes Then
    'Declares a new OleDbCommand which deletes the record which contains the text from the textbox if user selects 'Yes'

```

```

Dim ID As String = txtID.Text
Dim cmd As New OleDbCommand("DELETE FROM Wages WHERE ID = " & ID, connection)
'Opens connection, executes the command and closes the connection
connection.Open()
cmd.ExecuteNonQuery()
connection.Close()
'Output on overview form
Wages__Overview.lblDataStatus.Visible = True
Wages__Overview.lblDataStatus.ForeColor = Color.Green
Wages__Overview.lblDataStatus.Text = "Record deleted."
'Closes the form
Me.Close()
End If
End Sub

Private Sub Wages__View_and_Edit_Closing(sender As Object, e As System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    'Runs subroutine to populate the listview on the overview form if this form is closing
    Wages__Overview.populateListView()
    'Closes form completely
    Me.Dispose()
End Sub

Private Sub dtpStart_ValueChanged(sender As Object, e As EventArgs) Handles dtpStart.ValueChanged
    'Sets minimum value of the End datepicker to the value of the Start datepicker
    dtpEnd.MinDate = dtpStart.Value.Date
End Sub

Private Sub PayAddition_TextChanged(sender As Object, e As EventArgs) Handles txtGross.TextChanged, txtPAYE.TextChanged,
txtKiwiSaver.TextChanged
    'Declares variables and gives them values based on their relevant textboxes
    Dim g, p, k, t As Double
    g = Val(txtGross.Text)
    p = Val(txtPAYE.Text)
    k = Val(txtKiwiSaver.Text)
    'Subtracts the values in the textboxes for the wages
    t = g - p - k
    txtNet.Text = t
End Sub

Private Sub txt_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtGross.KeyPress,
    txtPAYE.KeyPress, txtKiwiSaver.KeyPress, txtNet.KeyPress
    'Runs subroutine on textFormatting class to allow numbers, 1 decimal point and 2 numbers after the decimal point
    textFormatting.twoDecimalPoints(sender, e)
End Sub
End Class

```

Testing

Testing Schedule

The testing schedule has been compiled based on form alphabetical order. Video evidence is provided at the end of testing evidence. The testing schedule has been based upon the testing plan included earlier on in this project.

Additionally, invalid data cannot be entered into the program as controls have been programmed to only accept specific inputs. For example, textboxes containing monetary values have been programmed to accept numbers only, one decimal point and two numbers after the decimal point. Furthermore, the size of the controls have been limited to match the maximum length of the fields in the database. Thus, these act as a form of 'validation'.

About.vb

Test	Expected Outcome	Result	Evidence
Close Button	Closes the form	Success	Fig 1.1
Queries & Support Label	Opens Contact Support form	Success	Fig 1.2

AccountManager.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Enter Button with correct password in textbox	N/A	Enables the rest of the form	Success	Fig 2.1
Enter Button with incorrect password in textbox	N/A	Output error on label	Success	Fig 2.2
Add Button with relevant values in textboxes	Normal	Account added to database	Success	Fig 2.3
Add Button with extreme values in the textboxes	Extreme	Account added to database	Success	Fig 2.4
Add Button with no values in the textboxes	Invalid	Output error message	Success	Fig 2.5
Delete Button with confirm checkbox checked	N/A	Account deleted from database	Success	Fig 2.6
Delete Button with confirm checkbox unchecked	N/A	Output error message	Success	Fig 2.7
View Button	N/A	Account information loaded into ListView	Success	Fig 2.8
Clear Button	N/A	Clears the ListView	Success	Fig 2.9
Update Button with relevant values in textboxes	Normal	Account updated	Success	Fig 2.10
Update Button with extreme values in the textboxes	Extreme	Account updated	Success	Fig 2.11
Update Button with no values in the textboxes	Invalid	Output error message	Success	Fig 2.12
Close Button	N/A	Closes Form	Success	Fig 2.13
Lock Controls Button	N/A	Disables the group controls	Success	Fig 2.14

Business Expenses - Overview.vb

Test	Expected Outcome	Result	Evidence
New Expense Button clicked	Business Expense - New Form opened	Success	Fig 3.1
Search Button clicked and existing Tracking Number entered.	Business Expense – View and Edit Form opened with preloaded details	Success	Fig 3.2
Item selected on pending ListView and Delete Button clicked	Record deleted from database and ListView updated	Success	Fig 3.3
Item selected on completed ListView and Delete Button clicked	Record deleted from database and ListView updated	Success	Fig 3.4
Item selected on pending ListView and View or Edit Button clicked	Business Expense – View and Edit Form opened with loaded details	Success	Fig 3.5
Item selected on completed ListView and View or Edit Button clicked	Business Expense – View and Edit Form opened with loaded details	Success	Fig 3.6
Print All Data and Find Total Checkbox checked and Filter Button clicked	Print Dialog opened then Print Preview Dialog opened with correct data with total calculated	Success	Fig 3.7
Print All Data and Find Total Checkbox unchecked and Filter Button clicked	Print Dialog opened then Print Preview Dialog opened with correct data with no total calculated	Success	Fig 3.8
Filter Button clicked with filters applied	The correct data is filtered	Success	Vid 1.0
Print All Data Checkbox checked with other checkboxes checked	The other checkboxes are unchecked	Success	Vid 1.1
Clicking on the header of the columns of the ListView	Sorts the relevant columns	Success	Vid A0

Business Expenses - New.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Tracking Number that exists in the BEA table is entered into textbox	Normal	The relevant data from the BEA record is loaded into the controls	Success	Fig 4.1
Tracking Number that does not exist in the BEA table is entered into textbox	Invalid	The controls are set to blank	Success	Fig 4.2
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 4.3
Save Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated	Success	Fig 4.4
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database and the ListView is updated	Success	Fig 4.5
Save Button is clicked with controls left blank	Invalid	Output error message to user	Success	Fig 4.6
Save Button is clicked but a record with the tracking number already exists	Invalid	Output error message to user	Success	Fig 4.7
User selects 'Company' as Payment Method	N/A	Reimbursement checkbox is unchecked and disabled	Success	Fig 4.8
The text in the Quantity or Unit Price textbox is changed	N/A	The Total Price is automatically calculated	Success	Fig 4.9

Business Expenses - View and Edit.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 5.1
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated	Success	Fig 5.2
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated	Success	Fig 5.3
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated	Success	Fig 5.4
Update Button with no data in controls	Invalid	Output error message to user	Success	Fig 5.5
The text in the Quantity or Unit Price textbox is changed	N/A	The Total Price is automatically calculated	Success	Fig 5.6

Business Expense Applications - Overview.vb

Test	Expected Outcome	Result	Evidence
New Application Button clicked	Business Expense Application - New Form opened	Success	Fig 6.1
Search Button clicked and existing Tracking Number entered.	Business Expense Application - View and Edit Form opened with preloaded details	Success	Fig 6.2
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated	Success	Fig 6.3
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details	Success	Fig 6.4
Clicking on headers of the columns of the ListView	Sorts the relevant columns	Success	Vid A1

Business Expense Application - New.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Form loaded	N/A	Tracking Number automatically assigned	Success	Fig 7.1
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 7.2
Save Button is clicked with valid data in the controls	Normal	Data is saved to database, email sent and ListView updated.	Success	Fig 7.3
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database, email sent and ListView updated.	Success	Fig 7.4
Save Button is clicked with controls left blank	Invalid	Output error message to user	Success	Fig 7.5
Save Button is clicked with valid data but no internet connection.	N/A	Output message telling user record was saved but email was not sent	Success	Fig 7.6
The text in the Quantity or Unit Price textbox is changed	N/A	The Total Price is automatically calculated	Success	Fig 7.7

Business Expense Application - View and Edit.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 8.1
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated	Success	Fig 8.2
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated	Success	Fig 8.3
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated	Success	Fig 8.4
Update Button with no data in controls	Invalid	Output error message to user	Success	Fig 8.5
Update Button with Approved checkbox checked from an initial unchecked state	Normal	The record is updated in the database, email sent to the applicant and ListView updated.	Success	Fig 8.6

Contact.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Send Button with relevant data in textboxes and richtextbox. Checkbox checked.	Normal	Sends email to the support email address	Success	Fig 9.1
Send Button with relevant data in textboxes and richtextbox. Checkbox unchecked.	Normal	Outputs error message	Success	Fig 9.2
Send button with relevant data in controls	Extreme	Sends email	Success	Fig 9.3
Send Button with email address in incorrect format	Invalid	Outputs error message	Success	Fig 9.4
Send Button with no data in controls	Invalid	Outputs error message	Success	Fig 9.5

Investment.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Add Button with data in controls	Normal	Adds record to database. ListView updated	Success	Fig 10.1
Add Button with data in controls	Extreme	Adds record to database. ListView updated	Success	Fig 10.2
Add Button with no data in controls	Invalid	Outputs error message	Success	Fig 10.3
Delete Button with item in ListView selected	N/A	Deletes record from database. ListView updated	Success	Fig 10.4
View and Edit Button with item in ListView selected	N/A	Opens the Investment – Edit form with relevant data	Success	Fig 10.5
Clicking on the header of the columns of the ListView	N/A	Sorts the relevant columns	Success	Fig 10.6 & Vid A2

Investment - Edit.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button	N/A	Closes the form	Success	Fig 11.1
Update Button with data in controls	Normal	Updates the record in the database. ListView on overview form updated	Success	Fig 11.2
Update Button with data in controls	Extreme	Updates the record in the database. Listview on overview form updated	Success	Fig 11.3
Update Button with no data in controls	Invalid	Output error message to user	Success	Fig 11.4

Login.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Login Button with incorrect username and password	Invalid	Outputs error message. Increases attempts by 1	Success	Fig 12.1
Login Button with correct username and password	Normal	Outputs success message and closes Login.vb and opens the Main Menu	Success	Fig 12.2
Login Button with incorrect username and password when at attempt 5 out of 5	Invalid	Outputs error message and exits the program	Success	Fig 12.3
All the controls on the ToolStrip (6)	N/A	Opens the relevant form or outputs a relevant message	Success	Vid 2.1

MainMenu.vb

Test	Expected Outcome	Result	Evidence
All the controls on the ToolStrip (17)	Opens the relevant forms	Success	Vid 3.1
Travel Application Button Clicked	Opens Travel Application - Overview form	Success	Fig 13.1
Business Application Button Clicked	Opens Business Expense Application - Overview form	Success	Fig 13.2
Revenues Clicked	Opens Revenues - Overview form	Success	Fig 13.3
Travel Expenses Button Clicked	Opens Travel Expense - Overview form	Success	Fig 13.4
Business Expenses Button Clicked	Opens Business Expense - Overview form	Success	Fig 13.5
Contact Support Button Clicked	Opens Contact Support form	Success	Fig 13.6
Investment Button Clicked	Opens Investment form	Success	Fig 13.7
Wages Button Clicked	Opens Wages - Overview form	Success	Fig 13.8
Account Manager Button Clicked	Opens Account Manager form	Success	Fig 13.9
Settings Button Clicked	Opens Settings form	Success	Fig 13.10
Help Button clicked	Shows message with instructions	Success	Fig 13.11
Logout Button clicked	Closes all open forms and then opens Login form	Success	Fig 13.12
Exit Button clicked and 'Yes' selected	Exits the application	Success	Fig 13.13

Revenues - Overview.vb

Test	Expected Outcome	Result	Evidence
New Revenue Button clicked	Revenues - New Form opened	Success	Fig 14.1
Search Button clicked and existing PO Number entered	Revenues - View and Edit Form opened with preloaded details	Success	Fig 14.2
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated	Success	Fig 14.3
Item selected on ListView and View or Edit Button clicked	Revenues - View and Edit Form opened with loaded details	Success	Fig 14.4
Dates selected and Filter Button clicked	Correct data filtered and displayed on the ListView	Success	Fig 14.5
Clear Button clicked	The filter is cleared and ListView refreshed with all the data in the Revenues table.	Success	Fig 14.6
Clicking on headers of the columns of the ListView	Sorts the relevant columns	Success	Vid A3

Revenues - New.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 15.1
Add Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated	Success	Fig 15.2
Add Button is clicked with extreme data in the controls	Extreme	Data is saved to database and the ListView is updated	Success	Fig 15.3
Add Button is clicked with controls left blank	Invalid	Output error message to user	Success	Fig 15.4
Add Button is clicked but a record with the PO Number already exists	Invalid	Output error message to user	Success	Fig 15.5
The text in the Subtotal or GST textbox is changed	N/A	The Total is automatically calculated	Success	Fig 15.6

Revenues - View and Edit.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Closes the form	Success	Fig 16.1
Delete Button is clicked and user selects 'Yes'	N/A	Deletes the record from the database and updates the ListView	Success	Fig 16.2
Update Button with data in controls	Normal	Updates the record in the database. ListView on overview form updated	Success	Fig 16.3
Update Button with data in controls	Extreme	Updates the record in the database. ListView on overview form updated	Success	Fig 16.4
Update Button with no data in controls	Invalid	Output error message to user	Success	Fig 16.5
The text in the Subtotal or GST textbox is changed	N/A	The Total is automatically calculated	Success	Fig 15.6

SearchBox.vb

Test	Expected Outcome	Result	Evidence
Search Button clicked with valid query in textbox	Opens relevant form with data loaded	Success	Fig 17.1
Cancel Button clicked	Form closes	Success	Fig 17.2

Settings.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Browse Button clicked and new valid file selected	Valid	Updates the setting for the database location	Success	Fig 18.1
Backup Button clicked and save selected	Valid	A copy of the database is copied into the specified directory	Success	Fig 18.2
Update Button clicked with valid email in textbox	Valid	Settings for application email address updated	Success	Fig 18.3
Update Button clicked with extreme email in textbox	Extreme	Settings for application email address updated	Success	Fig 18.4
Update Button clicked with invalid email in textbox	Invalid	Output error message	Success	Fig 18.5
View Variables Button clicked	N/A	Opens Variables Settings form	Success	Fig 18.6
Help Button clicked	N/A	Message box with information shown	Success	Fig 18.7
Close Button clicked	N/A	The form is closed	Success	Fig 18.8

SettingsVariables.vb

Test	Expected Outcome	Result	Evidence
Enter Button clicked with correct password in textbox	Form enabled and settings variable values loaded	Success	Fig 19.1
Enter Button clicked with incorrect password in textbox	Output error to user	Success	Fig 19.2
Close Button clicked	The form is closed	Success	Fig 19.3

StatusBox.vb

Test	Expected Outcome	Result	Evidence
OK Button is clicked	Form closes	Success	Fig 20.1

Travel Application - Overview.vb

Test	Expected Outcome	Result	Evidence
New Application Button clicked	Business Expense Application - New Form opened	Success	Fig 21.1
Search Button clicked and existing Tracking Number entered.	Business Expense Application - View and Edit Form opened with preloaded details	Success	Fig 21.2
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated	Success	Fig 21.3
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details	Success	Fig 21.4
Clicking on headers of the columns of the ListView	Sorts the relevant columns	Success	Vid A4

Travel Application - New.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Form loaded	N/A	Tracking Number automatically assigned	Success	Fig 22.1
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 22.2
Save Button is clicked with valid data in the controls	Normal	Data is saved to database, email sent and ListView updated.	Success	Fig 22.3
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database, email sent and ListView updated.	Success	Fig 22.4
Save Button is clicked with controls left blank	Invalid	Output error message to user	Success	Fig 22.5
Save Button is clicked with valid data but no internet connection.	N/A	Output message telling user record was saved but email was not sent	Success	Fig 22.6
The text in the Domestic, International or Accommodation costs textbox is changed	N/A	The Total costs is automatically calculated	Success	Fig 22.7

Travel Application - View and Edit.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 23.1
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated	Success	Fig 23.2
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated	Success	Fig 23.3
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated	Success	Fig 23.4
Update Button with no data in controls	Invalid	Output error message to user	Success	Fig 23.5
Update Button with Approved checkbox checked from an initial unchecked state	Normal	The record is updated in the database, email sent to the applicant and ListView updated.	Success	Fig 23.6
The text in the Domestic, International or Accommodation costs textbox is changed	N/A	The Total costs is automatically calculated	Success	Fig 22.7

Travel Expense - Overview.vb

Test	Expected Outcome	Result	Evidence
New Expense Button clicked	Business Expense Application - New Form opened	Success	Fig 24.1
Search Button clicked and existing Tracking Number entered.	Business Expense Application - View and Edit Form opened with preloaded details	Success	Fig 24.2
Item selected on ListView and Print Button clicked	Brings up print dialog and then print preview dialog with all data on the relevant tracking number	Success	Fig 24.3
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated	Success	Fig 24.4
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details	Success	Fig 24.5

Travel Expense.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Tracking Number that exists in the TEA table is entered into textbox	Normal	The relevant data from the TEA record is loaded into the controls. ListView is updated	Success	Fig 25.1
Tracking Number that does not exist in the TEA table is entered into textbox	Invalid	The controls are set to blank	Success	Fig 25.2
Save Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated	Success	Fig 25.3
Save Button is clicked with extreme data in the controls	Extreme	Data is saved to database and the ListView is updated	Success	Fig 25.4
Save Button is clicked with controls left blank	Invalid	Output error message to user	Success	Fig 25.5
The Clear Button is clicked	N/A	Any text in controls is cleared	Success	Fig 25.6
New Expense Button clicked	N/A	The form is reset to accept a new record entry	Success	Fig 25.7
Item is selected in ListView and Delete Button (bottom) clicked. User selects 'Yes'	N/A	Record is deleted from database and ListView updated	Success	Fig 25.8
Item is selected in ListView and View or Edit Button clicked.	N/A	The relevant details for the expense is loaded onto the form	Success	Fig 25.9
Record loaded to View and Edit and Delete Button clicked	N/A	Record deleted from database and ListView updated	Success	Fig 25.10
Update Button clicked with normal data in controls	Normal	Record and ListView updated	Success	Fig 25.11

Test	Type of Data	Expected Outcome	Result	Evidence
Update Button clicked with extreme data in controls	Extreme	Record and Listview updated	Success	Fig 25.12
Update Button clicked with no data in controls	Invalid	Error message output	Success	Fig 25.13
All Reimbursed Button clicked and 'Yes' selected	N/A	All Records have their Reimbursed field set to true	Success	Fig 25.14
Print Button clicked	N/A	All records for the tracking expense are collected into a report	Success	Fig 25.15
Clicking on headers of the columns of the ListView	N/A	Sorts the relevant columns	Success	Vid A5

Wages - Overview.vb

Test	Expected Outcome	Result	Evidence
New Record Button clicked	Wages - New Form opened	Success	Fig 26.1
Item selected on ListView and Delete Button clicked	Record deleted from database and ListView updated	Success	Fig 26.2
Item selected on ListView and View or Edit Button clicked	Business Expense Application – View and Edit Form opened with loaded details	Success	Fig 26.3
Filter Button clicked with filter options selected	Data is filtered and the ListView is updated accordingly	Success	Fig 26.4 & Vid 4.1
Clear Filter Button is clicked	The ListView is reset and all data is viewed	Success	Fig 26.5
Print All Data radio button is selected and Print Button is clicked	Print dialog and print preview dialog with all data from the Wages table is produced into a report	Success	Fig 26.6
Print Filtered Data radio button is selected and Print Button is clicked	Print dialog and print preview dialog with all data from the ListView produced into a report	Success	Fig 26.7
Clicking on headers of the columns of the ListView	Sorts the relevant columns	Success	Vid A6

Wages - New.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 27.1
Save Button is clicked with valid data in the controls	Normal	Data is saved to database and the ListView is updated	Success	Fig 27.2
Save Button is clicked with borderline data in the controls	Extreme	Data is saved to database and the ListView is updated	Success	Fig 27.3
Save Button is clicked with no data in controls	Invalid	Output error message	Success	Fig 27.4
The text in the Gross Pay, PAYE or KiwiSaver text box is changed	N/A	The Net Pay is automatically calculated	Success	Fig 27.5

Wages - View and Edit.vb

Test	Type of Data	Expected Outcome	Result	Evidence
Close Button is clicked and user selects 'Yes'	N/A	Form is closed	Success	Fig 28.1
Delete Button is clicked and user selects 'Yes'	N/A	The record is deleted from the database and the ListView is updated	Success	Fig 28.2
Update Button with valid data in controls	Normal	The record is updated in the database and the ListView is updated	Success	Fig 28.3
Update Button with extreme data in controls	Extreme	The record is updated in the database and the ListView is updated	Success	Fig 28.4
Update Button with no data in controls	Invalid	Output error message to user	Success	Fig 28.5
The text in the Gross Pay, PAYE or KiwiSaver text box is changed	N/A	The Net Pay is automatically calculated	Success	Fig 28.6

Testing Evidence

Irrelevant parts of the forms to testing have been cropped out for ease of viewing.

Fig 1.1

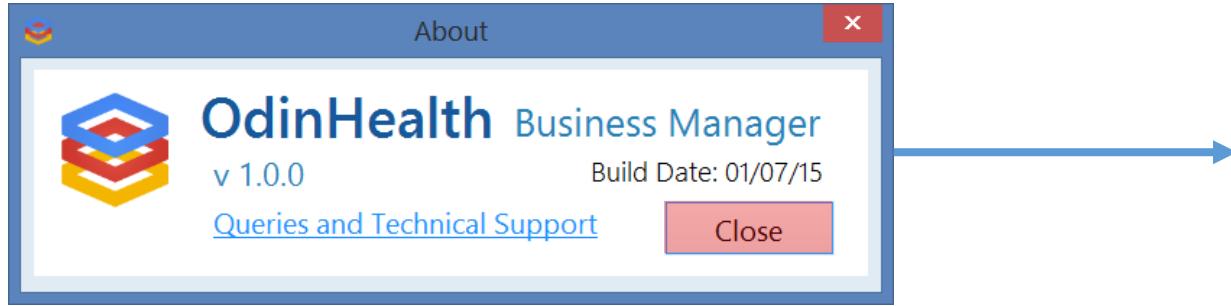
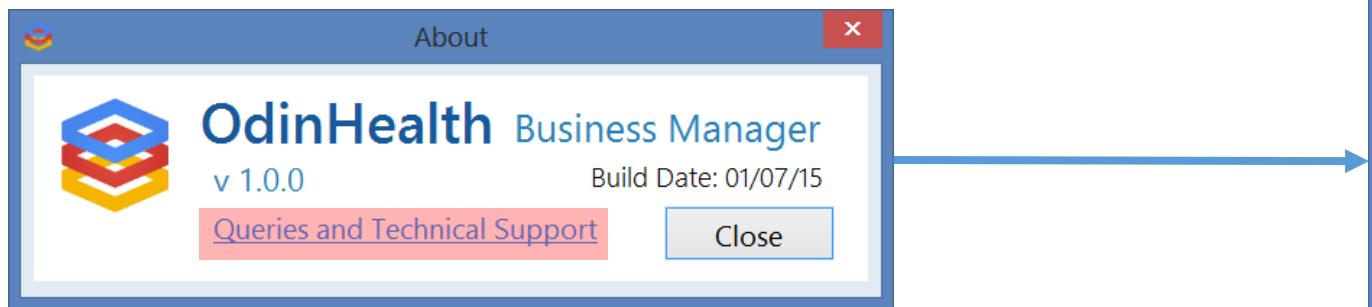


Fig 1.2



A screenshot of a "Contact Support" dialog box. The title bar has a blue header with the word "Contact Support" and a red "x" button. The main content area includes fields for "Name" (with a blue input box), "Email" (with a blue input box), and "Message" (with a large blue text area). Below these fields is a checkbox labeled "This Email is not spam." and a character limit indicator "0/1000". At the bottom, there is a note "The email may take up to 10 seconds to send. Thank you for your patience." and a blue "Send" button.

Fig 2.1

Account Manager

Please enter the Admin password to enable the Account Manager:

***** Enter

Add New Account

Username: Password: Full Name:
 Admin?

Delete Account

Username: Confirm

View Existing Accounts

Username	Password	Full Name	Admin
TIP! Resize the columns of the 'View' box to see more data.			

Lock Controls Close

Account Manager

Please enter the Admin password to enable the Account Manager:

***** Enter Correct password.

Add New Account

Username: Password: Full Name:
 Admin?

Delete Account

Username: Confirm

View Existing Accounts

Username	Password	Full Name	Admin
TIP! Resize the columns of the 'View' box to see more data.			

Lock Controls Close

Fig 2.2

Account Manager

Please enter the Admin password to enable the Account Manager:

***** Enter

Add New Account

Username: Password: Full Name:
 Admin?

Delete Account

Username: Confirm

View Existing Accounts

Username	Password	Full Name	Admin
TIP! Resize the columns of the 'View' box to see more data.			

Lock Controls Close

Account Manager

Please enter the Admin password to enable the Account Manager:

***** Enter Incorrect password.

Add New Account

Username: Password: Full Name:
 Admin?

Delete Account

Username: Confirm

View Existing Accounts

Username	Password	Full Name	Admin
TIP! Resize the columns of the 'View' box to see more data.			

Lock Controls Close

Fig 2.3

Add New Account

Username

Password

Full Name

Admin?

Add New Account

Username

Password

Full Name

Admin?

Account successfully added.

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	brucelee123	Bruce Lee	True
daniel.dyer	penguins	Daniel Dyer	False
william.shen	password	William Shen	True

Fig 2.4

Add New Account

Username

Password

Full Name

Admin?

Add New Account

Username

Password

Full Name

Admin?

Account successfully added.

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	brucelee123	Bruce Lee	True
daniel.dyer	penguins	Daniel Dyer	False
h.wolfeschl...	we85b7nv29034...	Hubert Bla...	False
william.shen	password	William Sh...	True

Fig 2.5

The diagram shows two identical 'Add New Account' forms side-by-side, connected by a blue arrow pointing from the first to the second. Both forms contain fields for Username, Password, and Full Name, along with checkboxes for 'Admin?' and a red 'Add' button.

Add New Account

Username
Password
Full Name
 Admin?

Add New Account

Username
Password
Full Name
 Admin?
Please confirm the details.

Fig 2.6

The diagram shows three screens connected by blue arrows. The first screen is 'Delete Account' with a dropdown menu showing 'h.wolfeschlegelsteinhausenbe...'. The second screen is also 'Delete Account' with a dropdown menu showing 'daniel.dyer', a 'Confirm' checkbox (unchecked), and a blue 'Delete' button. A green message 'Account has been deleted.' is displayed below the button. The third screen is 'View Existing Accounts' showing a table of user data.

Delete Account

Username
 Confirm

Delete Account

Username
 Confirm
Account has been deleted.

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	brucelee123	Bruce Lee	True
daniel.dyer	penguins	Daniel Dyer	False
william.shen	password	William Shen	True

Fig 2.7

The diagram shows two 'Delete Account' screens connected by a blue arrow. Both screens have a dropdown menu set to 'bruce.lee' and a red 'Delete' button. The second screen also features a 'Confirm' checkbox (unchecked) and a red message 'Please confirm your choice.' below the button.

Delete Account

Username
 Confirm

Delete Account

Username
 Confirm
Please confirm your choice.

Fig 2.8

View Existing Accounts

Username	Password	Full Name	Admin

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	brucelee123	Bruce Lee	True
daniel.dyer	penguins	Daniel Dyer	False
william.shen	password	William Shen	True

Fig 2.9

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	brucelee123	Bruce Lee	True
daniel.dyer	penguins	Daniel Dyer	False
william.shen	password	William Shen	True

View Existing Accounts

Username	Password	Full Name	Admin

Fig 2.10

Update Account

Username	<input type="text" value="bruce.lee"/>
Password	<input type="text" value="notsecurepassword"/>
Full Name	<input type="text" value="Brucey Lee"/>
<input type="checkbox"/> Admin?	<input type="button" value="Update"/>

Update Account

Username	<input type="text" value="bruce.lee"/>
Password	<input type="text"/>
Full Name	<input type="text"/>
<input type="checkbox"/> Admin?	<input type="button" value="Update"/>

Account has been updated.

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	notsecurepassword	Brucey Lee	False
daniel.dyer	penguins	Daniel Dyer	False
william.shen	password	William Shen	True

Fig 2.11

Update Account

Username: bruce.lee
Password: 3456789123456789123456789
Full Name: BruceBruceBruceBruce Lee
 Admin? Update

Update Account

Username: bruce.lee
Password:
Full Name:
 Admin? Update

Account has been updated.

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	123456789123...	BruceBruceB...	True
daniel.dyer	penguins	Daniel Dyer	False
william.shen	password	William Shen	True

Fig 2.12

Update Account

Username: bruce.lee
Password:
Full Name:
 Admin? Update

Update Account

Username: bruce.lee
Password:
Full Name:
 Admin? Update

Please confirm the details.

Fig 2.13

Account Manager

Please enter the Admin password to enable the Account Manager:

Enter Correct password

Add New Account
Username:
Password:
Full Name:
 Admin? Add

Delete Account
Username: bruce.lee
 Confirm Delete

View Existing Accounts

Username	Password	Full Name	Admin
bruce.lee	123456789123...	BruceBruceB...	True
daniel.dyer	penguins	Daniel Dyer	False
william.shen	password	William Shen	True

TIP! Resize the columns of the 'View' box to see more data.

Clear View Lock Controls Close

Fig 2.14

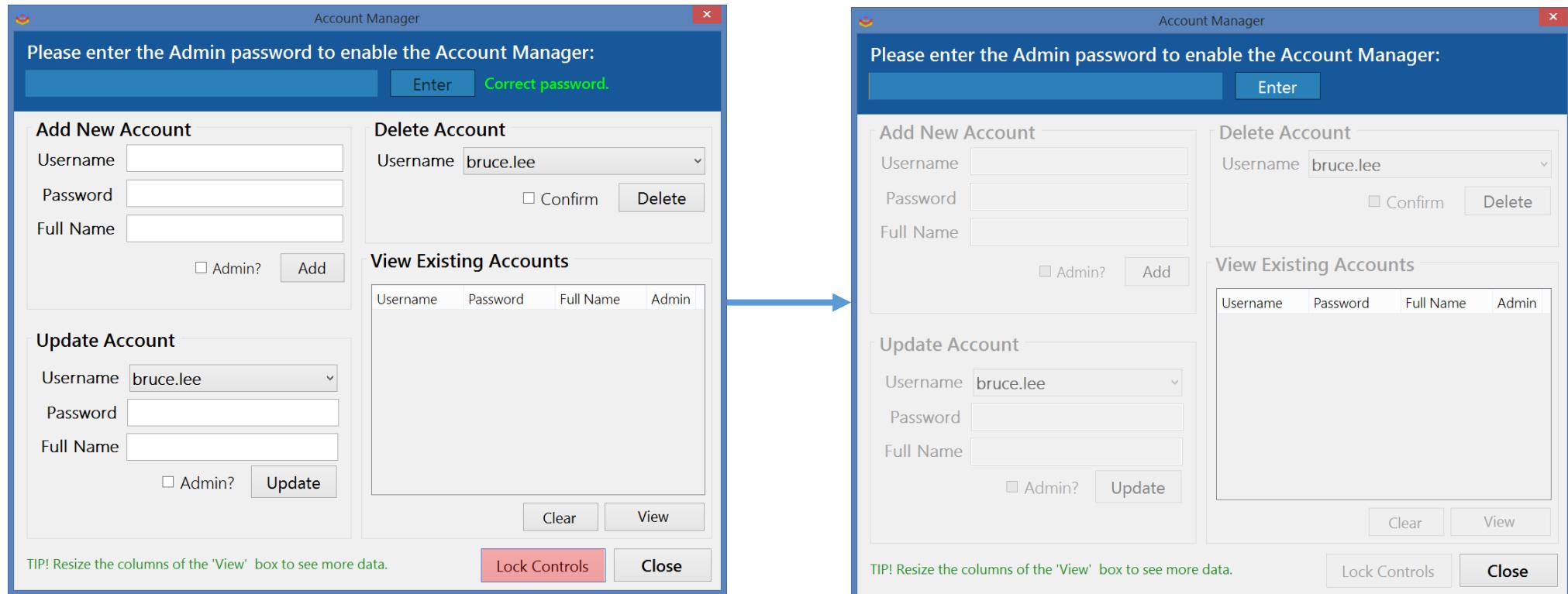


Fig 3.1

Business Expense - Overview

Pending Reimbursement				
Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS0305150	Pedro Quintus	16/07/2015	Monitors	2500
BS11061501	Richard Hendrix	15/07/2015	Microwave	350
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Completed Reimbursement				
Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS1106150	William Shen	27/06/2015	Ubiquiti Unifi AP	53136
BS2105150	William Shen	6/07/2015	LED Lamps	460

Select a row from the tables above:

View or Edit **Delete** **Search** **New Expense**



Business Expense - New

Tracking Number Employee Name

Date Purchased Vendor Payment Method

16/07/2015 PB Tech Company

Item Description Quantity

Unit Price Total Price NZD NZD

Reimbursement required?

Enter the Tracking Number and all relevant data will 'appear.'

Close **Save**

Fig 3.2

Business Expense - Overview

Pending Reimbursement				
Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS0305150	Pedro Quintus	16/07/2015	Monitors	2500
BS11061501	Richard Hendrix	15/07/2015	Microwave	350
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Completed Reimbursement				
Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS1106150	William Shen	27/06/2015	Ubiquiti Unifi AP	53136
BS2105150	William Shen	6/07/2015	LED Lamps	460

Select a row from the tables above:

View or Edit **Delete** **Search** **New Expense**




Please enter the Tracking Number.

BS1106150

Cancel **Search**

Business Expense - View and Edit

Reimbursed?

Tracking Number Employee Name

BS1106150 William Shen

Date Purchased Vendor Payment Method

27/06/2015 PB Tech Company

Item Description Quantity

Ubiquiti Unifi AP 123

Unit Price Total Price NZD NZD

Reimbursement required?

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Fig 3.3

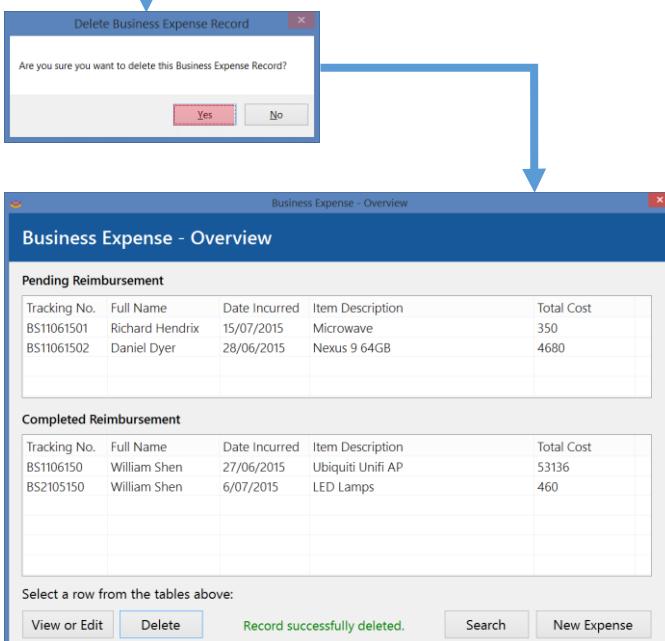
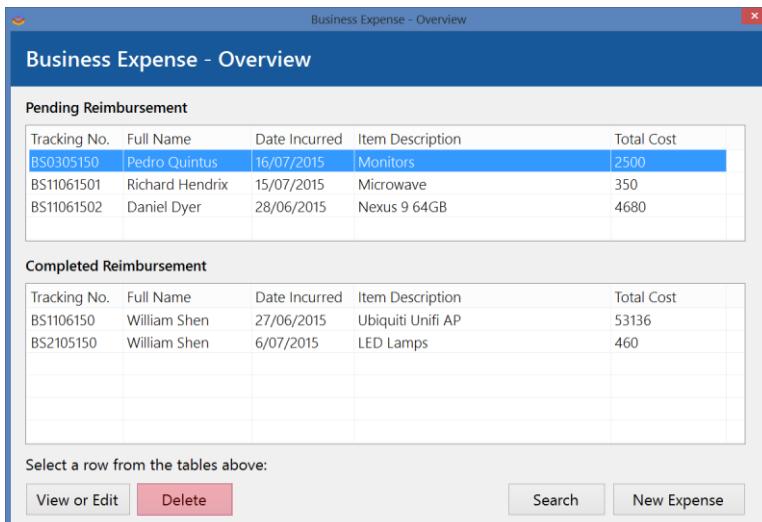


Fig 3.4

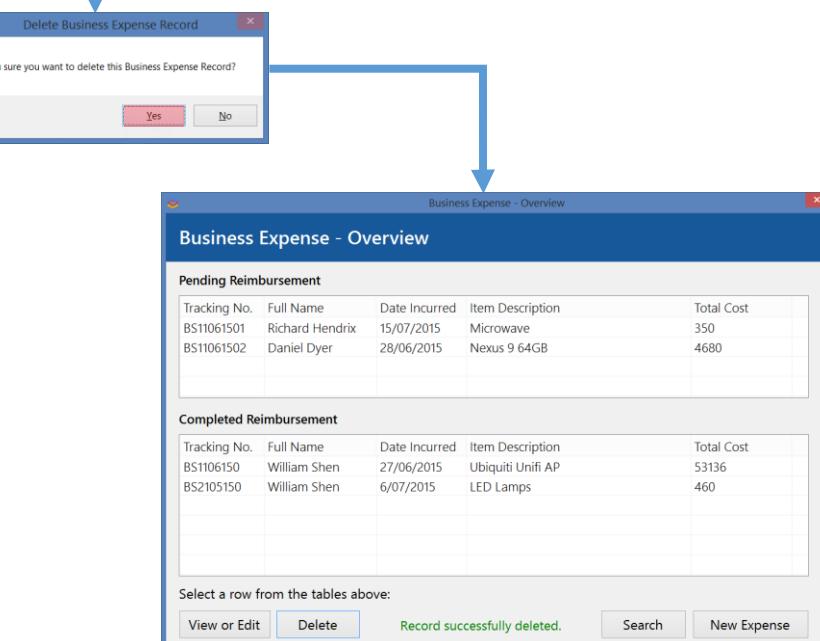
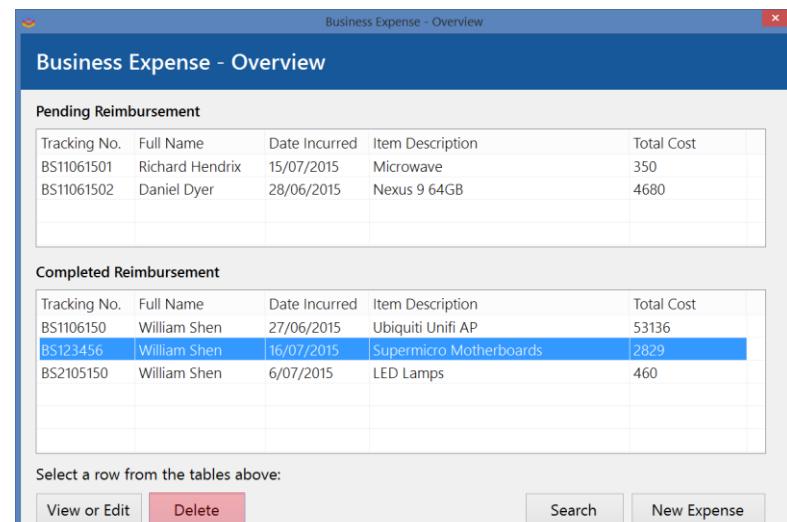


Fig 3.5

Business Expense - Overview

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS11061501	Richard Hendrix	15/07/2015	Microwave	350
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS1106150	William Shen	27/06/2015	Ubiquiti Unifi AP	53136
BS2105150	William Shen	6/07/2015	LED Lamps	460

Select a row from the tables above:

View or Edit **Delete** **Search** **New Expense**

Business Expense - View and Edit

Reimbursed?

Tracking Number	Employee Name	
BS11061502	Daniel Dyer	
Date Purchased	Vendor	Payment Method
28/06/2015	Google	Personal
Item Description	Quantity	
Nexus 9 64GB	6	
Unit Price	Total Price	
780 NZD	4680 NZD	
<input checked="" type="checkbox"/> Reimbursement required?		

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Fig 3.6

Business Expense - Overview

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS11061501	Richard Hendrix	15/07/2015	Microwave	350
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
BS1106150	William Shen	27/06/2015	Ubiquiti Unifi AP	53136
BS2105150	William Shen	6/07/2015	LED Lamps	460

Select a row from the tables above:

View or Edit **Delete** **Search** **New Expense**

Business Expense - View and Edit

Reimbursed?

Tracking Number	Employee Name	
BS1106150	William Shen	
Date Purchased	Vendor	Payment Method
27/06/2015	PB Tech	Company
Item Description	Quantity	
Ubiquiti Unifi AP	123	
Unit Price	Total Price	
432 NZD	53136 NZD	
<input type="checkbox"/> Reimbursement required?		

Update the relevant data and press 'Update'.

Close **Delete** **Update**

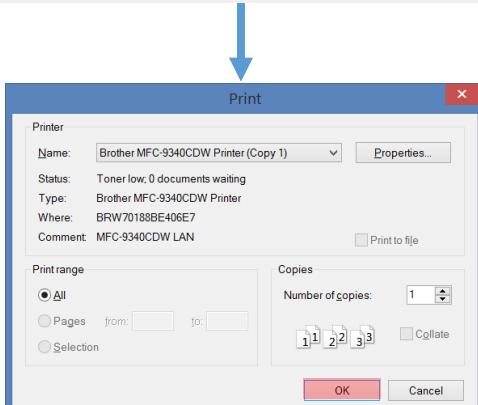
Fig 3.7

Print by:

Expenses Between and Completed Reimbursement

Employee Name Pending Reimbursement

Print all data Find Total



Business Expenses

Tracking No.	Name	Date	Description & Qty	Vendor	Total	Reimbursed
BS1106150	William Shen	27/06/15	Ubiquiti Unifi AP x123	PB Tech	\$53136	True
BS11061502	Daniel Dyer	28/06/15	Nexus 9 64GB x6	Google	\$4680	False
BS2105150	William Shen	06/07/15	LED Lamps x20	PB Tech	\$460	True
BS11061501	Richard Hendrix	15/07/15	Microwave x1	Panasonic	\$350	False
			TOTAL =		\$58626	

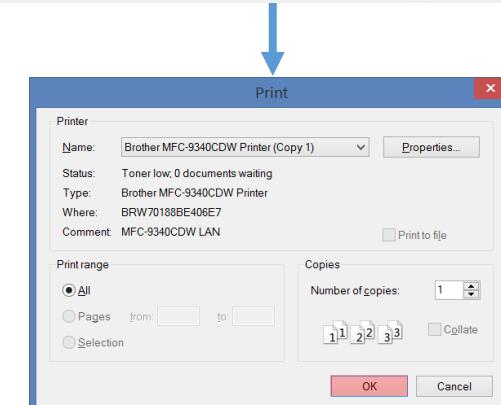
Fig 3.8

Print by:

Expenses Between and Completed Reimbursement

Employee Name Pending Reimbursement

Print all data Find Total



Business Expenses

Tracking No.	Name	Date	Description & Qty	Vendor	Total	Reimbursed
BS1106150	William Shen	27/06/15	Ubiquiti Unifi AP x123	PB Tech	\$53136	True
BS11061502	Daniel Dyer	28/06/15	Nexus 9 64GB x6	Google	\$4680	False
BS2105150	William Shen	06/07/15	LED Lamps x20	PB Tech	\$460	True
BS11061501	Richard Hendrix	15/07/15	Microwave x1	Panasonic	\$350	False

Fig 4.1

Business Expense - New

Tracking Number	Employee Name	
BS210515		
Date Purchased	Vendor	Payment Method
16/07/2015		
Item Description	Quantity	
Unit Price	Total Price	
	NZD	NZD
<input type="checkbox"/> Reimbursement required?		
Enter the Tracking Number and all relevant data will 'appear'.		
<input type="button" value="Close"/> <input type="button" value="Save"/>		

Fig 4.2

Business Expense - New

Tracking Number	Employee Name	
123456789		
Date Purchased	Vendor	Payment Method
16/07/2015		
Item Description	Quantity	
Unit Price	Total Price	
	NZD	NZD
<input type="checkbox"/> Reimbursement required?		
Enter the Tracking Number and all relevant data will 'appear'.		
<input type="button" value="Close"/> <input type="button" value="Save"/>		

Business Expense - New

Tracking Number	Employee Name	
BS2105150	William Shen	
Date Purchased	Vendor	Payment Method
16/07/2015	PB Tech	
Item Description	Quantity	
LED Lamps	20	
Unit Price	Total Price	
23 NZD	460 NZD	
<input type="checkbox"/> Reimbursement required?		
Enter the Tracking Number and all relevant data will 'appear'.		
<input type="button" value="Close"/> <input type="button" value="Save"/>		

Data referenced from Business Expense Applications

Business Expense Application - View and Edit

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS2105150
Item Description	Quantity	
LED Lamps	20	
Unit Price	Total Price	Vendor
23 NZD	460 NZD	PB Tech
Reason for Purchase		
We need to create a bright environment.		
Update the relevant data and press 'Update'.		
<input type="button" value="Close"/> <input type="button" value="Delete"/> <input type="button" value="Update"/>		

Fig 4.3

Business Expense - New

Tracking Number	Employee Name	
<input type="text"/>	<input type="text"/>	
Date Purchased	Vendor	Payment Method
16/07/2015	<input type="text"/>	<input type="button"/>
Item Description		Quantity
<input type="text"/>		<input type="text"/>
Unit Price	Total Price	
<input type="text"/> NZD	<input type="text"/> NZD	<input type="checkbox"/> Reimbursement required?
Enter the Tracking Number and all relevant data will 'appear'.		
<input type="button"/> Close		<input type="button"/> Save

Fig 4.4

Business Expense - New

Tracking Number	Employee Name	
12345	<input type="text"/>	
Date Purchased	Vendor	Payment Method
16/07/2015	<input type="text"/> Noel Leeming	<input type="button"/>
Item Description		Quantity
<input type="text"/> Samsung TV		<input type="text"/> 2
Unit Price	Total Price	
<input type="text"/> 1500 NZD	<input type="text"/> 3000 NZD	<input type="checkbox"/> Reimbursement required?
Enter the Tracking Number and all relevant data will 'appear'.		
<input type="button"/> Close		<input type="button"/> Save

Fig 4.5

Business Expense - New

Tracking Number	Employee Name	
0123456789	William William William William William William William	
Date Purchased	Vendor	Payment Method
16/07/2015	Z NZ	<input type="button"/>
Item Description		Quantity
Servers Servers Servers Servers Servers Servers Serv		<input type="text"/> 99999
Unit Price	Total Price	
<input type="text"/> 9999999999 NZD	<input type="text"/> 99998999990000 NZD	<input type="checkbox"/> Reimbursement required?
Enter the Tracking Number and all relevant data will 'appear'.		
<input type="button"/> Close		<input type="button"/> Save

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
12345	William Shen	16/07/2015	Samsung TV	3000
BS11061501	William Shen	27/06/2015	Ubiquiti UniFi AP	53136
BS2105150	William Shen	6/07/2015	LED Lamps	460

Select a row from the tables above:

View or Edit Delete Record successfully added. Search New Expense

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
0123456789	William William ...	16/07/2015	Servers Servers Servers Servers Serv...	99998999990000
BS11061501	Richard Hendrix	15/07/2015	Microwave	350
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Fig 4.6

Business Expense - New

Tracking Number Employee Name
[] []

Date Purchased Vendor Payment Method
16/07/2015 [] Company

Item Description Quantity
[] []

Unit Price Total Price
[] NZD [] NZD Reimbursement required?

Enter the Tracking Number and all relevant data will 'appear'. Close Save

Business Expense - New

Tracking Number Employee Name
[] []

Date Purchased Vendor Payment Method
16/07/2015 [] Company

Item Description Quantity
[] []

Unit Price Total Price
[] NZD [] NZD Reimbursement required?

Please ensure all fields are filled out correctly. Close Save

Fig 4.7

Business Expense - New

Tracking Number Employee Name
12345 Bruce Lee

Date Purchased Vendor Payment Method
16/07/2015 [] Company

Item Description Quantity
Sushi [] 60

Unit Price Total Price
1 NZD [] 60 NZD Reimbursement required?

Enter the Tracking Number and all relevant data will 'appear'. Close Save

Business Expense - New

Tracking Number Employee Name
12345 Bruce Lee

Date Purchased Vendor Payment Method
16/07/2015 [] Company

Item Description Quantity
Sushi [] 60

Unit Price Total Price
1 NZD [] 60 NZD Reimbursement required?

A record for the tracking number already exists. Close Save

Fig 4.9

Item Description Quantity
Chicken Nuggets [] 100

Unit Price Total Price
[] NZD [] 0 NZD Reimbursement required?

Fig 4.8

Business Expense - New

Tracking Number Employee Name
[] []

Date Purchased Vendor Payment Method
16/07/2015 [] Company

Item Description Quantity
[] []

Unit Price Total Price
[] NZD [] NZD Reimbursement required?

Enter the Tracking Number and all relevant data will 'appear'. Close Save

Business Expense - New

Tracking Number Employee Name
[] []

Date Purchased Vendor Payment Method
16/07/2015 [] Company

Item Description Quantity
[] []

Unit Price Total Price
[] NZD [] NZD Reimbursement required?

Enter the Tracking Number and all relevant data will 'appear'. Close Save

Item Description Quantity
Chicken Nuggets [] 100

Unit Price Total Price
0.87 NZD [] 87 NZD Reimbursement required?

Fig 5.1

The screenshot shows the 'Business Expense - View and Edit' window. It contains fields for Tracking Number (BS1106150), Employee Name (William Shen), Date Purchased (27/06/2015), Vendor (PB Tech), Payment Method (Company), Item Description (Ubiquiti Unifi AP), Quantity (123), Unit Price (432 NZD), Total Price (53136 NZD), and a checked checkbox for 'Reimbursement required?'. Below the form is a message: 'Update the relevant data and press 'Update''. At the bottom are buttons for Close, Delete, and Update. A blue arrow points from the bottom right of the main window to a small 'Close' dialog box. This dialog box contains the question 'Are you sure you want to close this form?' with Yes and No buttons.

Fig 5.3

The screenshot shows the 'Business Expense - View and Edit' window. It contains fields for Tracking Number (BS2105150), Employee Name (William Shen), Date Purchased (6/07/2015), Vendor (PB Tech), Payment Method (Personal), Item Description (LED Lamps), Quantity (200), Unit Price (23 NZD), Total Price (4600 NZD), and a checked checkbox for 'Reimbursement required?'. Below the form is a message: 'Update the relevant data and press 'Update''. At the bottom are buttons for Close, Delete, and Update. A blue arrow points from the bottom right of the main window to a 'Completed Reimbursement' table. The table has columns: Tracking No., Full Name, Date Incurred, Item Description, and Total Cost. It lists two rows: one for BS2105150 (William Shen, 6/07/2015, LED Lamps, 4600) and one for 12345 (William Shen, 16/07/2015, Samsung TV, 3000). Below the table is a message: 'Record successfully updated.' and buttons for View or Edit, Delete, Search, and New Expense.

Fig 5.2

The screenshot shows the 'Business Expense - View and Edit' window, identical to Fig 5.1 but with different tracking numbers. It contains fields for Tracking Number (BS1106150), Employee Name (William Shen), Date Purchased (27/06/2015), Vendor (PB Tech), Payment Method (Company), Item Description (Ubiquiti Unifi AP), Quantity (123), Unit Price (432 NZD), Total Price (53136 NZD), and a checked checkbox for 'Reimbursement required?'. Below the form is a message: 'Update the relevant data and press 'Update''. At the bottom are buttons for Close, Delete, and Update. A blue arrow points from the bottom right of the main window to a 'Delete Application' dialog box. This dialog box contains the question 'Are you sure you want to delete this Business Expense Application?' with Yes and No buttons.

Fig 5.4

The diagram illustrates a data entry process. On the left, a window titled "Business Expense - View and Edit" shows a record being entered. The tracking number is 0123456789, the employee name is William, and the date purchased is 16/07/2015. The item description is Servers, and the quantity is 99999. The total price is 99998999990000 NZD. The "Reimbursed?" checkbox is unchecked. A blue arrow points from this window to a central grid titled "Pending Reimbursement". The grid lists three entries: BS11061501 (Richard Hendrix, 15/07/2015, Microwave, 350), BS11061502 (Daniel Dyer, 28/06/2015, Nexus 9 64GB, 4680), and the new entry for William.

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
0123456789	William William ...	16/07/2015	Servers Servers Servers Servers Serv...	99998999990000
BS11061501	Richard Hendrix	15/07/2015	Microwave	350
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Fig 5.5

This diagram shows a validation process. The initial window on the left has a tracking number of BS2105150, an employee name of William Shen, and a date purchased of 6/07/2015. The item description is empty, and the quantity is also empty. The total price is 0 NZD. The "Reimbursed?" checkbox is checked. A blue arrow points from this window to a second, identical-looking window on the right. In this second window, a red error message at the bottom states: "Please ensure all fields are filled out correctly." The tracking number, employee name, and date purchased remain the same, but the item description and quantity fields are now highlighted in red, indicating they are required but empty.

Fig 5.6

This diagram shows an update process. The initial window on the left has a tracking number of BS2105150, an employee name of William Shen, and a date purchased of 6/07/2015. The item description is Samsung TV, and the quantity is 2. The total price is 3000 NZD. The "Reimbursed?" checkbox is checked. A blue arrow points from this window to a second window on the right. In this second window, the item description is changed to Samsung TV, and the quantity is updated to 20. The total price is now 30000 NZD. The "Reimbursed?" checkbox remains checked.

Fig 6.1

Business Expense Application - Overview

Pending Applications (Unapproved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Past Applications (Approved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250
BS0807150	William Shen	Chicken Nuggets	69	69
BS1106151	William Shen	Dell Projector	1	1200

Select a row from the tables above:

View or Edit **Delete** **Search** **New Application**

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607150
Item Description		Quantity
Unit Price	Total Price	Vendor
1200 NZD	1200 NZD	Dell.co.nz
Reason for Purchase		
bruce lee sushi		

Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.

Close **Submit**

Fig 6.2

Business Expense Application - Overview

Pending Applications (Unapproved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Past Applications (Approved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250
BS0807150	William Shen	Chicken Nuggets	69	69
BS1106151	William Shen	Dell Projector	1	1200

Select a row from the tables above:

View or Edit **Delete** **Search** **New Application**

Please enter the Tracking Number.
BS1106151

Cancel **Search**

Business Expense Application - View and Edit

Approved?

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1106151
Item Description		Quantity
Dell Projector		1
Unit Price	Total Price	Vendor
1200 NZD	1200 NZD	Dell.co.nz
Reason for Purchase		
bruce lee sushi		

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Fig 6.3

The diagram illustrates a user interaction flow for managing business expense applications. It starts with a list of approved applications, where a specific row is selected. This leads to a confirmation dialog asking if the user wants to delete the application. If confirmed, the application is deleted, and the final step shows the updated list of approved applications.

Past Applications (Approved)				
Tracking No.	Full Name	Description	Quantity	Total Price
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250
BS0807150	William Shen	Chicken Nuggets	69	69
BS1106151	William Shen	Dell Projector	1	1200

Select a row from the tables above:

View or Edit **Delete** **Search** **New Application**

Delete Business Expense Application

Are you sure you want to delete this Business Expense Application?

Yes **No**

Past Applications (Approved)				
Tracking No.	Full Name	Description	Quantity	Total Price
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250
BS1106151	William Shen	Dell Projector	1	1200

Select a row from the tables above:

View or Edit **Delete** **Record successfully deleted.** **Search** **New Application**

Fig 6.4

The diagram illustrates a user interaction flow for managing business expense applications. It starts with a list of pending and approved applications. A specific pending application is selected, which then opens a detailed view and edit window. This window allows the user to update various fields such as employee name, item description, unit price, total price, vendor, and reason for purchase. The window also includes buttons for closing, deleting, or updating the record.

Pending Applications (Unapproved)				
Tracking No.	Full Name	Description	Quantity	Total Price
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Select a row from the tables above:

View or Edit **Delete** **Record successfully deleted.** **Search** **New Application**

Business Expense Application - View and Edit		
<input type="checkbox"/> Approved?		
Employee Name	Email Address	Tracking Number
Marco Yu	william.shen@odinhealth.co.nz	BS2305150
Item Description		
Dell U2515H Monitors		
Quantity	12	
Unit Price	Total Price	Vendor
546 NZD	6552 NZD	Dell
Reason for Purchase		
My current Surface Pro screen is way too small		

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Fig 7.1

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607150
Item Description		Quantity
Unit Price	Total Price	Vendor
180 NZD	900 NZD	Microsoft
Reason for Purchase		
For word processing and spreadsheets		
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.		
<input type="button" value="Close"/>		<input type="button" value="Submit"/>

Fig 7.2

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607150
Item Description		Quantity
Unit Price	Total Price	Vendor
180 NZD	900 NZD	Microsoft
Reason for Purchase		
For word processing and spreadsheets		
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.		
<input type="button" value="Close"/>		<input type="button" value="Submit"/>

Fig 7.3

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607150
Item Description		Quantity
Office 365 Software		5
Unit Price	Total Price	Vendor
180 NZD	900 NZD	Microsoft
Reason for Purchase		
For word processing and spreadsheets		
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.		
<input type="button" value="Close"/>		<input type="button" value="Submit"/>

Pending Applications (Unapproved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS1607150	William Shen	Office 365 Software	5	900
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Business Expense Application for William Shen

Date of Submission: 16/07/2015
Return Email: wiliam.shen@odinhealth.co.nz

Item Description: Office 365 Software (Quantity: 5)
Unit Price: \$180 NZD
Total Price: \$900 NZD
Vendor: Microsoft

Reason for Expense: For word processing and spreadsheets

To approve this application, open the Business Manager program and navigate to the Business Expense Applications, select the application click the 'View and Edit' button and then tick the approved box and 'Update'.

Fig 7.4

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607151
Item Description	Quantity	
OnePlus One	99999	
Unit Price	Total Price	Vendor
9999999999 NZD	9999899999900001 NZD	us One OnePlus One OnePlus One On
Reason for Purchase		
OnePlus One #android		
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.		
Close		Submit

Pending Applications (Unapproved)				
Tracking No.	Full Name	Description	Quantity	Total Price
BS1607150	William Shen	Office 365 Software	5	900
BS1607151	William Shen	OnePlus One OnePlus One OnePlus ...	99999	99999000000
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Business Expense Application for William Shen

Date of Submission: 16/07/2015
Return Email: wiliam.shen@odinhealth.co.nz

Item Description: OnePlus One
OnePlus One (Quantity: 99999)
Unit Price: \$100000 NZD
Total Price: \$999990000 NZD
Vendor: OnePlus One OnePlus One OnePlus One OnePlus One

To approve this application, open the Business Manager program and navigate to the Business Expense Applications, select the application click the 'View and Edit' button and then tick the approved box and 'Update'.

Fig 7.5

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607152
Item Description	Quantity	
<input type="text"/>		<input type="text"/>
Unit Price	Total Price	Vendor
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/>
Reason for Purchase	<input type="text"/>	

Business Expense Application - New

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607152
Item Description	Quantity	
Unit Price	Total Price	Vendor
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/>
Reason for Purchase	<input type="text"/>	

Please ensure all fields are filled out correctly.

Fig 7.6

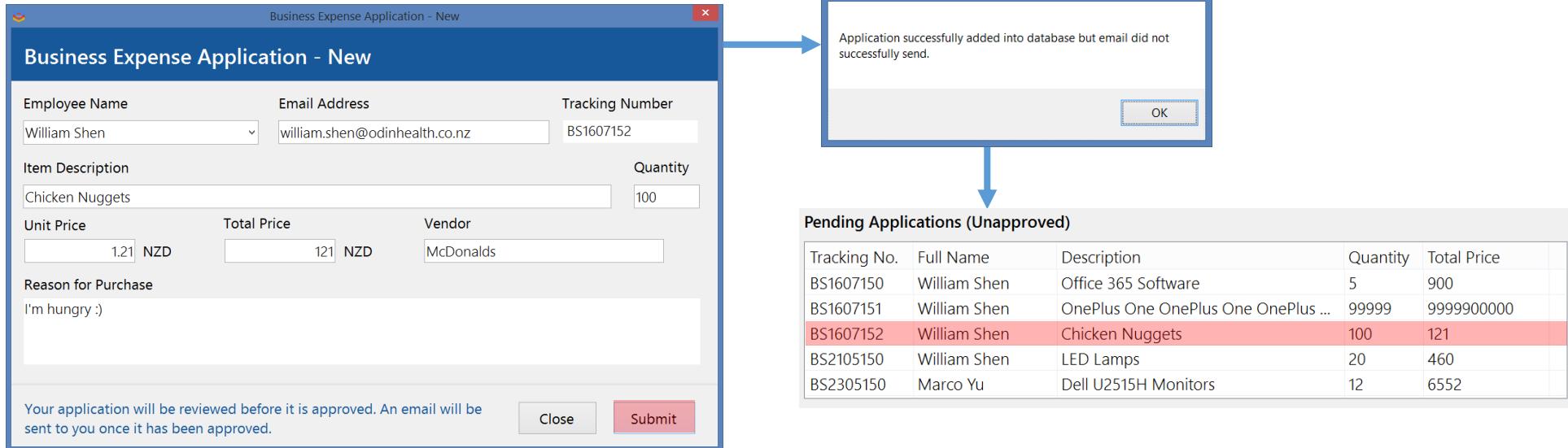


Fig 7.7

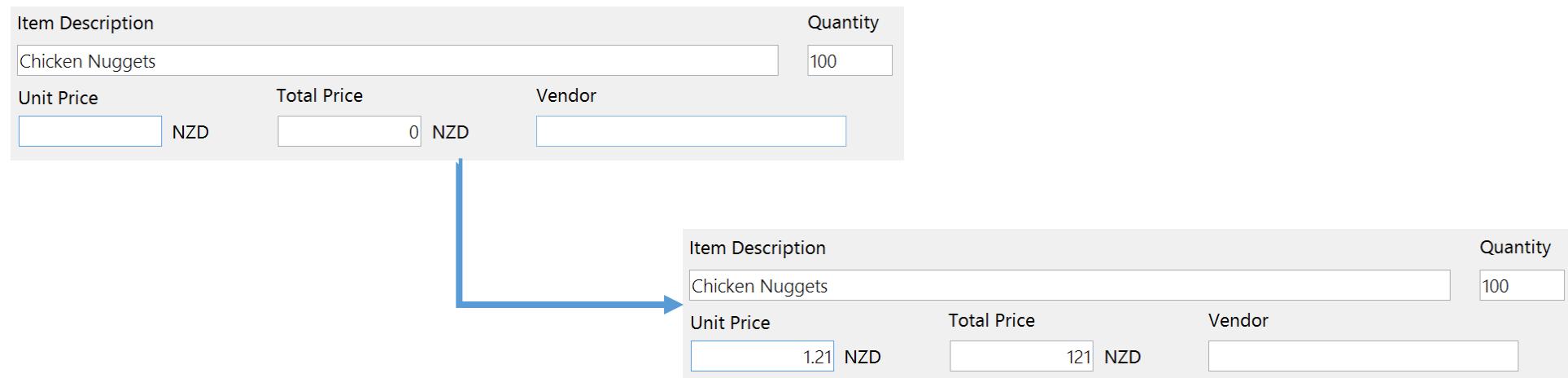


Fig 8.1

Business Expense Application - View and Edit

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS2105150
Item Description		Quantity
LED Lamps		20
Unit Price	Total Price	Vendor
23 NZD	460 NZD	PB Tech
Reason for Purchase		
We need to create a bright environment.		
Update the relevant data and press 'Update'.		
<input type="button" value="Close"/>	<input type="button" value="Delete"/>	<input type="button" value="Update"/>

Fig 8.3

Business Expense Application - View and Edit

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607152
Item Description		Quantity
Chicken Niblets		100
Unit Price	Total Price	Vendor
1.21 NZD	121 NZD	McDonalds
Reason for Purchase		
I'm hungry :)		
Update the relevant data and press 'Update'.		
<input type="button" value="Close"/>	<input type="button" value="Delete"/>	<input type="button" value="Update"/>

Fig 8.2

Business Expense Application - View and Edit

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607150
Item Description		Quantity
Office 365 Software		5
Unit Price	Total Price	Vendor
180 NZD	900 NZD	Microsoft
Reason for Purchase		
For word processing and spreadsheets		
Update the relevant data and press 'Update'.		
<input type="button" value="Close"/>	<input type="button" value="Delete"/>	<input type="button" value="Update"/>

Pending Applications (Unapproved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS1607151	William Shen	OnePlus One OnePlus One OnePlus ...	99999	9999900000
BS1607152	William Shen	Chicken Niblets	100	121
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Pending Applications (Unapproved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS1607151	William Shen	OnePlus One OnePlus One OnePlus ...	99999	9999900000
BS1607152	William Shen	Chicken Nuggets	100	121
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Fig 8.4

Pending Applications (Unapproved)				
Tracking No.	Full Name	Description	Quantity	Total Price
BS1607151	William Shen	OnePlus One OnePlus One OnePlus ...	99999	9999900000
BS1607152	William Shen	Chicken Niblets	100	121
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U2515H Monitors	12	6552

Fig 8.5

Business Expense Application - View and Edit

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607151
Item Description	Quantity	
Unit Price	Total Price	Vendor
100	0 NZD	
Reason for Purchase		
Update the relevant data and press 'Update'.		
<input type="button" value="Close"/> <input type="button" value="Delete"/>		<input type="button" value="Update"/>

Business Expense Application - View and Edit

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1607151
Item Description	Quantity	
<input type="text"/>	<input type="text"/>	
Unit Price	Total Price	Vendor
<input type="text"/> NZD	<input type="text"/> 0 NZD	<input type="text"/>
Reason for Purchase	<input type="text"/>	

Please ensure all fields are filled out correctly.

Approved?

Fig 8.6

Business Expense Application - View and Edit

Approved?

Employee Name	Email Address	Tracking Number
William Shen	w.shen@kings.net.nz	BS1607152
Item Description	Quantity	
Chicken Niblets	100	
Unit Price	Total Price	Vendor
1.21 NZD	121 NZD	McDonalds
Reason for Purchase		
I'm hungry :)		

Update the relevant data and press 'Update'.

Update

Past Applications (Approved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250
BS1106151	William Shen	Dell Projector	1	1200
BS1607152	William Shen	Chicken Niblets	100	121

Select a row from the tables above:

View or Edit **Delete** Record successfully updated. **Search** **New Application**

Business Expense Application for William Shen has been approved.

Date of Approval: 16/07/2015
Details for BS1607152

Item Description: Chicken Niblets (Quantity: 100)
Unit Price: \$1.21 NZD
Total Price: \$121 NZD
Vendor: McDonalds

Reason for Expense: I'm hungry :)

You have been approved for this business expense. You may now go ahead and purchase the relevant expense!

Fig 9.1

The figure consists of two side-by-side screenshots of a 'Contact Support' application window. Both windows have a blue header bar with the title 'Contact Support' and a small logo. The left window shows filled input fields: 'Name' (William Shen), 'Email' (w.shen@kings.net.nz), and a 'Message' box containing 'I am having troubles logging in please help.' A checked checkbox 'This Email is not spam.' is present, along with a note 'The email may take up to 10 seconds to send. Thank you for your patience.' and a 'Send' button. The character count '44/1000' is shown in the top right. The right window shows the same fields but with empty input boxes. A checked checkbox 'This Email is not spam.' is present, along with a green success message 'Your message has been successfully sent. Thank You.' and a 'Send' button. The character count '0/1000' is shown in the top right.

Fig 9.2

The figure consists of two side-by-side screenshots of a 'Contact Support' application window. Both windows have a blue header bar with the title 'Contact Support' and a small logo. The left window shows filled input fields: 'Name' (William Shen), 'Email' (w.shen@kings.net.nz), and a 'Message' box containing 'I am having troubles logging in please help.' An unchecked checkbox 'This Email is not spam.' is present, along with a note 'The email may take up to 10 seconds to send. Thank you for your patience.' and a 'Send' button. The character count '44/1000' is shown in the top right. The right window shows the same fields but with empty input boxes. An unchecked checkbox 'This Email is not spam.' is present, along with a red error message 'Please ensure that all fields are filled out and the 'checkbox' is checked.' and a 'Send' button. The character count '0/1000' is shown in the top right.

shen.odinhealth@gmail.com

to w.shen+odin ▾



Support Request for 'Odin Health Business Manager'

Time Submitted: 15/07/2015 16:48:36

Name: William Shen

Email Address: w.shen@kings.net.nz

Problem Description:

I am having troubles logging in please help.

Fig 9.3

Contact Support

Contact Support

Name:

IliamWilliamWilliamWilliamWilliam Shen

Email:

yuiopasdfghjklzcvbnm@123456789.xyz

Message:

This is not spam. This is not spam. This is not ^
spam. This is not spam. This is not spam.
This is not spam. This is not spam. This is not
spam. This is not spam. This is not spam. This
is not spam. This is not spam. This is not
spam. This is not spam. This is not spam. This
is not spam. This is not spam. This is not
spam. This is not spam. This is not spam. This
is not spam!!

This Email is not spam. 1000/1000

The email may take up to 10 seconds to send. Thank you for your patience.

Send

Contact Support

Contact Support

Name:

Email:

Message:

This Email is not spam. 0/1000

Your message has been successfully sent. Thank You.

Send

Fig 9.4

Contact Support

Contact Support

Name:

Email:

Message:

This Email is not spam.

6/1000

The email may take up to 10 seconds to send. Thank you for your patience.

Send

Contact Support

Contact Support

Name:

Email:

Message:

This Email is not spam.

6/1000

Please check that the email address you entered was in a correct format.

Send

Support Request for 'Odin Health Business Manager'

Time Submitted: 15/07/2015 16:57:50

Name: William Shen

Email Address: `qwertyuiopasdfghjklzxcvbnm@123456789.xyz`

Problem Description:

Fig 9.5

The diagram illustrates a user interface for contacting support, showing two states of the form.

Screenshot 1 (Left):

- Form title: Contact Support
- Fields:
 - Name: [Input field]
 - Email: [Input field]
 - Message: [Text area]
- Checkboxes:
 - This Email is not spam.
 - 0/1000
- Text: The email may take up to 10 seconds to send. Thank you for your patience.
- Buttons:
 - Send (disabled)

Screenshot 2 (Right):

- Form title: Contact Support
- Fields:
 - Name: [Input field]
 - Email: [Input field]
 - Message: [Text area]
- Checkboxes:
 - This Email is not spam.
 - 0/1000
- Text: Please ensure that all fields are filled out and the 'checkbox' is checked.
- Buttons:
 - Send (enabled)

Fig 10.1

Investor
Ministry of Business Development

Date Amount
8/07/2015 156423 NZD Add

Enter all relevant data into the controls and press "Add".

The screenshot shows the 'Investment' form with the following data:

ID	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
31	Ministry of Business Develop...	8/07/2015	156423

Below the table are 'Edit' and 'Delete' buttons. At the bottom, there is another input section for a new investment record:

Investor
Date Amount
15/07/2015 NZD Add

Investment information added into database.

Fig 10.2

Investor
endo Nintendo Nintendo Nintendo Nintendo Nintendo Nintendo

Date Amount
11/07/2013 999999999 NZD Add

Enter all relevant data into the controls and press "Add".

The screenshot shows the 'Investment' form with the following data:

ID	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
31	Ministry of Business Develop...	8/07/2015	156423
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...

Below the table are 'Edit' and 'Delete' buttons. At the bottom, there is another input section for a new investment record:

Investor
Date Amount
15/07/2015 NZD Add

Investment information added into database.

Fig 10.3

Investor			
Date	Amount	NZD	Add
15/07/2015		NZD	Add

Enter all relevant data into the controls and press "Add".

Investor			
Date	Amount	NZD	Add
15/07/2015		NZD	Add

Please ensure all fields are filled out correctly.

Fig 10.4

ID	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
31	Ministry of Business Develop...	8/07/2015	156423
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...

Delete Record

Are you sure you want to delete this Investment record?

Yes **No**

ID	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...

Edit **Record successfully deleted.** Delete

Fig 10.5

ID	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...

Edit Delete



Investment - Edit

ID Investor
32 Nintendo Nintendo Nintendo Nintendo Ninten...

Date Amount
11/07/2013 NZD

Update

Close Update all relevant data and press "Update".

ID	Investor	Date	< Amount
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
27	Shanghai Medical Investors	3/03/2015	34843

Fig 10.6

> ...	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...

< ...	Investor	Date	Amount
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...
30	China HIMMS	5/02/2015	65421.56
29	Healthcare Solutions	15/06/2015	69612
27	Shanghai Medical Investors	3/03/2015	34843

ID	> Investor	Date	Amount
30	China HIMMS	5/02/2015	65421.56
29	Healthcare Solutions	15/06/2015	69612
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...
27	Shanghai Medical Investors	3/03/2015	34843

ID	< Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	34843
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56

ID	Investor	Date	> Amount
27	Shanghai Medical Investors	3/03/2015	34843
30	China HIMMS	5/02/2015	65421.56
29	Healthcare Solutions	15/06/2015	69612
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...

ID	Investor	< Date	Amount
29	Healthcare Solutions	15/06/2015	69612
27	Shanghai Medical Investors	3/03/2015	34843
30	China HIMMS	5/02/2015	65421.56
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...

ID	Investor	> Date	Amount
32	Nintendo Nintendo Nintend...	11/07/2013	9999999...
30	China HIMMS	5/02/2015	65421.56
27	Shanghai Medical Investors	3/03/2015	34843
29	Healthcare Solutions	15/06/2015	69612

Fig 11.1

Investment - Edit

ID	Investor
27	Shanghai Medical Investors
Date	Amount
3/03/2015	34843 NZD
<input type="button" value="Update"/>	
<input type="button" value="Close"/>	Update all relevant data and press "Update".

Fig 11.2

Investment - Edit

ID	Investor
27	Shanghai Medical Investors
Date	Amount
3/03/2015	123456 NZD
<input type="button" value="Update"/>	
<input type="button" value="Close"/>	Update all relevant data and press "Update".

Fig 11.3

Investment - Edit

ID	Investor
32	Pokemon Pokemon Pokemon Pokemon Pokemon
Date	Amount
11/07/2013	9999999999 NZD
<input type="button" value="Update"/>	
<input type="button" value="Close"/>	Update all relevant data and press "Update".

ID	Investor	Date	Amount
27	Shanghai Medical Investors	3/03/2015	123456
29	Healthcare Solutions	15/06/2015	69612
30	China HIMMS	5/02/2015	65421.56
32	Pokemon Pokemon Pokemo...	11/07/2013	9999999...

< ... Investor Date Amount

32	Nintendo Nintendo Nintend...	11/07/2013	9999999...
30	China HIMMS	5/02/2015	65421.56
29	Healthcare Solutions	15/06/2015	69612
27	Shanghai Medical Investors	3/03/2015	123456

Record successfully updated.

Fig 11.4

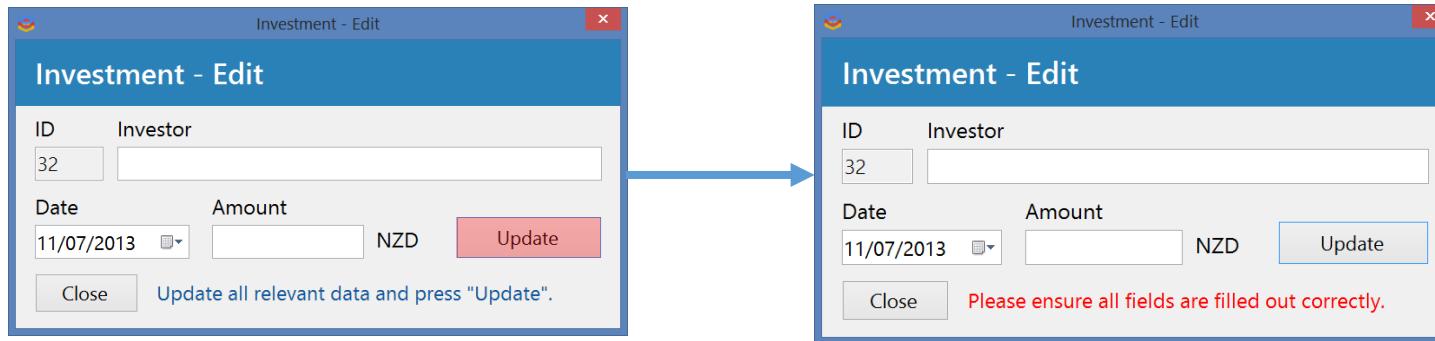


Fig 12.1



Fig 12.2



Fig 12.3



Fig 13.1 to 13.5

Main Menu

Welcome to the Main Menu:
Please select the form you wish to get to.

- Travel Application (Icon: Airplane)
- Business Application (Icon: Money with plus sign)
- Revenues (Icon: Bag with dollar sign)
- Travel Expenses (Icon: Car)
- Business Expenses (Icon: Calculator)
- Contact Support (Icon: Question mark)
- Investment (Icon: Building)
- Wages (Icon: Money with dollar sign)
- Account Manager (Icon: Padlock)
- Settings (Icon: Gear)

For a simplified menu, use the 'View' option at the top of the form.

Travel Application - Overview

Pending Applications (Unapproved)

Tracking No.	Full Name	Description	Quantity	Total Price
TV0507150	Richard Hendrix	OnePlus One OnePlus One OnePlus ...	99999	9999900000
TV0607150	Daniel Dyer	LED Lamps	20	460
TV0706150	Marco Yu	Dell U251H Monitors	12	6552

Past Applications (Approved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250
BS1106151	William Shen	Dell Projector	1	1200
BS1607152	William Shen	Chicken Niblets	100	121

Select a row from the tables above:

Business Expense Applications - Overview

Pending Applications (Unapproved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS1607151	William Shen	OnePlus One OnePlus One OnePlus ...	99999	9999900000
BS2105150	William Shen	LED Lamps	20	460
BS2305150	Marco Yu	Dell U251H Monitors	12	6552

Past Applications (Approved)

Tracking No.	Full Name	Description	Quantity	Total Price
BS0607150	Daniel Dyer	Nexus 9 64GB	5	4250
BS1106151	William Shen	Dell Projector	1	1200
BS1607152	William Shen	Chicken Niblets	100	121

Select a row from the tables above:

Revenues - Overview

Revenue Records

PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
234234	420Dustil	21/06/2015	Microsoft	23593.44
24356	1234	15/07/2015	56556	246

Select a row from the tables above:

Travel Expense - Overview

Tracking No.	Name	Period	Amount (NZD)	Reimbursed
TV0706150	John Shen	28/06/2015 - 16/07/2015	9191.89	False
TV0706151	William Shen	2/07/2015 - 5/07/2015	3049.92	True

Business Expense - Overview

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
0123456789	William William ...	16/07/2015	Servers Servers Servers Servers ...	999899990000
BS11061501	Richard Hendrix	15/07/2015	Microwave	350
BS11061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
12345	William Shen	16/07/2015	Samsung TV	3000
BS2105150	William Shen	6/07/2015	LED Lamps	4600

Select a row from the tables above:

Print by:

- Expenses Between 16/07/2015 and 16/07/2015
- Completed Reimbursement
- Employee Name Daniel Dyer
- Pending Reimbursement
- Print all data

Print Total **Print**

Fig 13.6 to 13.10

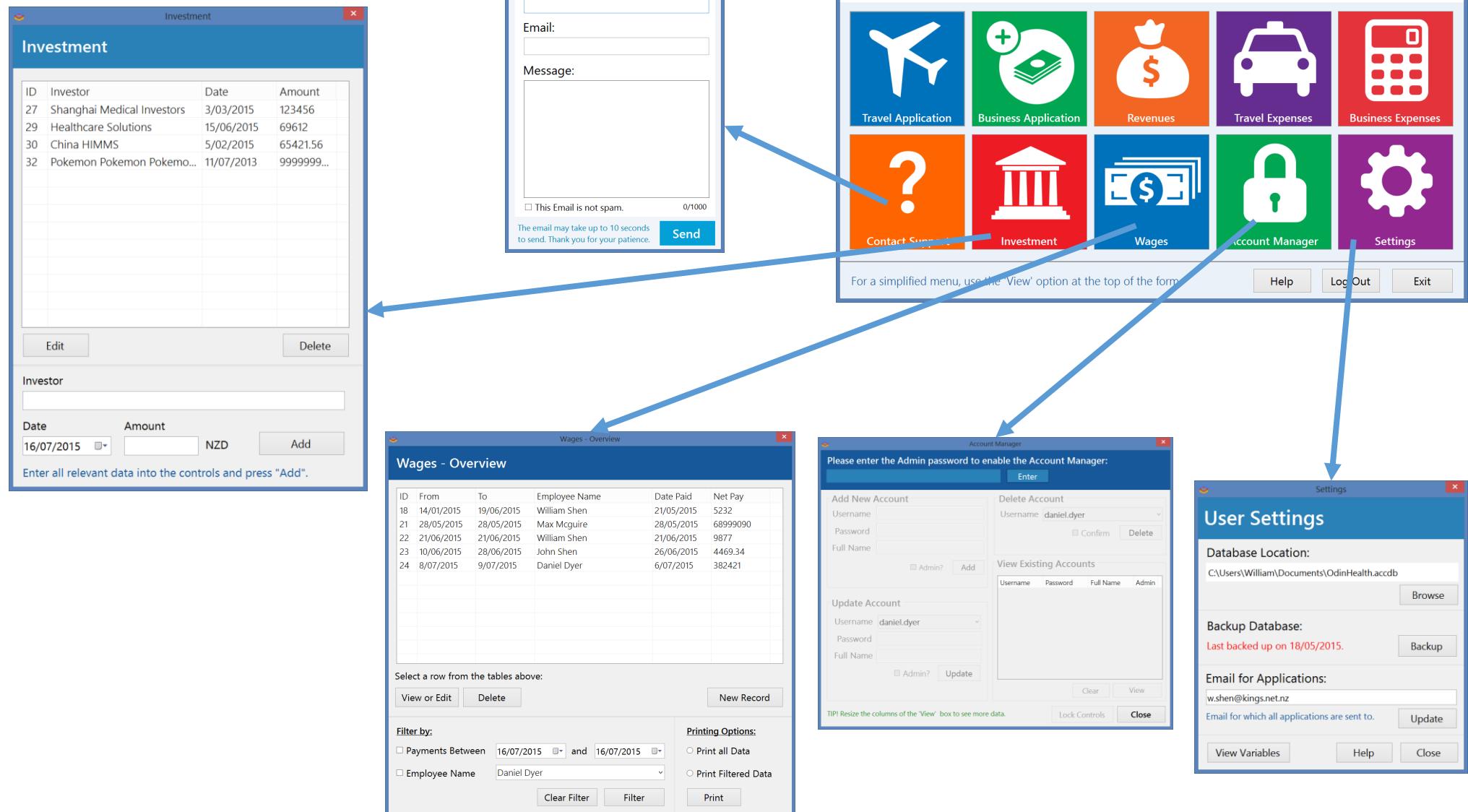


Fig 13.11 —

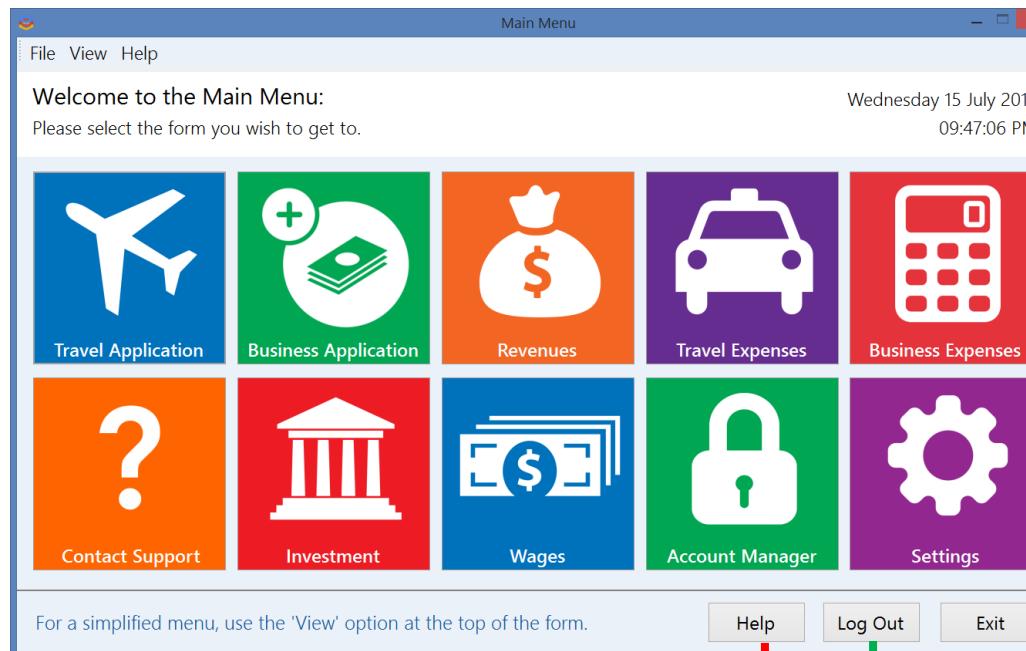


Fig 13.12 —

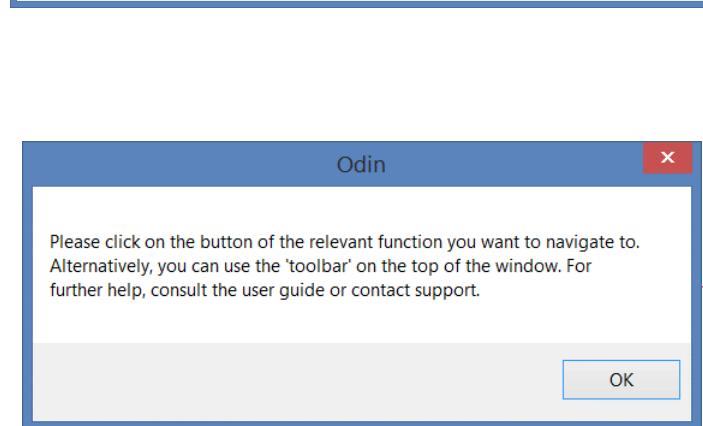


Fig 13.13 —

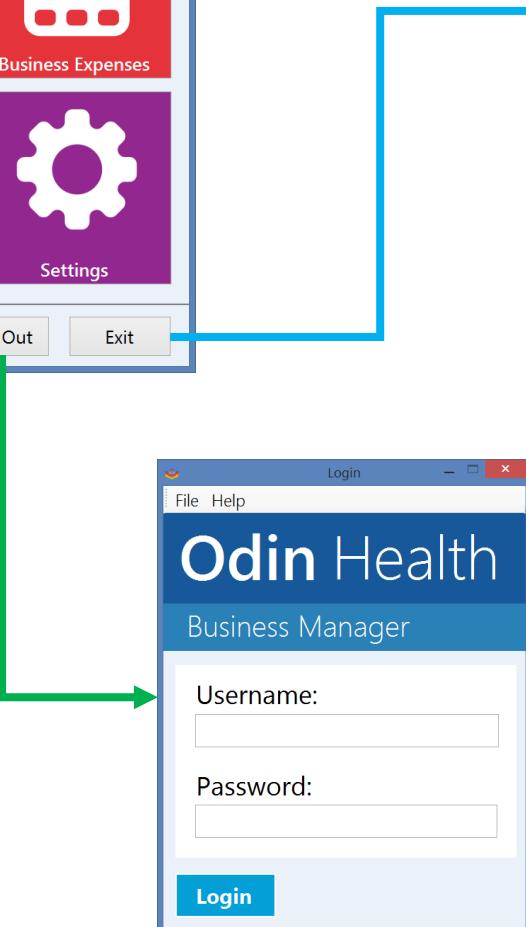


Fig 14.1

Revenues - Overview

Revenue Records				
PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
234234	420Dustll	21/06/2015	Microsoft	23593.44
24356	1234	15/07/2015	56556	246

Select a row from the tables above:

View or Edit **Delete** **Search** **New Revenue**

Filter by:

Payments Between and **Filter** **Clear**

Revenues - New

PO Number	Invoice Number	Invoice Date
<input type="text"/>	<input type="text"/>	<input type="text" value="16/07/2015"/>

Customer (Purchaser)

Subtotal **GST** **Total**

NZD NZD NZD

Fill in all the relevant fields and then click 'Add'.

Close **Add**

Fig 14.2

Revenues - Overview

Revenue Records				
PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
234234	420Dustll	21/06/2015	Microsoft	23593.44
24356	1234	15/07/2015	56556	246

Select a row from the tables above:

View or Edit **Delete** **Search** **New Revenue**

Filter by:

Payments Between and **Filter** **Clear**

Please enter the PO Number.

2435

Cancel **Search**

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
2435	23424	7/05/2015

Customer (Purchaser)

Subtotal	GST	Total			
235234	NZD	23423	NZD	258657	NZD

Update in all relevant fields and then click 'Update'.

Close **Delete** **Update**

Fig 14.3

Revenues - Overview

Revenue Records				
PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
234234	420Dustll	21/06/2015	Microsoft	23593.44
24356	1234	15/07/2015	56556	246

Select a row from the tables above:

View or Edit **Delete** **Search** **New Revenue**

Delete Record

Are you sure you want to delete this Revenue record?

Yes **No**

Record deleted.

Search **New Revenue**

Fig 14.4

Revenues - Overview

Revenue Records				
PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015		246

Select a row from the tables above:

View or Edit **Delete** **Search** **New Revenue**

Filter by:

Payments Between **4/06/2015** and **16/07/2015** **Filter** **Clear**

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
15421354	20151001	10/01/2015
Customer (Purchaser)		
ShenZhen Government		
Subtotal	GST	Total
799235.26 NZD	0.00 NZD	799235.26 NZD

Update in all relevant fields and then click 'Update'.

View or Edit **Delete** **Update**

Fig 14.5

Filter by:

Payments Between **4/06/2015** and **16/07/2015** **Filter** **Clear**

PO Number **Invoice No.** **Date** **Customer** **Total**

1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015	56556	246

Fig 14.6

Filter by:

Payments Between **4/06/2015** and **16/07/2015** **Filter** **Clear**

Data successfully filtered.

Revenue Records

PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015	56556	246

Select a row from the tables above:

View or Edit **Delete** **Search** **New Revenue**

Filter by:

Payments Between **16/07/2015** and **16/07/2015** **Filter** **Clear**

Fig 15.1

Revenues - New

PO Number	Invoice Number	Invoice Date
<input type="text"/>	<input type="text"/>	16/07/2015 <input type="button" value="▼"/>

Customer (Purchaser)

Subtotal GST Total

<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/> NZD
--------------------------	--------------------------	--------------------------

Fill in all the relevant fields and then click 'Add'.

Fig 15.2

Revenues - New

PO Number	Invoice Number	Invoice Date
170620150	165423	16/07/2015 <input type="button" value="▼"/>

Customer (Purchaser)

Software Company Inc

Subtotal GST Total

95000	NZD	100	NZD	95100	NZD
-------	-----	-----	-----	-------	-----

Fill in all the relevant fields and then click 'Add'.

Fig 15.3

Revenues - New

PO Number	Invoice Number	Invoice Date
DOGE123	3457236	16/07/2015 <input type="button" value="▼"/>

Customer (Purchaser)

William Shen William Shen William Shen William

Subtotal GST Total

9999999999	NZD	9999999999	NZD	19999999998	NZD
------------	-----	------------	-----	-------------	-----

Fill in all the relevant fields and then click 'Add'.

Revenue Records

PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015	56556	246
DOGE123	3457236	16/07/2015	William Shen William Shen Willia...	19999999998
170620150	165423	16/07/2015	Software Company Inc	95100

Revenue Records

PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015	56556	246
170620150	165423	16/07/2015	Software Company Inc	95100

Select a row from the tables above:

Record added.

Fig 15.4

Revenues - New

PO Number	Invoice Number	Invoice Date
<input type="text"/>	<input type="text"/>	16/07/2015 <input type="button" value="▼"/>
Customer (Purchaser)		
<input type="text"/>		
Subtotal	GST	Total
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/> NZD

Fill in all the relevant fields and then click 'Add'.

Revenues - New

PO Number	Invoice Number	Invoice Date
<input type="text"/>	<input type="text"/>	16/07/2015 <input type="button" value="▼"/>
Customer (Purchaser)		
<input type="text"/>		
Subtotal	GST	Total
<input type="text"/> NZD	<input type="text"/> NZD	<input type="text"/> NZD

Please ensure all fields are filled out correctly.

Fig 15.5

Revenues - New

PO Number	Invoice Number	Invoice Date
DOGE123	greetings	16/07/2015 <input type="button" value="▼"/>
Customer (Purchaser)		
<input type="text"/>		
Subtotal	GST	Total
<input type="text"/> 123 NZD	<input type="text"/> 123 NZD	<input type="text"/> 246 NZD

Fill in all the relevant fields and then click 'Add'.

Revenues - New

PO Number	Invoice Number	Invoice Date
DOGE123	greetings	16/07/2015 <input type="button" value="▼"/>
Customer (Purchaser)		
<input type="text"/>		
Subtotal	GST	Total
<input type="text"/> 123 NZD	<input type="text"/> 123 NZD	<input type="text"/> 246 NZD

A record for the PO Number already exists.

Fig 15.6

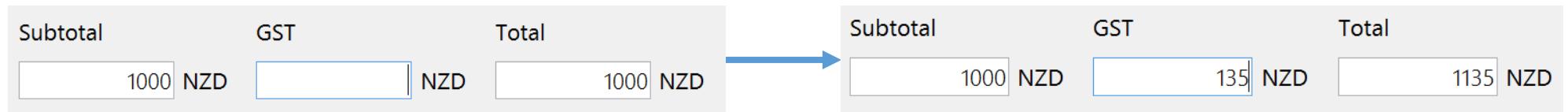


Fig 16.1

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
170620150	165423	16/07/2015
Customer (Purchaser)		
Software Company Inc		
Subtotal	GST	Total
95000 NZD	100 NZD	95100 NZD

Update in all relevant fields and then click 'Update'. Close Delete Update

Fig 16.2

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
DOGE123	3457236	16/07/2015
Customer (Purchaser)		
William Shen William Shen William Shen William		
Subtotal	GST	Total
9999999999 NZD	9999999999 NZD	19999999998 NZD

Update in all relevant fields and then click 'Update'. Close Delete Update

Revenue Records

PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015	56556	246
170620150	165423	16/07/2015	Software Company Inc	95100

Delete Application

Are you sure you want to delete this Business Expense Application?

Yes No

Fig 16.3

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
170620150	165423	16/07/2015
Customer (Purchaser)		
Software Company Inc		
Subtotal	GST	Total
95000 NZD	1200 NZD	96200 NZD
Update in all relevant fields and then click 'Update'.		
<input type="button" value="Close"/>	<input type="button" value="Delete"/>	<input type="button" value="Update"/>

Revenue Records

PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015	56556	246
170620150	165423	16/07/2015	Software Company Inc	96200

Revenue Records

PO Number	Invoice No.	Date	Customer	Total
15421354	20151001	10/01/2015	ShenZhen Government	799235.26
2435	23424	7/05/2015	Pokemon	258657
1234542465	Tec	4/06/2015	Shanghai Kingstar Winning123	4657
24356	1234	15/07/2015	56556	246
170620150	165423	16/07/2015	Software Company Inc Software ...	19999999998

Fig 16.5

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
170620150		16/07/2015
Customer (Purchaser)		
Software Company Inc		
Subtotal	GST	Total
95000 NZD	1200 NZD	96200 NZD
Update in all relevant fields and then click 'Update'.		
<input type="button" value="Close"/>	<input type="button" value="Delete"/>	<input type="button" value="Update"/>

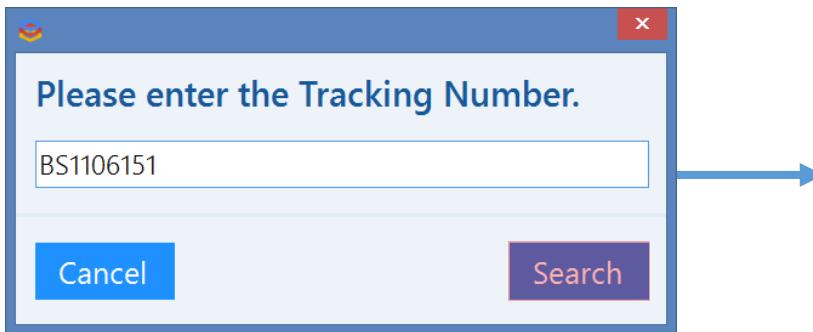
Please ensure all fields are filled out correctly.

Fig 16.4

Revenues - View and Edit

PO Number	Invoice Number	Invoice Date
170620150	165423	16/07/2015
Customer (Purchaser)		
Software Company Inc Software Company Inc Software		
Subtotal	GST	Total
9999999999 NZD	9999999999 NZD	19999999998 NZD
Update in all relevant fields and then click 'Update'.		
<input type="button" value="Close"/>	<input type="button" value="Delete"/>	<input type="button" value="Update"/>

Fig 17.1



A screenshot of the "Business Expense Application - View and Edit" window. The window title bar includes the application name and a checked "Approved?" checkbox. The main form contains the following data:

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	BS1106151

Below this, there are fields for "Item Description" (Dell Projector), "Quantity" (1), "Unit Price" (1200 NZD), "Total Price" (1200 NZD), and "Vendor" (Dell.co.nz). A "Reason for Purchase" section notes "For viewing presentations to the potential purchasers". At the bottom, a message says "Update the relevant data and press 'Update'." and there are "Close", "Delete", and "Update" buttons.

Fig 17.2

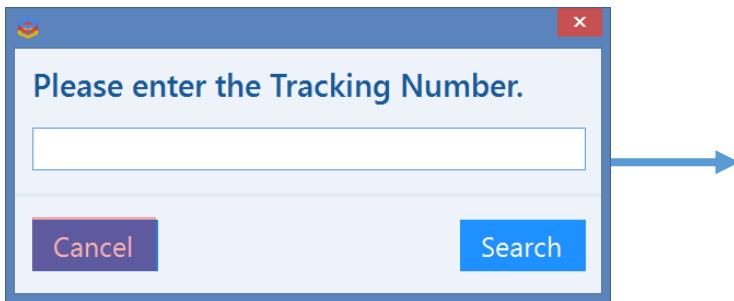


Fig 18.1

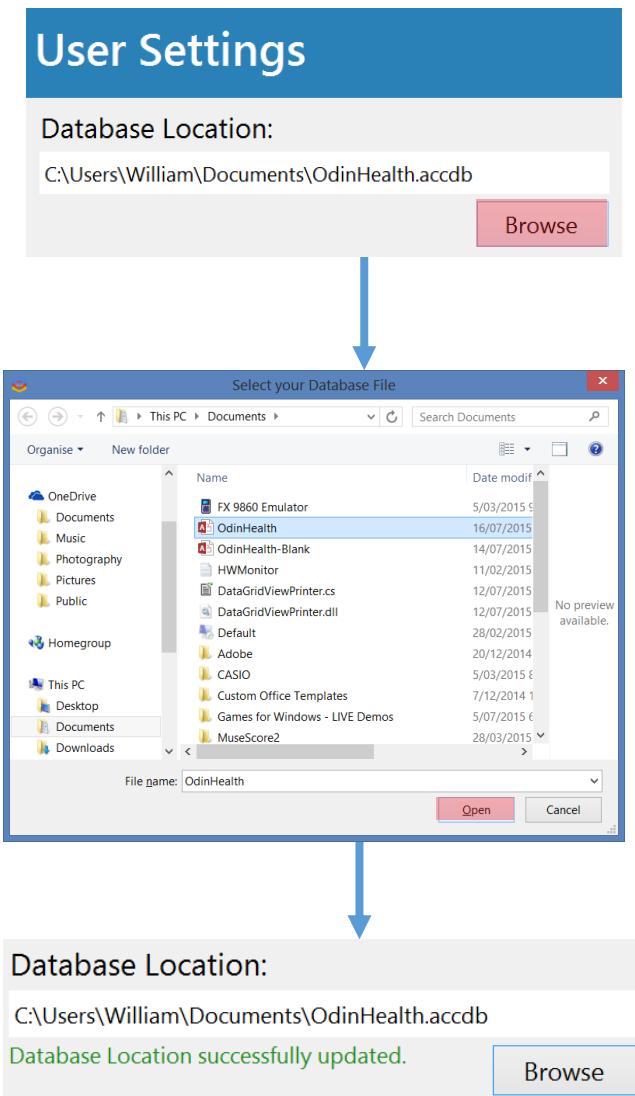


Fig 18.2

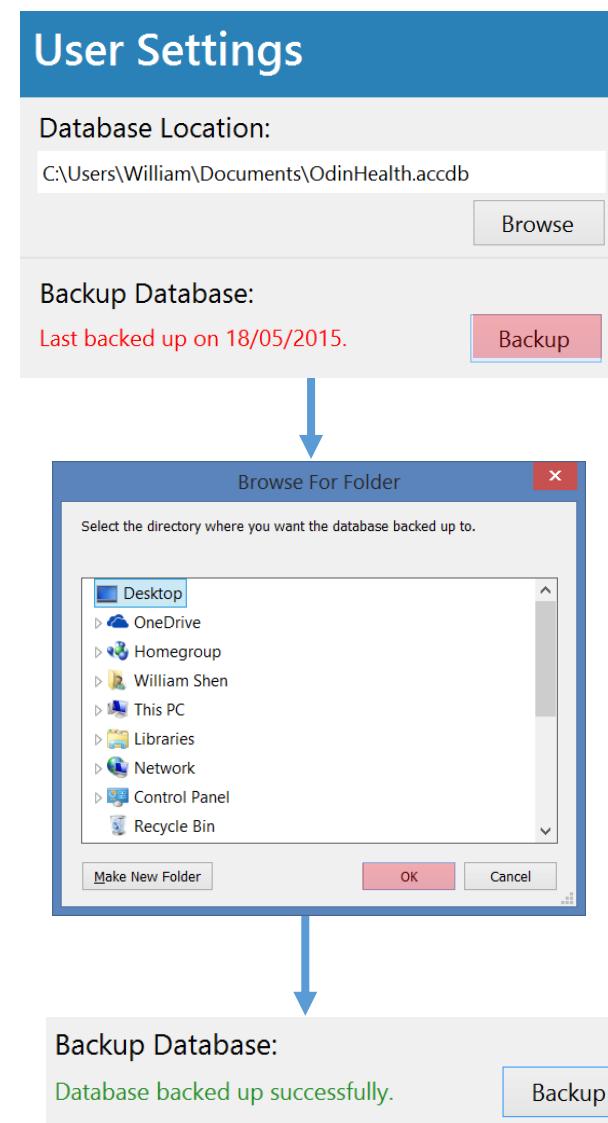


Fig 18.3

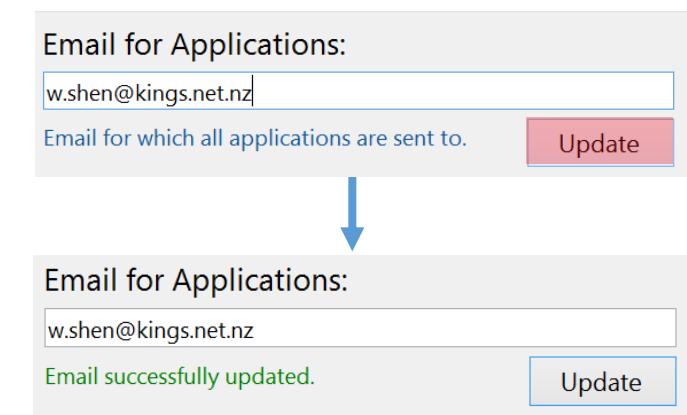


Fig 18.4

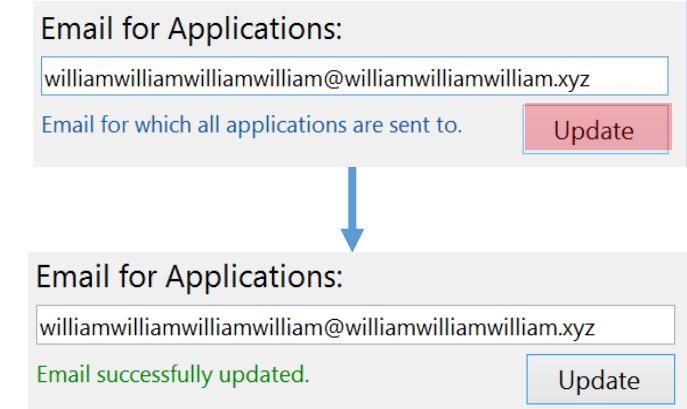


Fig 18.5

Email for Applications:

Email for which all applications are sent to.



Email for Applications:

Email address in incorrect format.

Fig 18.7

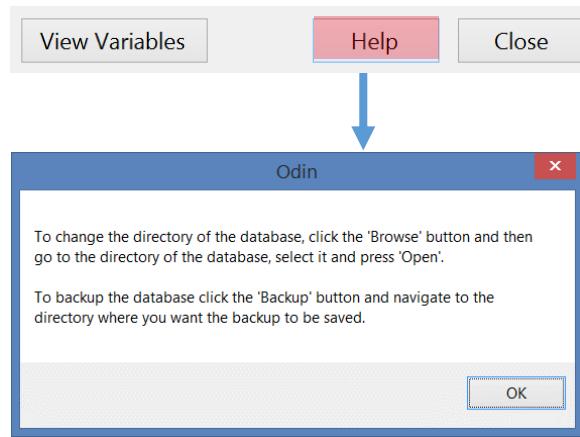


Fig 18.6

Settings

User Settings

Database Location:

Backup Database:

Last backed up on 16/07/2015.

Email for Applications:

Email for which all applications are sent to.



Settings - Variables

Please enter the Admin password to view variables:

dbLocation	<input type="text"/>	String
backupDate	<input type="text"/>	String
accountAdmin	<input type="text"/>	Boolean
accountUsername	<input type="text"/>	String
accountFullName	<input type="text"/>	String
applicationEmail	<input type="text"/>	String
TEADate	<input type="text"/>	Date
TEACount	<input type="text"/>	Integer
BEADate	<input type="text"/>	Date
BEACount	<input type="text"/>	Integer

These variables cannot be directly edited. Please use the 'Settings' form to change any relevant variables.

Fig 18.8

Settings

User Settings

Database Location:

Backup Database:

Last backed up on 16/07/2015.

Email for Applications:

Email for which all applications are sent to.



Fig 19.1

Settings - Variables

Please enter the Admin password to view variables:

dbLocation	<input type="text"/>	String
------------	----------------------	--------

dbLocation C:\Users\William\Documents\OdinHealth.accdb String

backupDate 16/07/2015 String

accountAdmin True Boolean

accountUsername william.shen String

accountFullName William Shen String

applicationEmail w.shen@kings.net.nz String

TEADate 16/07/2015 Date

TEACount 0 Integer

BEADate 16/07/2015 Date

BEACount 3 Integer

These variables cannot be directly edited. Please use the 'Settings' form to change any relevant variables.

Fig 19.2

Settings - Variables

Please enter the Admin password to view variables:

dbLocation	<input type="text"/>	String
------------	----------------------	--------

dbLocation C:\Users\William\Documents\OdinHealth.accdb String

Enter Incorrect password.

Fig 19.3

Settings - Variables

Please enter the Admin password to view variables:

***** Correct password.

dbLocation	<input type="text"/>	String
------------	----------------------	--------

dbLocation C:\Users\William\Documents\OdinHealth.accdb String

backupDate 16/07/2015 String

accountAdmin True Boolean

accountUsername william.shen String

accountFullName William Shen String

applicationEmail w.shen@kings.net.nz String

TEADate 16/07/2015 Date

TEACount 0 Integer

BEADate 16/07/2015 Date

BEACount 3 Integer

These variables cannot be directly edited. Please use the 'Settings' form to change any relevant variables.

Fig 20.1

Record not found.

A record for the tracking number does not exist in the database.

Press 'OK' to continue.

Fig 21.1

Travel Application - Overview

Pending Applications (Unapproved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0607150	Daniel Dyer	London UK	5	5602
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308

Past Applications (Approved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0706151	William Shen	Australia, USA, India	26	140000

Select a row from the tables above:

View or Edit **Delete** **Search** **New Application**

Travel Application - New

Employee Name: William Shen Email Address: william.shen@odinhealth.co.nz Tracking Number: TV1607150

Destinations (Countries and Cities): London UK Duration: 5 Days

Reason for Trip:

Please estimate the following costs:

Domestic Travel Costs	NZD	Accommodation Costs	NZD
International Travel Costs	NZD	Total Costs	NZD

Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.

Close **Submit**

Fig 21.2

Select a row from the tables above:

View or Edit **Delete** **Search** **New Application**

Please enter the Tracking Number.

TV0706151

Cancel **Search**

Travel Application - View and Edit

Employee Name: William Shen Email Address: wshen@kings.net.nz Tracking Number: TV0706151

Destinations (Countries and Cities): Australia, USA, India Duration: 26 Days

Reason for Trip: Learn about the different cultures in other countries. Make new business deals

Please estimate the following costs:

Domestic Travel Costs	10000 NZD	Accommodation Costs	10000 NZD
International Travel Costs	120000 NZD	Total Costs	140000 NZD

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Fig 21.3

Past Applications (Approved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0607150	Daniel Dyer	London UK	5	5602
TV0706151	William Shen	Australia, USA, India	26	140000

Select a row from the tables above:

View or Edit **Delete** **Search** **New Application**

Delete Travel Expense Application

Are you sure you want to delete this Travel Expense Application?

Yes **No**

Past Applications (Approved)

Tracking No.	Full Name	Destinations
TV0706151	William Shen	Australia, USA, India

Fig 21.4

Travel Application - Overview

Pending Applications (Unapproved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308

Past Applications (Approved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0706151	William Shen	Australia, USA, India	26	140000

Select a row from the tables above:

View or Edit **Delete** **Search** **New Application**

Travel Application - View and Edit

Approved?

Employee Name: Richard Hendrix Email Address: william.shen@odinhealth.co.nz Tracking Number: TV0507150

Destinations (Countries and Cities): San Francisco, New York, Chicago, USA Duration: 21 Days

Reason for Trip: TechCrunch Disrupt

Please estimate the following costs:

Domestic Travel Costs: 1234 NZD	Accommodation Costs: 345 NZD
International Travel Costs: 234 NZD	Total Costs: 1813 NZD

Update the relevant data and press 'Update'. **Close** **Delete** **Update**

Fig 22.1

Travel Application - New

Employee Name: William Shen Email Address: william.shen@odinhealth.co.nz Tracking Number: TV1607150

Destinations (Countries and Cities):

Duration: Days

Fig 22.2

Travel Application - New

Employee Name: William Shen Email Address: william.shen@odinhealth.co.nz Tracking Number: TV1607150

Destinations (Countries and Cities):

Duration: Days

Reason for Trip:

Please estimate the following costs:

Domestic Travel Costs: NZD Accommodation Costs: NZD

International Travel Costs: NZD Total Costs: NZD

Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.

Close **Submit**

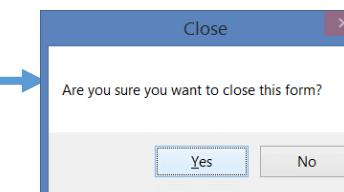


Fig 22.3

Travel Application - New

Employee Name	Email Address	Tracking Number	
William Shen	william.shen@odinhealth.co.nz	TV1607150	
Destinations (Countries and Cities)		Duration	
Sydney and Melbourne Australia		5 Days	
Reason for Trip			
Explore business solutions in other companies			
Please estimate the following costs:			
Domestic Travel Costs	500 NZD	Accommodation Costs	1000 NZD
International Travel Costs	1500 NZD	Total Costs	3000 NZD
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.			
<input type="button" value="Close"/>		<input type="button" value="Submit"/>	

Pending Applications (Unapproved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308
TV1607150	William Shen	Sydney and Melbourne Australia	5	3000

Travel Application for William Shen

Date of Submission: 16/07/2015
Return Email: william.shen@odinhealth.co.nz

Destinations: Sydney and Melbourne Australia (Duration: 5 Days)

Reason for Trip: Explore business solutions in other companies

Estimated Costs:

Domestic Travel Costs: \$500 NZD
International Travel Costs: \$1500 NZD
Accommodation Costs: \$1000 NZD
Total Costs: \$3000 NZD

To approve this application, open the Business Manager program and navigate to the Travel Applications, select the application click the 'View and Edit' button and then tick the approved box and 'Update'

Fig 22.4

Travel Application - New

Employee Name	Email Address	Tracking Number	
William Shen	william.shen@odinhealth.co.nz	TV1607153	
Destinations (Countries and Cities)		Duration	
Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai		9999 Days	
Reason for Trip			
Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai			
Please estimate the following costs:			
Domestic Travel Costs	9999999999 NZD	Accommodation Costs	9999999999 NZD
International Travel Costs	9999999999 NZD	Total Costs	2999999999 NZD
Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.			
<input type="button" value="Close"/>		<input type="button" value="Submit"/>	

Pending Applications (Unapproved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308
TV1607150	William Shen	Sydney and Melbourne Australia	5	3000
TV1607153	William Shen	Shanghai Shanghai Shanghai Shanghai	9999	2999999999

Travel Application for William Shen

Date of Submission: 16/07/2015
Return Email: william.shen@odinhealth.co.nz

Destinations: Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai (Duration: 9999 Days)

Reason for Trip: Shanghai Shanghai

Estimated Costs:

Domestic Travel Costs: \$9999999999 NZD
International Travel Costs: \$9999999999 NZD
Accommodation Costs: \$9999999999 NZD
Total Costs: \$2999999999 NZD

To approve this application, open the Business Manager program and navigate to the Travel Applications, select the application click the 'View and Edit' button and then tick the approved box and 'Update'

Fig 22.5

The screenshot shows a Windows application window titled "Travel Application - New". The form contains the following fields:

- Employee Name:** William Shen
- Email Address:** william.shen@odinhealth.co.nz
- Tracking Number:** TV1607154
- Destinations (Countries and Cities):** England, France, Spain, Italy, Europe
- Duration:** 24 Days
- Reason for Trip:** Vacation
- Please estimate the following costs:**
 - Domestic Travel Costs: [] NZD
 - Accommodation Costs: [] NZD
 - International Travel Costs: [] NZD
 - Total Costs: [] NZD
- Note:** Your application will be reviewed before it is approved. An email will be sent to you once it has been approved.

At the bottom right are "Close" and "Submit" buttons. A blue arrow points down from this window to the second window.

Second Window:

The second window is identical to the first, showing the same fields and note. At the bottom left, there is a red message: "Please ensure all fields are filled out correctly."

Fig 22.6

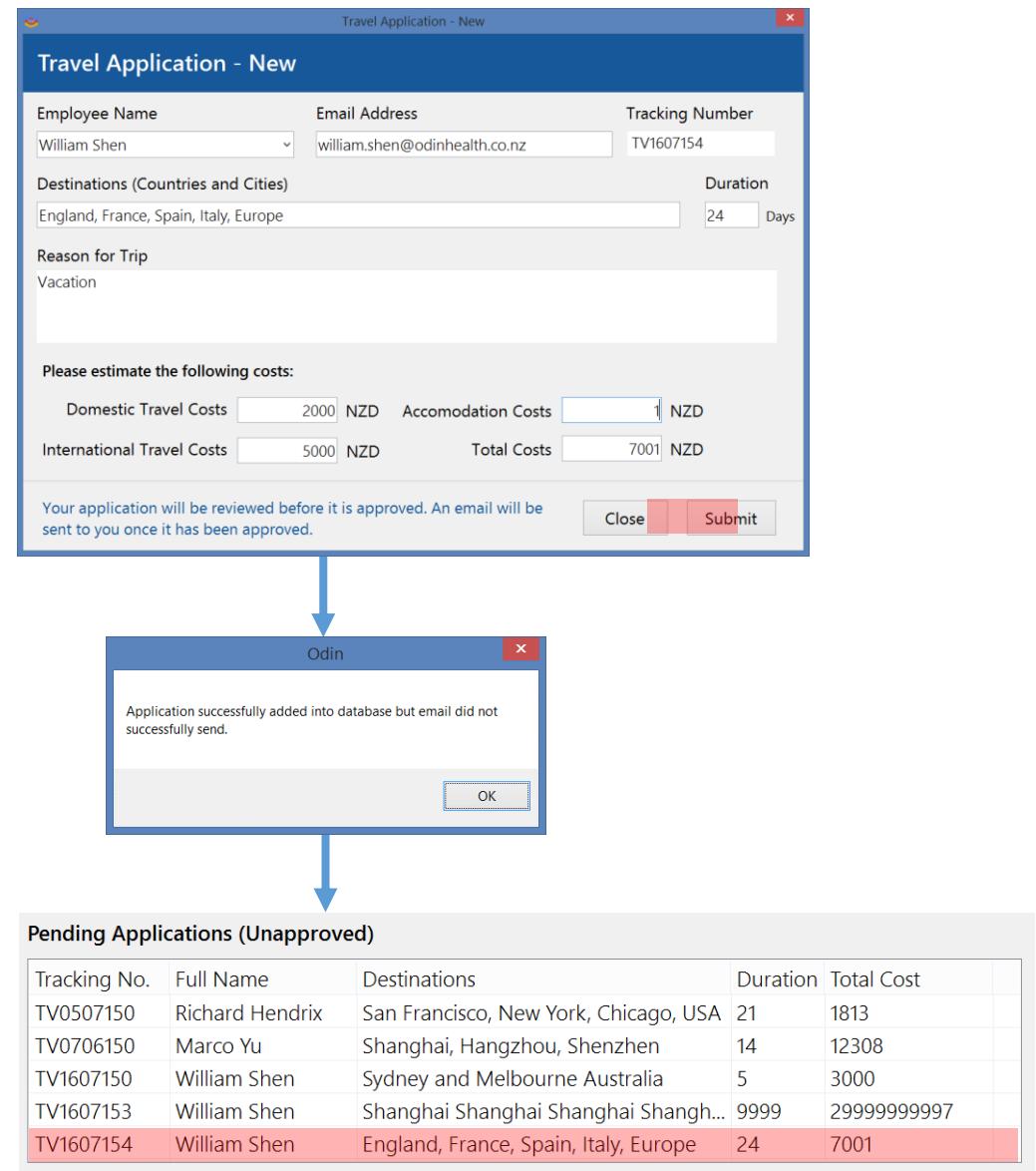


Fig 22.7

Please estimate the following costs:

Domestic Travel Costs	100	NZD	Accommodation Costs		NZD
International Travel Costs	200	NZD	Total Costs	300	NZD

Please estimate the following costs:

Domestic Travel Costs	100	NZD	Accommodation Costs	400	NZD
International Travel Costs	200	NZD	Total Costs	700	NZD

Fig 23.1

Travel Application - View and Edit

■ Approved?

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	TV1607154
Destinations (Countries and Cities)		Duration
England, France, Spain, Italy, Europe		24 Days
Reason for Trip		
Vacation		

Please estimate the following costs:

Domestic Travel Costs	2000	NZD	Accommodation Costs	1	NZD
International Travel Costs	5000	NZD	Total Costs	7001	NZD

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Fig 23.2

Travel Application - View and Edit

■ Approved?

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	TV1607154
Destinations (Countries and Cities)		Duration
England, France, Spain, Italy, Europe		24 Days
Reason for Trip		
Vacation		

Please estimate the following costs:

Domestic Travel Costs	2000	NZD	Accommodation Costs	1	NZD
International Travel Costs	5000	NZD	Total Costs	7001	NZD

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Delete Application

Are you sure you want to delete this Travel Expense Application?

Yes **No**

Pending Applications (Unapproved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308
TV1607150	William Shen	Sydney and Melbourne Australia	5	3000
TV1607153	William Shen	Shanghai Shanghai Shanghai Shang...	9999	29999999997

Fig 23.3

Travel Application - View and Edit

Approved?

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	TV1607150

Destinations (Countries and Cities) Duration

Sydney and Melbourne Australia	5 Days
--------------------------------	--------

Reason for Trip

Explore business solutions in other companies

Please estimate the following costs:

Domestic Travel Costs	750 NZD	Accommodation Costs	1000 NZD
International Travel Costs	1500 NZD	Total Costs	3250 NZD

Update the relevant data and press 'Update'.

→

Pending Applications (Unapproved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308
TV1607150	William Shen	Sydney and Melbourne Australia	5	3250
TV1607153	William Shen	Shanghai Shanghai Shanghai Shanghai...	9999	29999999997

Fig 23.4

Travel Application - View and Edit

Approved?

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	TV1607153

Destinations (Countries and Cities) Duration

Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai	9999 Days
--	-----------

Reason for Trip

Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai
Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai
Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai Shanghai

Please estimate the following costs:

Domestic Travel Costs	999999999 NZD	Accommodation Costs	999999999 NZD
International Travel Costs	999999999 NZD	Total Costs	2999999997 NZD

Update the relevant data and press 'Update'.

↓

Pending Applications (Unapproved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0507150	Richard Hendrix	San Francisco, New York, Chicago, USA	21	1813
TV0706150	Marco Yu	Shanghai, Hangzhou, Shenzhen	14	12308
TV1607150	William Shen	Sydney and Melbourne Australia	5	3250
TV1607153	William Shen	Shanghai Shanghai Shanghai Shanghai...	9999	29999999997

Please ensure all fields are filled out correctly.

Fig 23.6

Travel Application - View and Edit

Approved?

Employee Name	Email Address	Tracking Number
William Shen	william.shen@odinhealth.co.nz	TV1607150

Destinations (Countries and Cities) Duration

Sydney and Melbourne Australia 5 Days

Reason for Trip

Explore business solutions in other companies

Please estimate the following costs:

Domestic Travel Costs	750 NZD	Accommodation Costs	1000 NZD
International Travel Costs	1500 NZD	Total Costs	3250 NZD

Update the relevant data and press 'Update'.

Close **Delete** **Update**

Past Applications (Approved)

Tracking No.	Full Name	Destinations	Duration	Total Cost
TV0706150	William Shen	Australia, USA, India	26	140000
TV1607150	William Shen	Sydney and Melbourne Australia	5	3250

Travel Application for William Shen has been approved.

Date of Approval: 16/07/2015

Details for #TV1607150

Destinations: Sydney and Melbourne Australia (Duration: 5 Days)

Reason for Trip: Explore business solutions in other companies

Estimated Costs:

Domestic Travel Costs: \$750 NZD

International Travel Costs: \$1500 NZD

Accommodation Costs: \$1000 NZD

Total Costs: \$3250 NZD

You have been approved for this travel application. You may now go ahead and book or purchase the relevant expenses. Have a safe journey!

Fig 24.1

Travel Expense - Overview

Tracking No.	Name	Period	Amount (NZD)	Reimbursed
TV0706150	John Shen	28/06/2015 - 16/07/2015	9191.89	False
TV0706151	William Shen	2/07/2015 - 5/07/2015	3049.92	True

View or Edit **Delete** **Print** **Search** **New Expense**

Travel Expense

Travel Expense - New

Tracking Number	Employee Name	
<input type="text"/>	William Shen	
Date Incurred	Expense Type	Payment Method
16/07/2015	<input type="text"/>	<input type="text"/>
Item Description	<input type="checkbox"/> Reimburse?	<input type="checkbox"/> Bill Client?
Foreign Amount	Exchange Rate	NZD Amount
<input type="text"/> RMB	RMB to NZD	0 NZD

Please enter the Tracking Number.

Clear **Save**

ID	Date	Description	Type	NZD	Reimbursed
<input type="text"/>	<input type="checkbox"/>				

View or Edit **Delete** **Print** **New Expense**

Fig 24.2

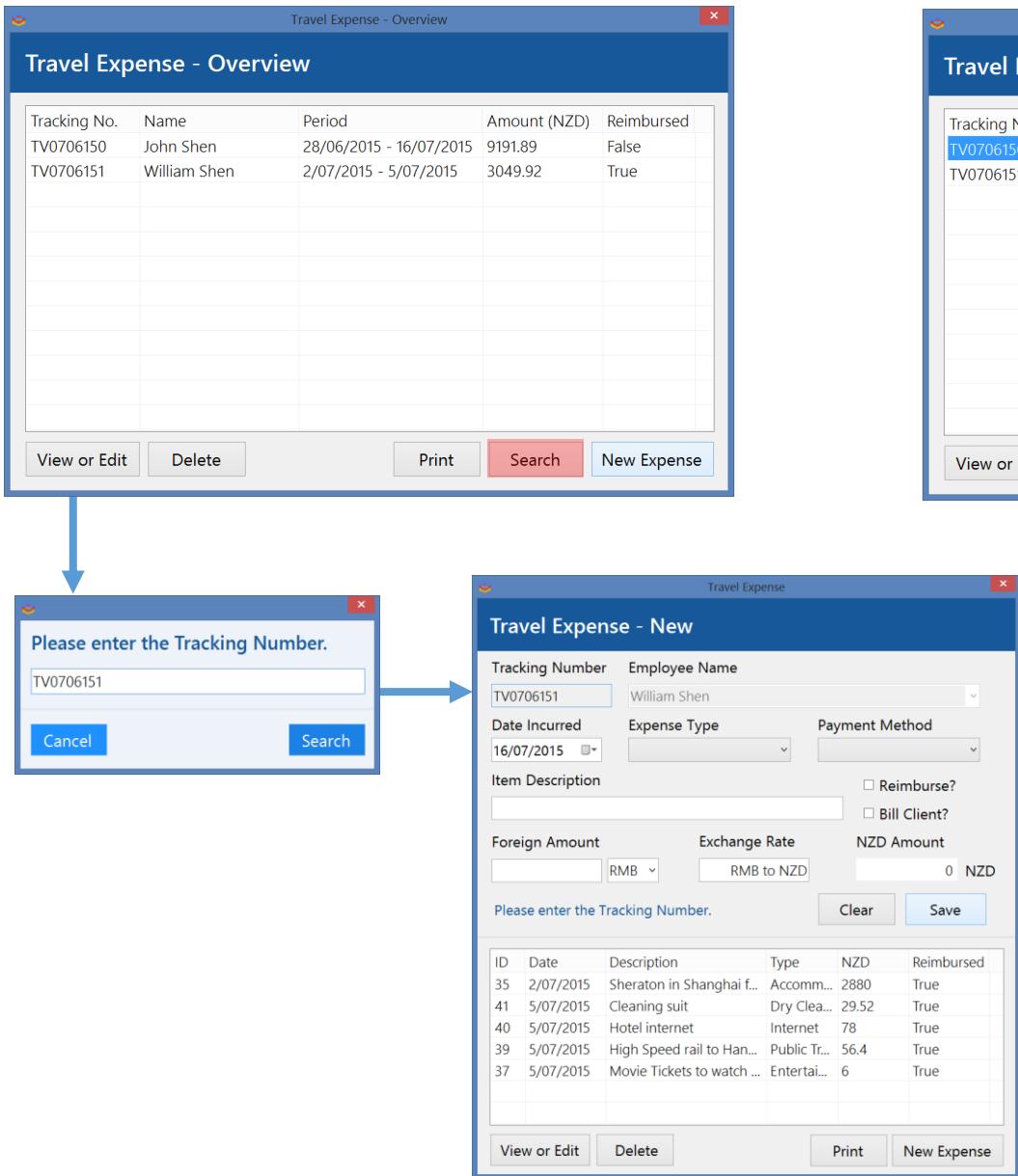


Fig 24.3

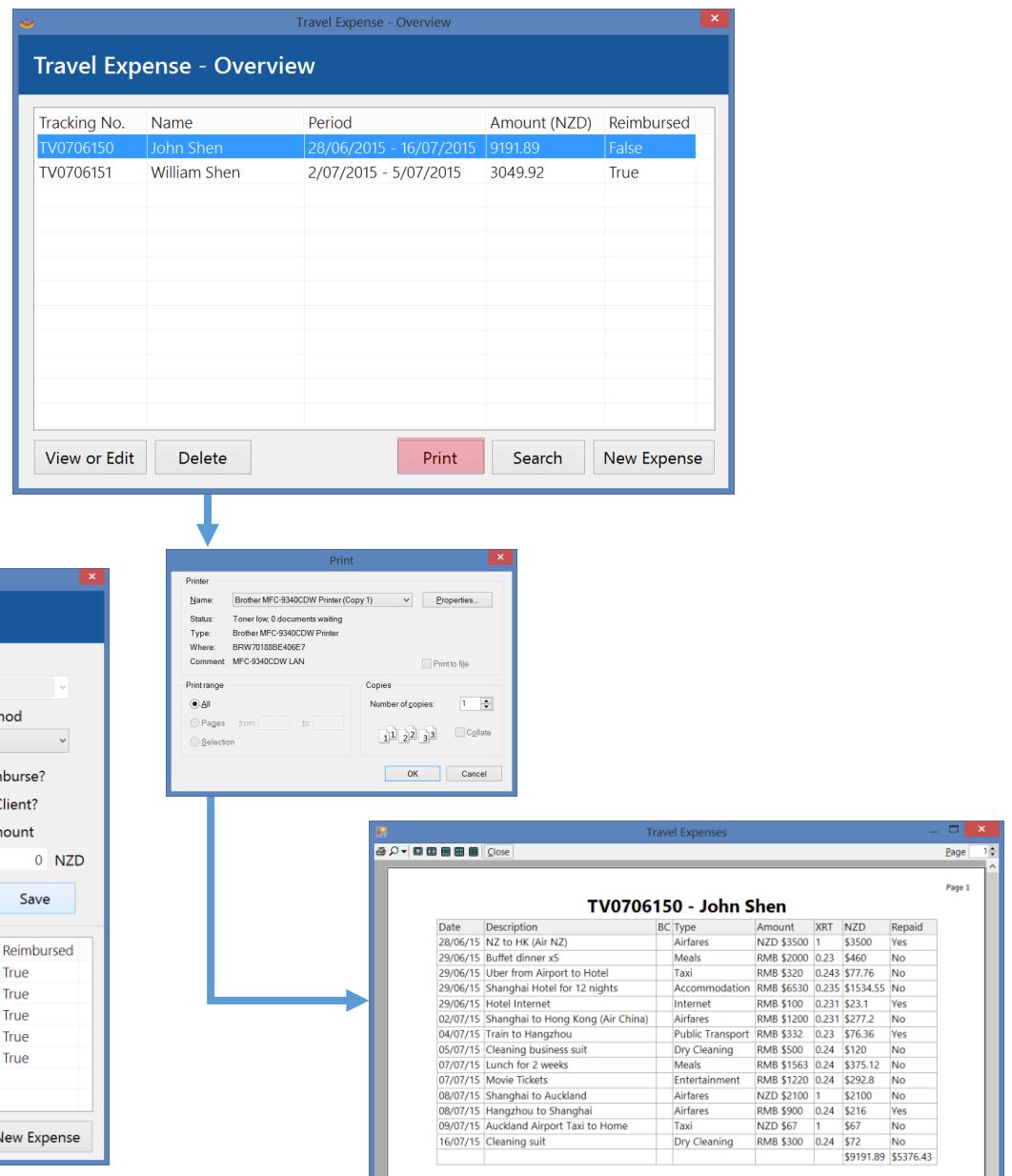


Fig 24.4

Travel Expense - Overview

Tracking No.	Name	Period	Amount (NZD)	Reimbursed
12345	Daniel Dyer	16/07/2015 - 16/07/2015	345345	True
TV0706150	John Shen	28/06/2015 - 16/07/2015	9191.89	False
TV0706151	William Shen	2/07/2015 - 5/07/2015	3049.92	True

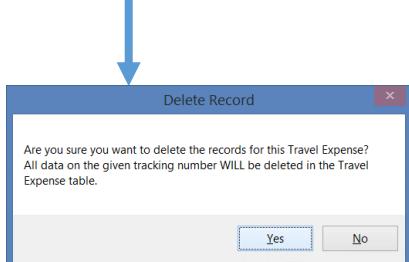
View or Edit Delete Print Search New Expense

Fig 24.5

Travel Expense - Overview

Tracking No.	Name	Period	Amount (NZD)	Reimbursed
TV0706150	John Shen	28/06/2015 - 16/07/2015	9191.89	False
TV0706151	William Shen	2/07/2015 - 5/07/2015	3049.92	True

View or Edit Delete Print Search New Expense



Travel Expense - Overview

Tracking No.	Name	Period	Amount (NZD)	Reimbursed
TV0706150	John Shen	28/06/2015 - 16/07/2015	9191.89	False
TV0706151	William Shen	2/07/2015 - 5/07/2015	3049.92	True

View or Edit Delete Records deleted. Print Search New Expense

Travel Expense - New

Tracking Number	Employee Name				
TV0706151	William Shen				
Date Incurred	Expense Type	Payment Method			
16/07/2015					
Item Description					
<input type="checkbox"/> Reimburse? <input type="checkbox"/> Bill Client?					
Foreign Amount	Exchange Rate	NZD Amount			
	RMB to NZD	0 NZD			
Please enter the Tracking Number. <input type="button" value="Clear"/> <input type="button" value="Save"/>					
ID	Date	Description	Type	NZD	Reimbursed
35	2/07/2015	Sheraton in Shanghai f...	Accomm...	2880	True
41	5/07/2015	Cleaning suit	Dry Clea...	29.52	True
40	5/07/2015	Hotel internet	Internet	78	True
39	5/07/2015	High Speed rail to Han...	Public Tr...	56.4	True
37	5/07/2015	Movie Tickets to watch ...	Entertai...	6	True

View or Edit Delete Print New Expense

Fig 25.1

Travel Expense - New

Tracking Number	Employee Name				
TV0706151	William Shen				
Date Incurred	Expense Type	Payment Method			
16/07/2015	<input type="button" value="▼"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>		
Item Description		<input type="checkbox"/> Reimburse? <input type="checkbox"/> Bill Client?			
Foreign Amount		Exchange Rate	NZD Amount		
<input type="button" value="RMB"/>		RMB to NZD	0 NZD		
Please enter the Tracking Number. <input type="button" value="Clear"/> <input type="button" value="Save"/>					
ID	Date	Description	Type	NZD	Reimbursed
35	2/07/2015	Sheraton in Shanghai f...	Accomm...	2880	True
41	5/07/2015	Cleaning suit	Dry Clea...	29.52	True
40	5/07/2015	Hotel internet	Internet	78	True
39	5/07/2015	High Speed rail to Han...	Public Tr...	56.4	True
37	5/07/2015	Movie Tickets to watch ...	Entertain...	6	True
<input type="button" value="View or Edit"/> <input type="button" value="Delete"/> <input type="button" value="Print"/> <input type="button" value="New Expense"/>					

Fig 25.2

Fig 25.3

Fig 25.4

Travel Expense

Travel Expense - New

Tracking Number	Employee Name				
TV0507150	Richard Hendrix				
Date Incurred	Expense Type	Payment Method			
16/07/2015	Internet	Personal			
Item Description	<input checked="" type="checkbox"/> Reimburse? <input type="checkbox"/> Bill Client?				
Hotel Internet Hotel Internet Hotel Internet Hotel Internet					
Foreign Amount	Exchange Rate	NZD Amount			
9999999999	USD	1564654598.4 NZD			
Please enter the Tracking Number.		<input type="button" value="Clear"/> <input type="button" value="Save"/>			
ID	Date	Description	Type	NZD	Reimbursed
54	16/07/2015	Auckland to San Franci...	Airfares	3213	True
View or Edit Delete Print New Expense					

1

© William Shen

Fig 25.5

The screenshot shows two instances of the 'Travel Expense - New' form side-by-side. Both forms have identical data: Tracking Number TV0507150, Employee Name Richard Hendrix, Date Incurred 16/07/2015, Expense Type Airfares, Payment Method Company, Item Description 'Auckland to San Francisco Return Flight', Reimburse? checked, Bill Client? unchecked, Foreign Amount 2100 USD, Exchange Rate 1.53, NZD Amount 3213 NZD. The second form has a red validation message at the bottom: 'Please ensure all fields contain valid data.' A blue arrow points from the first form to the second.

Fig 25.6

The screenshot shows the 'Travel Expense - New' form with the same data as Fig 25.5. The validation message 'Please ensure all fields contain valid data.' is still present. A blue arrow points down to the next screenshot, which shows the form after a save attempt.

Fig 25.7

The screenshot shows the 'Travel Expense - View and Edit' form. It displays the same data as the previous forms. A green success message at the bottom left says 'Record successfully loaded. Please make any desired changes and click 'Update''. A blue arrow points from the previous screenshot to this one. Below the main form is a table of expense records:

ID	Date	Description	Type	NZD	Reimbursed
55	16/07/2015	Hotel Internet Hotel Int...	Internet	1564654...	False
54	16/07/2015	Auckland to San Franci...	Airfares	3213	True

At the bottom are buttons: View or Edit, Delete, All Reimbursed, Print, and New Expense.

Fig 25.8

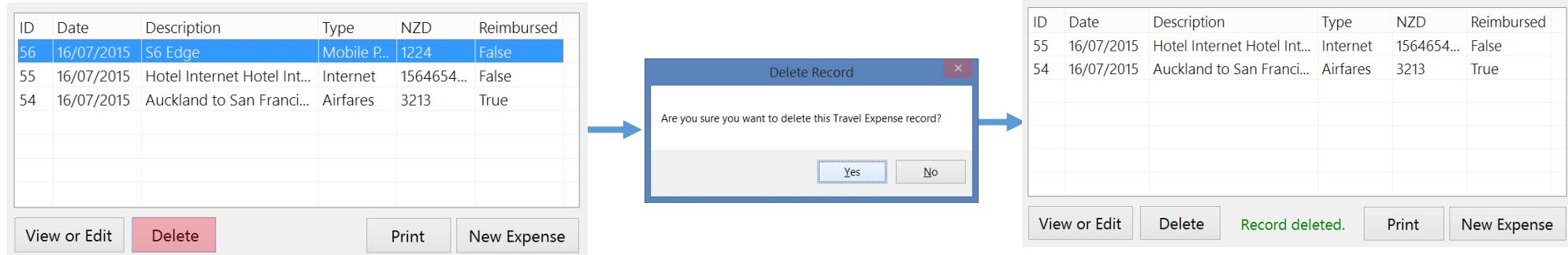


Fig 25.9

This screenshot shows a list of travel expense records. The second record (ID 55) is highlighted in blue. A blue arrow points from this record towards the "Delete" button at the bottom of the list.

ID	Date	Description	Type	NZD	Reimbursed
55	16/07/2015	Hotel Internet Hotel Int...	Internet	1564654...	False
54	16/07/2015	Auckland to San Franci...	Airfares	3213	True

Buttons at the bottom: View or Edit, Delete, Print, New Expense.

This screenshot shows a detailed view of a travel expense record. The tracking number TV0507150 and employee name Richard Hendrix are displayed. A blue arrow points from the "Delete" button at the bottom towards the "Delete Record" confirmation dialog in Fig 25.10.

Tracking Number	Employee Name
TV0507150	Richard Hendrix

Date Incurred	Expense Type	Payment Method
16/07/2015	Airfares	Company

Item Description	Reimburse?
Auckland to San Francisco Return Flight	<input type="checkbox"/>

Foreign Amount	Exchange Rate	NZD Amount
2100	USD	1.53
		3213 NZD

Message: Record successfully loaded. Please make any desired changes and click 'Update'.

Buttons at the bottom: Delete, Update.

Fig 25.10

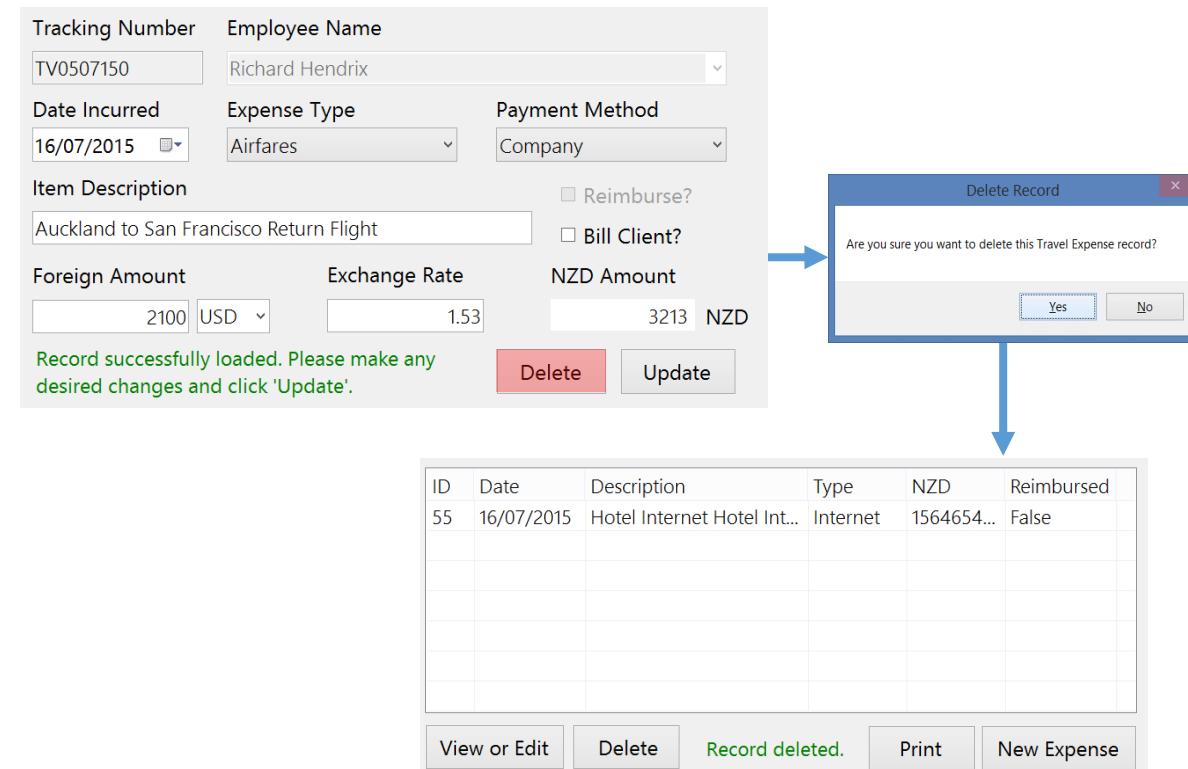


Fig 25.11

Travel Expense - View and Edit Reimbursed?

Tracking Number	Employee Name	
TV0507150	Richard Hendrix	
Date Incurred	Expense Type	Payment Method
16/07/2015	Meals	Company
Item Description		
Dinner at Restaurant		
<input type="checkbox"/> Reimburse?		<input type="checkbox"/> Bill Client?
Foreign Amount	Exchange Rate	NZD Amount
120	USD	1.53
183.6 NZD		
Record successfully loaded. Please make any desired changes and click 'Update'.		
<input type="button" value="Delete"/>		<input type="button" value="Update"/>

ID	Date	Description	Type	NZD	Reimbursed
57	16/07/2015	Dinner at Restaurant	Meals	183.6	True
55	16/07/2015	Hotel Internet Hotel Int...	Internet	1564654...	False

Fig 25.12

Travel Expense - View and Edit Reimbursed?

Tracking Number	Employee Name	
TV0507150	Richard Hendrix	
Date Incurred	Expense Type	Payment Method
16/07/2015	Internet	Personal
Item Description		
Hotel Internet Hotel Internet Hotel Internet Hotel Internet		
<input checked="" type="checkbox"/> Reimburse?		<input type="checkbox"/> Bill Client?
Foreign Amount	Exchange Rate	NZD Amount
9999999999	USD	222
22199999977 NZD		
Record successfully loaded. Please make any desired changes and click 'Update'.		
<input type="button" value="Delete"/>		<input type="button" value="Update"/>

ID	Date	Description	Type	NZD	Reimbursed
57	16/07/2015	Dinner at Restaurant	Meals	183.6	True
55	16/07/2015	Hotel Internet Hotel Int...	Internet	2219999...	False

Fig 25.14

ID	Date	Description	Type	NZD	Reimbursed
59	16/07/2015	Hotel	Accomm...	308	False
58	16/07/2015	Movie Tickets	Entertai...	30.8	False
57	16/07/2015	Dinner at Restaurant	Meals	183.6	True
55	16/07/2015	Hotel Internet Hotel Int...	Internet	2219999...	False

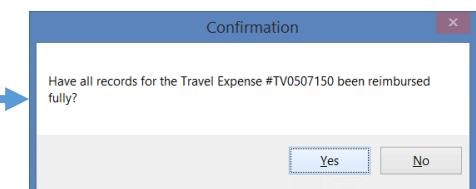


Fig 25.13

Travel Expense - View and Edit Reimbursed?

Tracking Number	Employee Name	
TV0507150	Richard Hendrix	
Date Incurred	Expense Type	Payment Method
16/07/2015	Meals	Company
Item Description		
<input type="checkbox"/> Reimburse?		
<input type="checkbox"/> Bill Client?		
Foreign Amount	Exchange Rate	NZD Amount
		0 NZD
Record successfully loaded. Please make any desired changes and click 'Update'.		
<input type="button" value="Delete"/>		<input type="button" value="Update"/>

Travel Expense - View and Edit Reimbursed?

Tracking Number	Employee Name	
TV0507150	Richard Hendrix	
Date Incurred	Expense Type	Payment Method
16/07/2015	Meals	Company
Item Description		
<input type="checkbox"/> Reimburse?		
<input type="checkbox"/> Bill Client?		
Foreign Amount	Exchange Rate	NZD Amount
		0 NZD
Please ensure all fields contain valid data.		
<input type="button" value="Delete"/>		<input type="button" value="Update"/>

ID	Date	Description	Type	NZD	Reimbursed
59	16/07/2015	Hotel	Accomm...	308	True
58	16/07/2015	Movie Tickets	Entertai...	30.8	True
57	16/07/2015	Dinner at Restaurant	Meals	183.6	True
55	16/07/2015	Hotel Internet Hotel Int...	Internet	221999...	True

Fig 25.15

ID	Date	Description	Type	NZD	Reimbursed
35	2/07/2015	Sheraton in Shanghai f...	Accomm...	2880	True
41	5/07/2015	Cleaning suit	Dry Clea...	29.52	True
40	5/07/2015	Hotel internet	Internet	78	True
39	5/07/2015	High Speed rail to Han...	Public Tr...	56.4	True
37	5/07/2015	Movie Tickets to watch ...	Entertain...	6	True

Print

New Expense

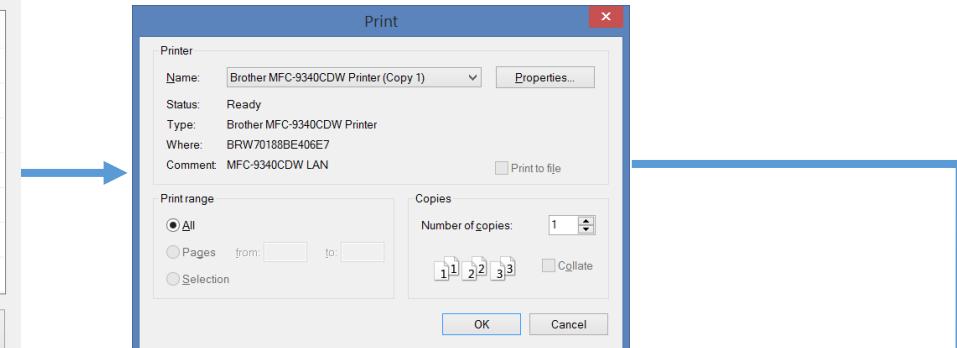


Fig 26.1

Wages - Overview

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	68999090
22	21/06/2015	21/06/2015	William Shen	21/06/2015	9877
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34
24	8/07/2015	9/07/2015	Daniel Dyer	6/07/2015	382421

Select a row from the tables above:

Filter by:

Payments Between and

Employee Name

Printing Options:

Print all Data
 Print Filtered Data

Travel Expenses

TV0706151 - William Shen

Date	Description	BC	Type	Amount	XRT	NZD	Repaid
02/07/15	Sheraton in Shanghai for 12 nights	Accommodation	RMB \$12000	0.24	\$2880	Yes	
05/07/15	Cleaning suit	Dry Cleaning	RMB \$123	0.24	\$29.52	Yes	
05/07/15	Hotel internet	Internet	RMB \$325	0.24	\$78	Yes	
05/07/15	High Speed rail to Hangzhou	Public Transport	RMB \$235	0.24	\$56.4	Yes	
05/07/15	Movie Tickets to watch Jurassic World	Entertainment	RMB \$25	0.24	\$6	Yes	
							\$3049.92
							\$0

Wages - New

Period for Payment

From To Date Paid

Employee Name

Gross Pay

PAYE NZD KiwiSaver NZD Net Pay NZD

Fill in the relevant details and press "Add".

Fig 26.2

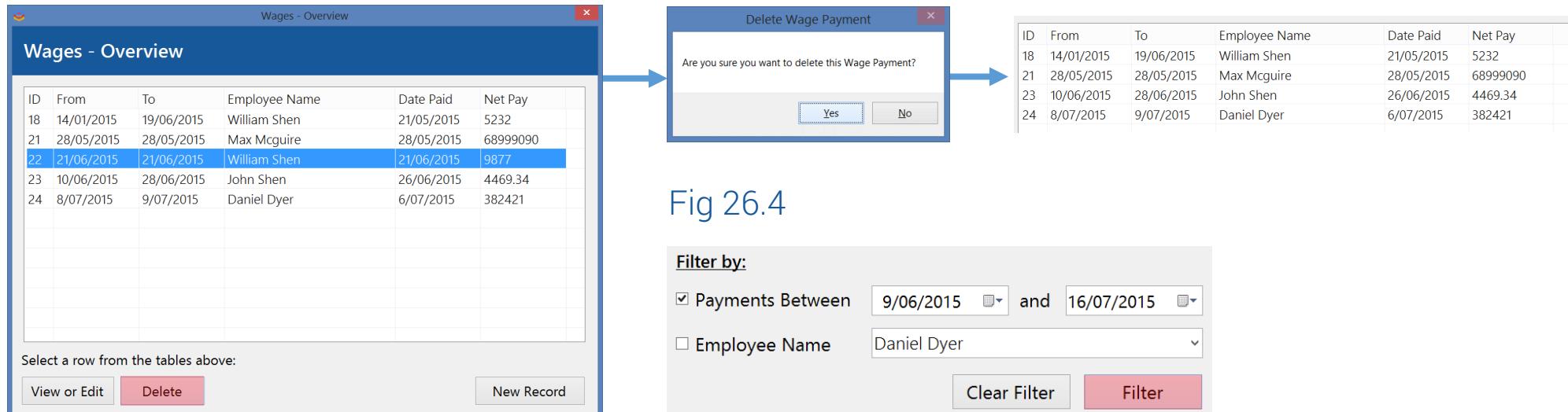


Fig 26.3

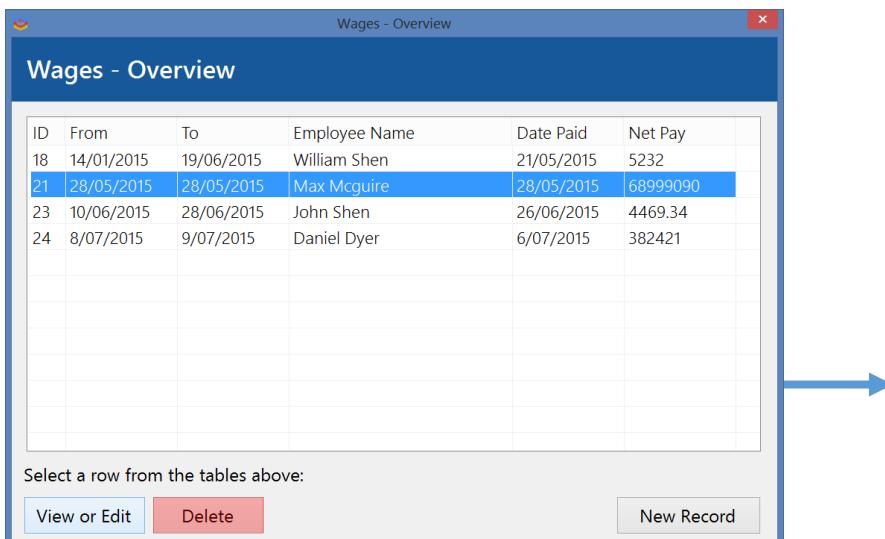


Fig 26.4

Filter by:

Payments Between and

Employee Name

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34
24	8/07/2015	9/07/2015	Daniel Dyer	6/07/2015	382421

Wages - View and Edit

Period for Payment Date Paid ID

From To Date Paid ID

Employee Name Gross Pay

PAYE KiwiSaver Net Pay

Press "Update" to update record.

Fig 26.5

Filter by:

Payments Between and

Employee Name

Data successfully filtered.

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	68999090
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34
24	8/07/2015	9/07/2015	Daniel Dyer	6/07/2015	382421

Fig 26.6

Printing Options:

Print all Data

Print Filtered Data

Print dialog box:

- Name: Brother MFC-9340CDW Printer (Copy 1)
- Status: Ready
- Type: Brother MFC-9340CDW Printer
- Where: BRW70188BE406E7
- Comment: MFC-9340CDW LAN
- Print range: All
- Copies: Number of copies: 1
- OK button

Wages - All table:

Dates	Name	Date Paid	Gross Pay	PAYE	KiwiSaver	Net Pay
14/01/15 - 19/06/15	William Shen	21/05/15	\$6000	\$5	\$763	\$5232
28/05/15 - 28/05/15	Max McGuire	28/05/15	\$69000000	\$455	\$455	\$68999090
10/06/15 - 28/06/15	John Shen	26/06/15	\$4535	\$65.33	\$0.33	\$4469.34
08/07/15 - 09/07/15	Daniel Dyer	06/07/15	\$562100	\$56223	\$123456	\$382421

Fig 26.7

Printing Options:

Print all Data

Print Filtered Data

Print dialog box:

- Name: Brother MFC-9340CDW Printer (Copy 1)
- Status: Ready
- Type: Brother MFC-9340CDW Printer
- Where: BRW70188BE406E7
- Comment: MFC-9340CDW LAN
- Print range: All
- Copies: Number of copies: 1
- OK button

Wages - Filtered table:

Dates	Name	Date Paid	Gross Pay	PAYE	KiwiSaver	Net Pay
14/01/15 - 19/06/15	William Shen	21/05/15	\$6000	\$5	\$763	\$5232
10/06/15 - 28/06/15	John Shen	26/06/15	\$4535	\$65.33	\$0.33	\$4469.34
08/07/15 - 09/07/15	Daniel Dyer	06/07/15	\$562100	\$56223	\$123456	\$382421

Fig 27.1

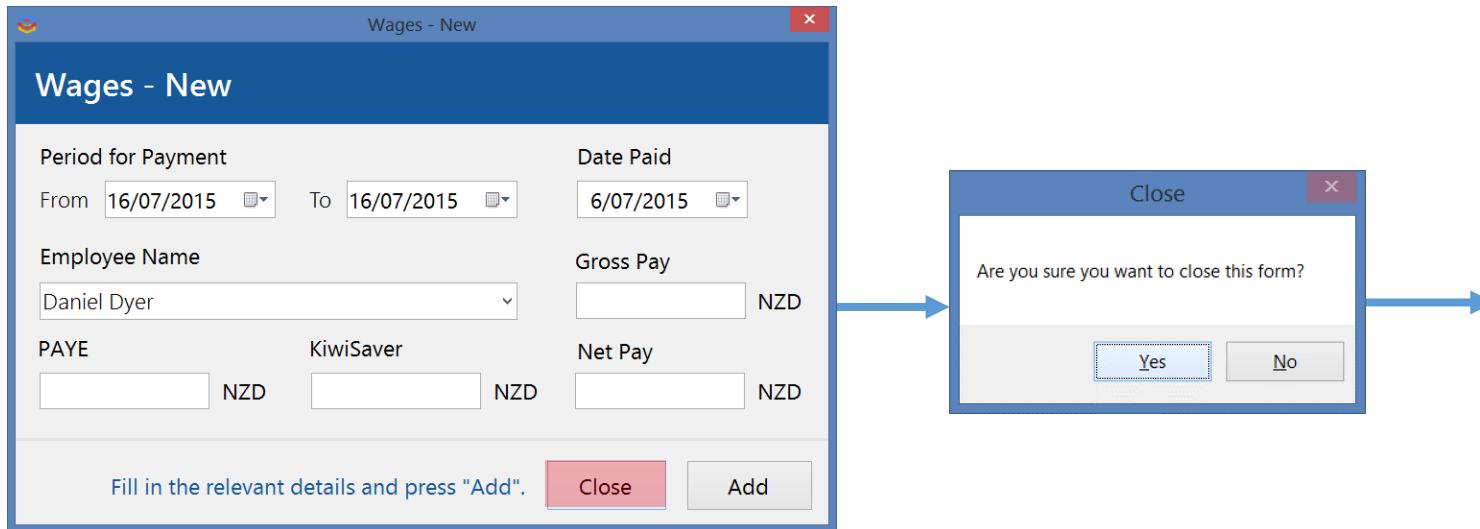


Fig 27.2

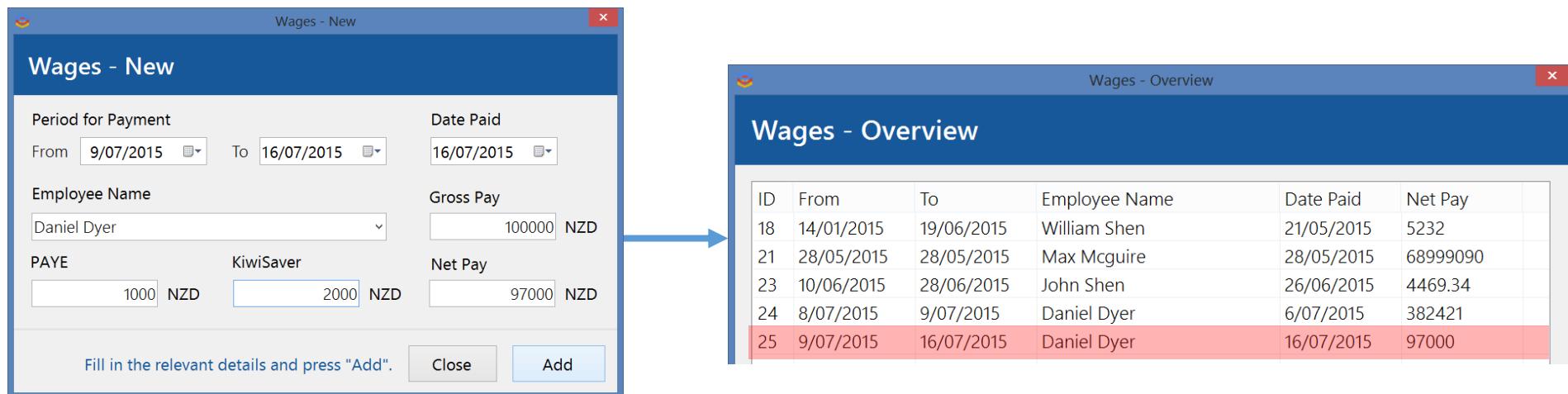


Fig 27.3

Wages - New

Period for Payment	Date Paid	
From 6/01/2015	To 12/06/2015	16/07/2015
Employee Name	Gross Pay	
William Shen	9999999999 NZD	
PAYE	Net Pay	
99999 NZD	99999 NZD	9999800001 NZD

Fill in the relevant details and press "Add".

Add

Wages - Overview

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	68999090
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34
24	8/07/2015	9/07/2015	Daniel Dyer	6/07/2015	382421
25	9/07/2015	16/07/2015	Daniel Dyer	16/07/2015	97000
26	6/01/2015	12/06/2015	William Shen	16/07/2015	9999800001

Fig 27.4

Wages - New

Period for Payment	Date Paid		
From 16/07/2015	To 16/07/2015	16/07/2015	
Employee Name	Gross Pay		
Daniel Dyer		NZD	
PAYE	KiwiSaver	Net Pay	
			NZD

Fill in the relevant details and press "Add".

Add

Wages - New

Period for Payment	Date Paid		
From 16/07/2015	To 16/07/2015	16/07/2015	
Employee Name	Gross Pay		
Daniel Dyer		NZD	
PAYE	KiwiSaver	Net Pay	
			NZD

Please ensure all fields are filled out correctly.

Add

Fig 27.5

The diagram illustrates the state transition of a wage payment record. It shows two screens side-by-side, connected by a blue arrow pointing from left to right.

Left Screen (Initial State):

- Employee Name:** Daniel Dyer
- Gross Pay:** 10000 NZD
- PAYE:** 1200 NZD
- KiwiSaver:** [Empty]
- Net Pay:** 8800 NZD

Right Screen (Final State):

- Employee Name:** Daniel Dyer
- Gross Pay:** 10000 NZD
- PAYE:** 1200 NZD
- KiwiSaver:** 365 NZD
- Net Pay:** 8435 NZD

Fig 28.1

Wages - View and Edit

Period for Payment: From 9/07/2015 To 16/07/2015 Date Paid 16/07/2015 ID 25

Employee Name: Daniel Dyer **Gross Pay:** 100000 NZD

PAYE: 1000 NZD **KiwiSaver:** 2000 NZD **Net Pay:** 97000 NZD

Press "Update" to update record. **Close** **Delete** **Update**

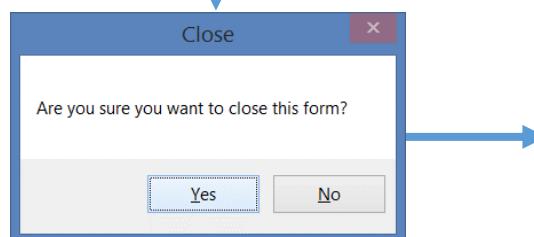


Fig 28.2

Wages - View and Edit

Period for Payment: From 9/07/2015 To 16/07/2015 Date Paid 16/07/2015 ID 25

Employee Name: Daniel Dyer **Gross Pay:** 100000 NZD

PAYE: 1000 NZD **KiwiSaver:** 2000 NZD **Net Pay:** 97000 NZD

Press "Update" to update record. **Close** **Delete** **Update**

Delete Wage Payment

Are you sure you want to delete this Wage Payment?

Yes **No**

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
21	28/05/2015	28/05/2015	Max Mcguire	28/05/2015	68999090
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34
24	8/07/2015	9/07/2015	Daniel Dyer	6/07/2015	382421
26	6/01/2015	12/06/2015	William Shen	16/07/2015	9999800001

Fig 28.3

Wages - View and Edit

Period for Payment	Date Paid	ID
From 8/07/2015	To 9/07/2015	9/07/2015 24
Employee Name		Gross Pay
Daniel Dyer		562100 NZD
PAYE	KiwiSaver	Net Pay
56223 NZD	12345 NZD	493532 NZD

Press "Update" to update record.

Buttons: Close, Delete, Update

Fig 28.4

Wages - View and Edit

Period for Payment	Date Paid	ID
From 6/01/2015	To 12/06/2015	16/07/2015 26
Employee Name		Gross Pay
William Shen		9999999999 NZD
PAYE	KiwiSaver	Net Pay
999999 NZD	999999 NZD	9998000001 NZD

Press "Update" to update record.

Buttons: Close, Delete, Update

Wages - Overview

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	68999090
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34
24	8/07/2015	9/07/2015	Daniel Dyer	9/07/2015	493532
26	6/01/2015	12/06/2015	William Shen	16/07/2015	9998000001

Select a row from the tables above:

Buttons: View or Edit, Delete, Record successfully updated, New Record

Wages - Overview

ID	From	To	Employee Name	Date Paid	Net Pay
18	14/01/2015	19/06/2015	William Shen	21/05/2015	5232
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	68999090
23	10/06/2015	28/06/2015	John Shen	26/06/2015	4469.34
24	8/07/2015	9/07/2015	Daniel Dyer	9/07/2015	493532
26	6/01/2015	12/06/2015	William Shen	16/07/2015	9998000001

Select a row from the tables above:

Buttons: View or Edit, Delete, Record successfully updated, New Record

Fig 28.5

The image shows two instances of a software window titled "Wages - View and Edit".

Left Window (Initial State):

- Period for Payment:** From 8/07/2015 To 9/07/2015 Date Paid 9/07/2015 ID 24
- Employee Name:** Daniel Dyer
- Gross Pay:** NZD
- PAYE:** NZD
- KiwiSaver:** NZD
- Net Pay:** 0 NZD
- Message:** Press "Update" to update record.
- Buttons:** Close, Delete, Update (highlighted in red)

Right Window (After Update):

- Period for Payment:** From 8/07/2015 To 9/07/2015 Date Paid 9/07/2015 ID 24
- Employee Name:** Daniel Dyer
- Gross Pay:** NZD
- PAYE:** NZD
- KiwiSaver:** NZD
- Net Pay:** 0 NZD
- Message:** Please ensure all fields are filled out correctly.
- Buttons:** Close, Delete, Update (highlighted in blue)

Fig 28.6

The image shows two instances of a software window titled "Wages - View and Edit".

Left Window (Initial State):

- Employee Name:** Daniel Dyer
- Gross Pay:** 562100 NZD
- PAYE:** 56223 NZD
- KiwiSaver:** NZD
- Net Pay:** 505877 NZD

Right Window (After Update):

- Employee Name:** Daniel Dyer
- Gross Pay:** 562100 NZD
- PAYE:** 56223 NZD
- KiwiSaver:** 1234 NZD
- Net Pay:** 504643 NZD

Video Evidence

Please refer to the links listed below:

Vid A0 - <https://youtu.be/TwtPTL2iaLY?t=0s>

Vid A6 - <https://youtu.be/TwtPTL2iaLY?t=36s>

Vid A2 - <https://youtu.be/TwtPTL2iaLY?t=1m6s>

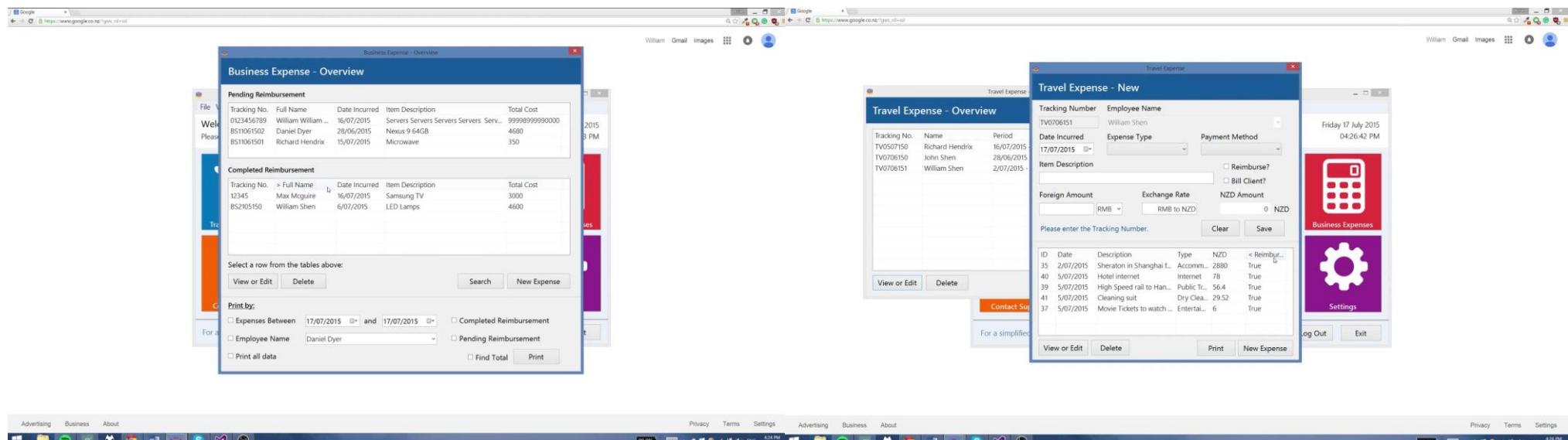
Vid A3 - <https://youtu.be/TwtPTL2iaLY?t=1m23s>

Vid A4 - <https://youtu.be/TwtPTL2iaLY?t=1m44s>

Vid A5 - <https://youtu.be/TwtPTL2iaLY?t=2m15s>

Vid A6 - <https://youtu.be/TwtPTL2iaLY?t=2m36s>

All the 'A' videos direct to the same video with different time frames.



Vid 1.0 - <https://www.youtube.com/watch?v=dOoA8HCyu4k>

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
0123456789	William Shen	16/07/2015	Servers	198000
BS1061501	Richard Hendrix	15/07/2015	Microwave	350
BS1061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680
BS987654	William Shen	17/07/2015	LED Lamps	5400

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
12345	Max McGuire	16/07/2015	Samsung TV	3000
BS123456	Daniel Dyer	14/07/2015	Ducky Keyboards	1080
BS2105150	William Shen	6/07/2015	LED Lamps	4600

Select a row from the tables above:

View or Edit | Delete | Search | New Expense

Print by:

Expenses Between 16/07/2015 and 17/07/2015 Completed Reimbursement
 Employee Name Daniel Dyer Pending Reimbursement
 Print all data Find Total | Print

Business Expense - Overview

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
0123456789	William Shen	16/07/2015	Servers	198000
BS1061501	Richard Hendrix	15/07/2015	Microwave	350
BS1061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680
BS987654	William Shen	17/07/2015	LED Lamps	5400

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
12345	Max McGuire	16/07/2015	Samsung TV	3000
BS123456	Daniel Dyer	14/07/2015	Ducky Keyboards	1080
BS2105150	William Shen	6/07/2015	LED Lamps	4600

Select a row from the tables above:

View or Edit | Delete | Search | New Expense

Print by:

Expenses Between 30/06/2015 and 24/07/2015 Completed Reimbursement
 Employee Name William Shen Pending Reimbursement
 Print all data Find Total | Print

Vid 1.1 - <https://www.youtube.com/watch?v=0qu2hvZfQ2U>

Pending Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
0123456789	William William	16/07/2015	Servers Servers Servers Serv. 9999999999999999	9999999999999999
BS1061501	Richard Hendrix	15/07/2015	Microwave	350
BS1061502	Daniel Dyer	28/06/2015	Nexus 9 64GB	4680

Completed Reimbursement

Tracking No.	Full Name	Date Incurred	Item Description	Total Cost
12345	William Shen	16/07/2015	Samsung TV	3000
BS2105150	William Shen	6/07/2015	LED Lamps	4600

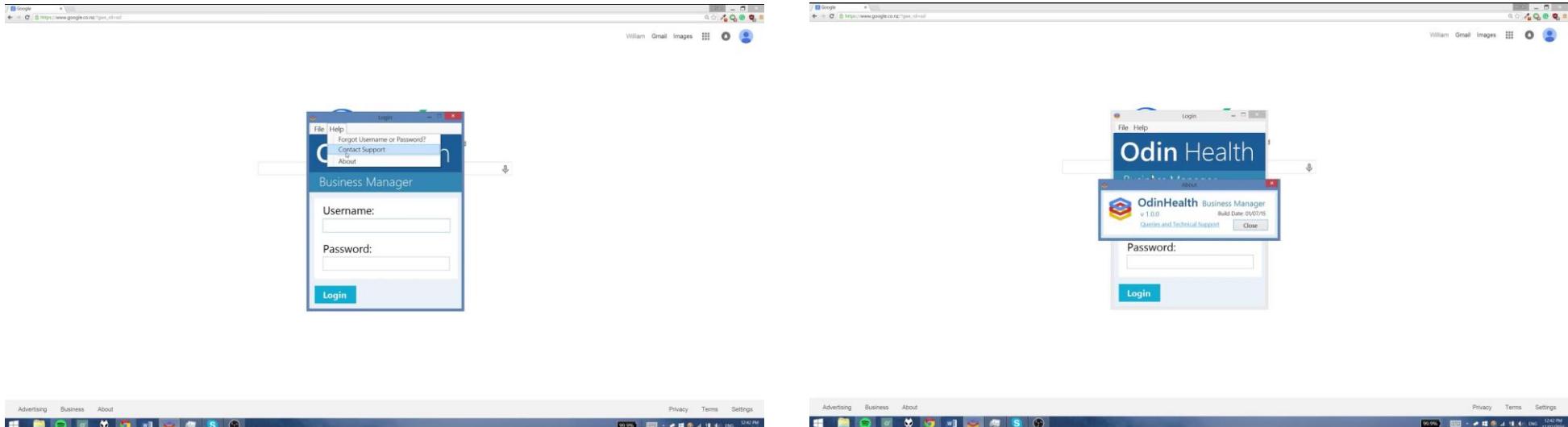
Select a row from the tables above:

View or Edit | Delete | Search | New Expense

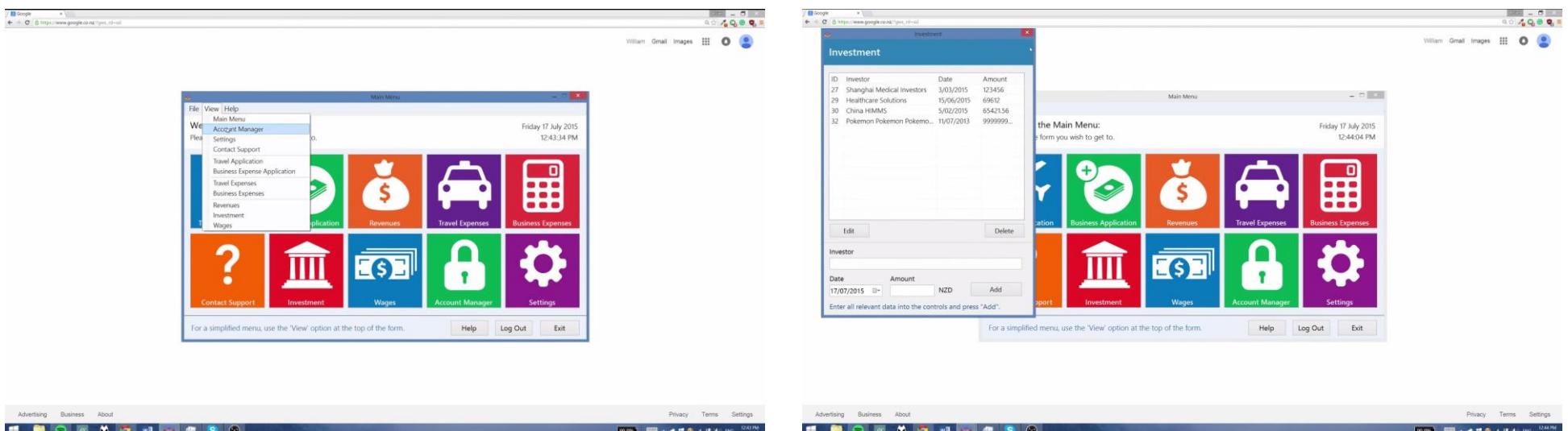
Print by:

Expenses Between 17/07/2015 and 17/07/2015 Completed Reimbursement
 Employee Name Daniel Dyer Pending Reimbursement
 Print all data Find Total | Print

Vid 2.1 - <https://www.youtube.com/watch?v=kuchisU5jc4>



Vid 3.1 - <https://www.youtube.com/watch?v=CIYNKGzJI7U>



Vid 4.1 - <https://www.youtube.com/watch?v=uwjOkZoPbFw>

ID	From	To	Employee Name	Date Paid	Net Pay
18	10/06/2015	19/06/2015	William Shen	10/06/2015	\$239990
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	68999000
23	10/06/2015	28/06/2015	John Smith	26/06/2015	446934
24	08/07/2015	09/07/2015	Daniel Dyer	09/07/2015	493532
26	09/07/2015	14/07/2015	William Shen	16/07/2015	999800001

ID	From	To	Employee Name	Date Paid	Net Pay
21	28/05/2015	28/05/2015	Max McGuire	28/05/2015	6899900



Installation

Implementation Plan

Discussing Types of Implementation

Here the different types of implementation are discussed. It should be noted that the current system is in the form of 'Excel' spreadsheets and thus there is a possible 'back-up' system to fall back to.

Method	Description	Advantages	Disadvantages
Phased	The new system is introduced in phases and gradually replaces parts of the old system.	Allows the users to gradually get used to the new system.	If the new system fails, data may be lost.
Parallel	The new system is started, but the old system is kept running in parallel.	If the new system fails, the old system acts as a back-up. Outputs from the systems can also be compared to ensure they are running correctly.	Entering data into two systems requires much more time and effort.
Direct	The old system is halted completely, and the new system is started immediately.	Takes minimal time and effort and the new system is up and running immediately.	If the new system fails, data may be lost.

Chosen Implementation

Keeping track of business expenses, travel expenses, revenues, wages and investments is an extremely important aspect of running a successful business. The current system that exists at Odin Health is very troublesome and inefficient, as both digital and physical documents may be misplaced easily. Thus, the chosen method is direct implementation.

To a business, time is money and by quickly switching over to the new system, efficiency levels will increase as will the data storage methods. Furthermore, the system has been extensively tested and with expected data and thus there is high confidence that the system will work without any major problems that could result in system failure.

If the new system were to catastrophically fail however, the client could easily fall-back and use the old system with its countless spreadsheets. Thus as shown, direct implementation would be the most cost effective and time saving implementation method for this particular system. Other implementation methods would only increase the time and workload of the users and are hence unsuitable.

Implementation Schedule

The client was consulted and shown the system as it was being developed to show him the progress that has been made. Here is an implementation schedule:

July 20th – The user will be introduced to the basics of the system and the installation procedure. Additionally, the user will be shown how to set the database location and relevant settings to ensure the system works to the user's requirements. The user will also be given the user guide and taught how to add accounts to login to the system with through the Account Manager.

July 22nd – The user will be shown how to use the Travel Expense Applications and Business Expense Applications aspect of the system. This includes adding, deleting, approving, modifying an application. Furthermore, the user will be given a detailed run through of how to use the Contact Support form, Settings form and the Account Manager. Finally, the user will be shown how to utilise the ListViews to sort the records.

July 24th – The user will be taught how to use the Revenues, Investment and Wages section of the system. This includes adding, deleting and modifying records and also printing records out for the Wages. Additionally, the user will be shown how to filter data through the system. Basic error messages and how to resolve them will also be covered. The system has been designed so that it is easy to navigate and use the forms as they all follow a similar structure (Overview, New, View and Edit).and contain similar controls (New, View and Edit, Search, Delete, etc.).

July 26th – The user will be taught how to utilise the Business Expense and Travel Expense functions of the system. The user will be shown how to add, delete and modify such expenses and furthermore be shown how to filter the data and print it. The expenses are one of the most important aspects of the system and will be discussed in depth.

July 27th – The user will be asked to perform both basic and advanced tasks with the system to ensure they know how to use it. The user will be told about back-up routines and how to restore data in case of a system failure. Furthermore, the user will be given the opportunity to ask any advanced questions before the system is

implemented. If there are no problems with the system, then it will be implemented on July 28th.

Training Details

The training will focus on using the actual program as a whole from July 22nd to July 26th. Throughout this period advanced tasks such as filtering data then printing it will be covered in depth. Additionally, basic tasks will be covered and the user will be asked to show that they understand how to use the program at these levels. The user will have the user guide available to read between training sessions to learn more about the system. Moreover, a technical guide will be provided to the user if more in-depth details are required.

Testing Details

The user will be required to complete the following tasks to ensure that the new system is able to perform tasks pragmatically to replace the old system:

Test	Check	Test	Check
Open the program		Delete a revenue record	
Changing the database location		Search for a revenue record	
Backup the database		Filter revenue records	
Change the email for applications		Add a travel expense	
Send an email to contact support		Edit a travel expense	
Add a new account		Delete a travel expense	
Edit an account		Search for travel expenses	
Delete an account		Print travel expenses	
Navigate to different forms using the Main Menu		Add a business expense	
Sort the ListViews		Edit a business expense	
Add a new travel application		Delete a business expense	
Edit a travel application		Search for a business expense	
Delete a travel application		Filter business expenses	
Search for a travel application		Print business expenses	
Approve a travel application		Add an investment record	
Add a new business expense application		Edit an investment record	

Test	Check	Test	Check
Edit a business expense application		Delete an investment record	
Delete a business expense application		Add a wage record	
Search for a business expense application		Edit a wage record	
Approve a business expense application		Delete a wage record	
Add a new revenue record		Filter wage records	
Edit a revenue record		Print wage records	

Changeover Details

Given that there are no problems with the system on the 27th of July, the system will be implemented the next day on the 28th in the morning to avoid any potential disruptions.

No extra hardware is to be installed but the Microsoft ACE OLEDB Provider needs to be installed so that the database may be accessed by the user. The program will be installed through an efficient and easy to use Setup Wizard.

Acceptance Statement

I, the client agree that direct implementation is the most suitable and pragmatic form of implementation to be carried out. Furthermore, I agree that the dates of the implementation schedule is feasible. Finally, I understand and accept the training, testing and changeover details.

Signed

Date

User Comments:

I have been shown the system and am satisfied with the functionality and efficiency of it. It fulfils my requirements and I am happy to test out and install the program to ensure that it is working properly.

Evaluation

Discussion of the degree of success in meeting the original objectives

The Odin Health Business Manager has been a major success. The scale of the project is quite large, and there are currently no known problems. Furthermore, the system successfully and efficiently replaces the previous spreadsheet solution and has thus allowed for the saving of time and money.

Now the client does not need to make a new spreadsheet for each expense, revenue, investment, etc. and can instead use the form design of the program to quickly and efficiently enter in data.

Advantages

- The use of a database is a major improvement over the old system of Excel spreadsheets. The database is a lot better at holding data.
- The program automatically produces reports easily which may be printed.
- The program allows the user to change the database location, application email and allows the user to quickly backup the database.
- The program is secure against any external users. All users must login with a username or password before using the system to its full capacity.
- Clear and consistent user interface design.
- Support of emailing applications and approving them.

Limitations

- Passwords are not encrypted. This may present a major risk if the system were to become widespread.
- The program may only be used in the local area network properly. However, it may be used with other databases outside this LAN. This could cause problems with the automatic tracking number assigner.
- No custom querying. Data can only be sorted or filtered using predefined fields in the program.

The following page fully discusses the objective agreed on in the design stage of the project and analyses each in detail.

Discussing Objectives

These are the objectives of the solution which were previously discussed and agreed on in the Requirements Specifications and Design Specifications. They will be discussed in depth here.

The solution must be able to:

Produce Printable Reports with the relevant data

The new system indeed allows the user to print reports with the data he or she wishes. Printing is supported on the business expenses, travel expenses and wage forms (i.e. the ones that require printing).

The solution allows the user to select the data mainly through filters and ListViews. For example on the business expenses printing function, there are sixteen possibilities the user could filter data through. Similarly, there are many filter options on the Wages form. On the Travel Expenses forms however, users can print data based on the tracking number, all data for one tracking number is printed.

The program also fulfils this objective through allowing the user to select the printer, how many copies he or she wants, and previewing the report before deciding to print it. This is extremely useful to ensure that all required data is present before printing.

Add, delete and update Usernames, Passwords and Full Names (Accounts)

The solution built for the client clearly meets this objective. Through a secured Account Manager, a user can choose to perform multiple functions including: adding an account, deleting an account, updating an account and viewing accounts.

These accounts secure the program from any external intruders. Despite the fact that passwords are not encrypted, the accounts system is still secure from most if not all potential intruders.

Provide a security barrier to prevent unauthorised access to the system.

The security barrier of the system is in essence the username and password assigned to each user. The login system prevents any external users from accessing the system. Furthermore, the program will exit itself if over 5 attempts have been exceeded.

Without a correct username and password, a user cannot access the intricate details of the system and is restricted to the Login form, Settings, Account Manager, Contact Support form and About form. By entering the correct

username and password, all other functions of the business manager are enabled.

Thus as shown, the solution clearly fulfils this objective.

Limit access to some sections of the system to selected Users.

The system indeed limits some important sections of the system to 'Admins'. These admins have permission to perform tasks such as change names of wage payments, approve business expense applications, approve travel applications, etc.

Accounts which do not have the admin permission have the relevant controls disabled and thus cannot use them. Hence the solution fulfils this objective.

Change the location of the database and back-up the database

It is extremely important that the user is able to change the location of the database so that he or she may access a local copy of the Odin Health database wherever they are. Furthermore, it is important to keep good back-up routines to ensure that the loss of data is almost impossible.

The solution achieves this requirement to a strong degree. Through the settings form, the user can change the location of the database easily through an open file dialog. Additionally, the user can back-up the database easily by choosing a directory and clicking 'OK'. The program also stores data on when the system was last backed up.

Allow users to send direct emails to the developer for Technical Support

Through the 'Contact Support' form, any user can send an email to the developer through the program asking any question or providing any comments. This form is also well designed and easy to use.

Thus, the Odin Health Business Manager fulfils this section of the objectives.

Allow users to apply for Travel or Business Expenses by filling in the relevant fields and sending a direct email to the CEO asking for approval.

Instead of sending a direct email to the CEO, the solution allows the user to define an email to send applications to. This is extremely helpful if someone else takes charge of handling these applications.

The solution indeed accomplishes this objective by allowing users to fill in a form based interface and then send it off by email to the relevant approver. All data from the form is sent with the email and therefore gives the approver a detailed view of each application.

Allow the CEO to approve user applications through the new system and send relevant emails if the application is approved.

The solution allows any user has been tagged as an 'Admin' to approve travel and business expense applications through their relevant 'View and Edit' forms. If the form was opened with an unchecked 'Approved' state and then is updated with a checked 'Approved' state then the program will send an email to the address of the applicant which he or she entered when making the application. Thus as shown, the solution accomplishes this objective previously agreed upon.

Add, delete and search these applications for expenses and automatically assign tracking numbers.

The system indeed allows users to add, delete, search and even update business expense and travel applications. Through the Overview, View and Edit, and New structure of the system.

Furthermore, through the use of complex algorithms and settings variables, the system can efficiently assign tracking numbers ensuring to clash unless different databases are constantly used. Hence the solution achieves this goal.

Allow users to add, modify and delete travel and business expenses.

Through the Overview, View and Edit, and New structure of the system as discussed above, the user can familiarly add, update and delete their travel and business expenses.

Moreover, the user can print these expenses in a useful and aesthetically pleasing report. Therefore, the solution efficiently executes this objective.

Add, edit and delete revenues and wages and all their relevant fields.

The user can definitely add, edit and delete revenues, wages and investments. This is done through the Overview, View and Edit and New structure of the system. Data may be added and updated using the cleverly designed form-based interface. Thus as shown, the solution fulfils this requirement.

Display all this data in a table-based view.

By using a specially designed algorithm, the solution allows data to be displayed in a table form on all 'Overview' forms, where ListViews are present in order to display this data. Due to the multitude of the fields, only the most important ones have been displayed on these ListViews.

Furthermore, the data in the ListViews may be sorted according to their columns. The ListViews are a prime example of how the solution achieves the objectives of the project.

Allow the entry, addition and deletion of data through a form-based interface.

As discussed in detail above, the majority of forms utilise a form-based interface to allow data to be added, updated and deleted. A form is a much easier way of entering data into the system than say through a data grid view.

Thus as shown, the solution evidently accomplishes this objective.

Sort data based on specific fields.

The solution indeed fulfils this requirement. As discussed in 'Display all this data in a table-based view', users can sort ListViews based on their columns by simply clicking on them. This data can be conveniently sorted in both ascending and descending order.

Query data based on specific fields.

Most functions of the system contain a search function where data may be searched based on PO Number, Tracking Number, etc. In addition to this, some fields contain filters by which users can filter, therefore query, data.

Although the fields which the user can query are limited, the solution nevertheless fulfils the requirement. Throughout the development stage, the user was consulted regarding what fields would be needed to be filtered. Hence the system fulfils this objective.

Evaluation of the client's and user's response to the system

The following is the client's comments on the requirements specification and how much the system satisfies them. The client has already been using the software for an effective amount of time.

General Requirements

1. The system must be efficient and quick to use.

The system does decrease the time required for me to carry out tasks that would have taken lots of time using the old spreadsheet system. However, sometimes there is lag which the developer has told me is due to constantly connecting to the database and such. The system also follows a similar structure throughout lots of the forms so it's easier to use for me.

2. Data should be stored and organised quickly and practically.

I have not seen the actual database itself but have interacted with it through the program. The solution does indeed view the data I need to see and I can also filter this data and print it out if I want. Also, I can search for records by their tracking number and sort the data in the tables.

3. The user interface must be unambiguous and easy to navigate and use.

Apart from adding the new travel expenses form, which is a bit more complex than the other forms, the system is easy to use and I can quickly go to the functions I want to use. Lots of the forms contain buttons in the same area so I can quickly find the things I need.

4. Reports must be able to be printed.

All the important data which I told the developer needs to be printed can be printed. These are the travel and business expenses and the wage payments. I can filter the data I want to print and then select to print the report in an easy to read table form. The table in fact looks better than the old spreadsheets.

5. Back-ups of the database must be easy to perform.

The developer showed me how to back up the system through the settings of the program. I can now easily back up the system without going through my computer to find the files. The system even shows me when I last backed up the database!

Hardware and Software Requirements

The new system must be able to be used on any computer running Windows 7, 8 or 8.1. A computer with a mouse, keyboard and printer would be necessary to utilise the system to its full potential.

The main database must be stored on the network so all employees can access it. Furthermore, an option to store the database locally (i.e. save it to a different directory), so that data may be accessed when the network cannot be accessed, is essential.

The new system must also be able to print reports to any printer that the client wishes. The required information must be able to be easily viewed and sorted for this printing.

The new system will be built on Microsoft Visual Basic .NET while using Microsoft Office Access 2013 to manage the database. This would require all Users computers to have the .NET Framework installed and relevant Microsoft Access Drivers or Microsoft Access 2013 to enable OLE DB (Object Linking and Embedding, Database) database connections.

Client Comments: I am able to use the Odin Health Business Manager on my Asus laptop with Windows 8.1. The database is not currently stored on the network but I have been told that I can easily copy it there and change the database location and then start using that database.

I can also print the information I want through a print setting window where I can select the printer I want to print to and how many copies. Next, the system will show me a preview of what the report will look like.

In regards to the installation to the system, I only needed to install the Access database software and run a setup wizard to install the program so it was really easy and quick to do.

Objectives of the Solution

Client comments are listed below. As shown, all objectives have been met.

Produce Printable Reports with the relevant data

All the important data which I told the developer needs to be printed can be printed. I can filter the data I want to print and then select to print the report in an easy to read table form.

Add, delete and update Usernames, Passwords and Full Names (Accounts)

I can use the account manager to add accounts to login with. There is an extra password to access the account manager so that is extra secure.

Provide a security barrier to prevent unauthorised access to the system.

If the user does not login to the system then he can't access the other functions of it. The user has to login to the system with a username and password

Limit access to some sections of the system to selected Users.

As an admin, all controls are enabled to me but for a normal user, they cannot access some things such as approving travel applications.

Change the location of the database and back-up the database

Through the settings menu I can change the location of the database and back it up easily.

Allow users to send direct emails to the developer for Technical Support

The contact support form allows me to send a message to the developer passing on any message. However once I do click send, the message is quite slow to send. I have to wait for up to 10 seconds.

Allow users to apply for Travel or Business Expenses by filling in the relevant fields and sending a direct email to the CEO asking for approval.

Users can send applications to me for their expenses. I can then approve them and they receive an email back. This is a lot easier than constantly exchanging emails between me and the employee and saves a lot of workload.

Allow the CEO to approve user applications through the new system and send relevant emails if the application is approved.

As I mentioned above I can approve the applications and an email will be sent to the person who submitted the application.

Add, delete and search these applications for expenses and automatically assign tracking numbers.

The program does automatically assign tracking number and I can easily delete and search these expenses through the overview forms.

Allow users to add, modify and delete travel and business expenses.

I can easily carry out these with the overview, new and view and edit forms.

Add, edit and delete revenues and wages and all their relevant fields.

I can easily carry out these with the overview, new and view and edit forms.

Display all this data in a table-based view.

All the data has been put in a table when I open it and I can then sort the rows or filter the data as I like.

Allow the entry, addition and deletion of data through a form-based interface.

All entry and updating of data is done through a form-based interface. This reduces any error for human input.

Sort data based on specific fields.

I can sort the tables on the forms by clicking the columns. I can sort in both ascending and descending order.

Query data based on specific fields.

I can search for expenses or revenues through their tracking number and PO Number. Although I can't filter data on all forms, there are advanced filtering on forms like the business expenses.

Current System Flaws

- If the size of a field is extensively long and then it is printed, the report may span onto two pages rather than one.
- Applications will not be resent if there is a failure on the first sending. The user must resubmit the application.

User Comments

As a user of this system, I must say it is very easy to navigate and use. Though I am concerned that the passwords are not encrypted, given the timescale of the project I assume it was not possible to include encryption. I can now easily apply for travel and business expenses and then have Kelly approve them and have an email sent to me.

I can also easily add new travel expenses after I come home after a trip in a form-based data entry form and then print the expenses after I've completed. This is much easier than making a new spreadsheet each time after a business trip.

Personally, I believe the printing function of the system works fantastically. The reports it produces are clear, concise and look quite good. I can also choose which printer I want to print to which is a bonus.

Overall, I think the system is a huge improvement over the almost non-existent old one. The new solution is easy to navigate around with its consistent screen layouts and also very efficient to use. Many bad business practices have now been removed.

Client Comments

I am overall very satisfied with the solution William has developed for me. The new system saves a lot of time and money and makes it easy to locate where important data has been stored. Now all the data is stored on one database instead of many spreadsheets. I can also easily back up this database and change its location through the settings.

The new solution also brings different things together into one program where employees can apply for travel and business expenses, add these expenses, add investments and revenues and add wage payments. The old system required separate spreadsheets for each of these things.

The design of the system is quite user friendly. The main menu has lots of symbols which represent their relevant form and most forms also follow a similar structure and have their buttons placed in relatively the same places.

However, there are aspects of the system which could be improved. Firstly, if applications are submitted with no internet connection then an email is not sent. The program should send an email once it is reconnected to the internet but currently does not do that. Another problem is that sometimes if a textbox is too long then when I print it, the table may span across two pages. However, given the short time scale of the project, the developer said there was not enough time to solve these problems and that they will be included in the next update.

In conclusion, I am very satisfied with the new business manager despite the problems listed above. The new system not only improves efficiency but also saves the company a lot of time and money. The system fully meets the requirements specification as agreed on earlier (I have also commented on them).

Final Developer Comments

Developing the Odin Health Business Manager was no easy tasks. As mentioned earlier in this project report, several difficult problems were encountered which took a substantial amount of time to sort out and perfect. I personally believe the project to be a major success.

Throughout the testing stage I constantly used the program to test out for any problems and thankfully there are none that I am aware of at the moment. By using the program I observed that the solution is easy to navigate and has large text for ease of reading. Furthermore, the forms were designed to have consistent layouts, once again taking into account the user design.

The programming stage of the project was extensive. Much of the validation was programmed to occur directly on the form by restricting user input into the controls. Furthermore, when the text was changed in some textboxes, automatic values would be calculated.

Discussing with the client, she is very happy with the overall solution as it saves her and the company a lot of time and money. Users have commented on the printing function of the system and how the layout is very consistent across the system.

In conclusion, the Odin Health Business Manager may now be considered a 'completed' solution to a high standard. As with most programs, maintenance will be performed to improve speed and performance over time and perhaps to add new features if required.

User Acceptance Document

As the end user of this system designed and developed by William Shen, I agree that the system works to my satisfaction and completes all the objectives that were agreed upon in the requirements specification.

In addition to this, the system:

- Exceeds my expectations
- Fulfils the design philosophy
- Is easy to use and navigate around
- Provides features over and above those requested
- Has no significant problems

Signed

Date