Cash Register Terminal

Case: Art Gallery Cash Terminal

Peter Shodeinde

Bachelor's Thesis

___.___.______ ________________________________

Valitse kohde.

SAVONIA UNIVERSITY OF APPLIED SCIENCES **THESIS**
Abstract

Field of Study
Technology, Communication and Transport

Degree Programme
Degree Programme in Information Technology

Author(s)
Peter Shodeinde

Title of Thesis
Cash Register Terminal, Case: Art Gallery Cash Terminal

Date	12 September 2012	Pages/Appendices	34 + 6 appendices

Supervisor(s)
Mr. Keijo Kuosmanen, Lecturer

Client Organisation/Partners
Galleria Carree Oy

Abstract

As a small company grows, there is a need for data to be stored in a centralized location. An efficient database that can be scaled to store larger information in the future is needed. The purpose of this project was to create a cash register database that can store art gallery information.

The task was to develop a desktop application for keeping records up-to- date. It uses Microsoft Access as the database, Access GUI and Visual Basic .Net programming for developing and implementing. The Access GUI was created using the Access forms with SQL data because of its user friendliness. Visual Basic .Net was used to create the interaction in the back-end side of the database. Information includes Bills and Receipt for each artworks, art work information, artist information and creating sales report for a certain period for items sold.

The thesis was done by using Microsoft technologies, gathering information from various online resources and textbook, analyzing the structure, and solving programmatic problems to make way for a better application.

The result of the thesis was a working cash register terminal that was tested by the client.

Keywords
Access Database, SQL, .Net Programming, Design, Implementation

Koulutusala
Technology, Communication and Transport

Koulutusohjelma
Degree Programme in Information Technology

Tekijä
Peter Shodeinde

Työn nimi
Cash Register Terminal, Case: Art Gallery Cash Terminal

Päiväys	10th September, 2012	Sivumäärä	34 + 6

Työn valvoja
Keijo Kuosmanen, Lecturer

Yritys
Galleria Carree Oy

Abstract

Yrityksen kasvaessa on tarpeen kiinnittää huomiota tietojen tallentamiseen keskitetysti. Ratkaisuna voi olla tietokanta, jossa huomioidaan myös tulevaisuuden kasvutarpeet. Tässä opinnäytetyössä luotiin sovellus ja tietokanta taidegallerian taidevälitystietojen tallentamiseen.

Tehtävänä oli kehittää työpöytäsovellus jolla voi syöttää ja seurata myynnissä olevien taideteosten myyntitietoja. Sovellus toteutettiin Microsoft Access-tietokannan päälle Visual Basic .Net-ohjelmointikielellä. Käyttöliittymä toteutettiin Accessin lomakkeilla. Tietokantaan talletetaan taidevälitykseen liittyen laskutustietoja, taideteosten tietoja, taiteilijoiden perustiedot ja tietyillä aikaväleillä toteutuneet myynnit.

Opinnäytetyössä sovellettiin Microsoftin teknologioita, kerättiin tietoa eri lähteistä, analysoitiin tiedon rakenne ja suunniteltiin sovellus sekä tietokanta.

Keywords
Access Database, SQL, .Net Programming, Design, Implementation

ABBREVIATIONS

SQL	Structured Query Language
PK	Primary Key
FK	Foreign Key
UI	User Interface
IDE	Integrated Development Environment
RDBMS	Relational Database Management System
DBMS	Database Management System
GPL	General Public License

Acknowledgements

I would like to thank God for guiding me from the start of this project. I would like to express my great appreciation to every individual who guided me throughout this project and provided me with the necessary background and the completion of this project.

I would like to show my appreciation to my supervisor, Mr Keijo Kuosmanen, for this guidance and support towards the completion of the project. Without his advice, this project would not have been completed.

My deepest gratitude goes to the head of my department, Mr. Arto Toppinen, for his support and for the chance to do this project.

Finally, I want to express my appreciation to my parents for bringing me up in the way of the Lord since I was young and giving me the chance to study in this school. I appreciate my friends and family in Nigeria and in other parts of the world for their support.

TABLE OF CONTENTS

1 INTRODUCTION

When the amount of data grows in a company, there is a need for a database system to store the data regularly. The need for an art gallery database includes storing data information, sales information for monthly reports.

This thesis presents the development of data management systems for holding cash information like the products sold, monthly flow of cash, and balance sheet information for the month. All this will be explained in detail in the following chapters. The objective is to design a data management system that can store information like art work, and sales information. In developing the database system, there will be a need to decide what the best tools are for this. The client selected Microsoft Access and Visual Basic .Net, since they were familiar tools she works with. In order to achieve these goals, important questions that are asked include what is the information that will be stored in the database, how many tables will be needed and how will these tables relate to each other. These are some good points to note when designing a cash register.

The Microsoft Access is where the work begins, it includes creating the models for each table in the database, analysing the content of the tables, designing the User Interface that the client will use to input data information into the database. The Visual Basic .Net will be used for creating dynamic interaction in the database, like generating the bank reference number for each customer that buys artwork, and wants make the payment to the art gallery's bank account.

2 INTEGRATED DEVELOPMENT ENVIRONMENTS

Before the development and implementation of the Cash Register Terminal, it is important to note the variety of technologies and tools available for this purpose. This chapter introduces these technologies and tools that can be used to create the Integrated Development Environment for the cash register terminal.

2.1 Database Management System

The Database Management System (DBMS) is a system that makes developers or users to create and maintain the database as well as to provide access to it. The next section, will talk about the SQL and Embedded SQL.

2.1.1 Structural Query Language (SQL)

Structural Query Language is a tool for performing operations in database systems. These operations include inserting new data into the database, retrieving data from the database, modifying of data stored in the database, and deleting of data from the database. It is a part of a database management system which provides communication to it. Figure 1 shows how the SQL is used to interact with the database management system. (Groff and Weinberg 1999, 9)

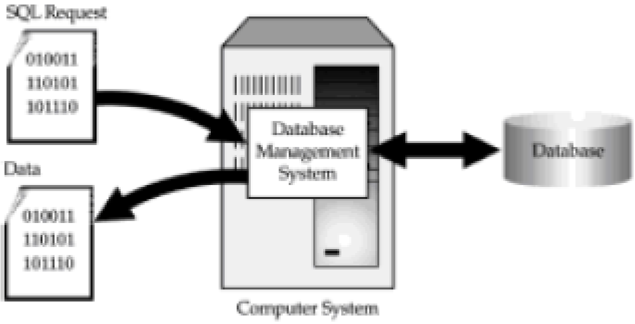


Figure 1 Data access using SQL.

2.1.2 Embedded Structural Query Language (Embedded SQL)

Embedded Structural Query Language is a language used for developing application programs. These provides an interface of embedding SQL into programming languages like Java, C, C++, and Ruby which allows for the interactive with database objects when developing an application using these programming. (Oracle8 2010, 93)

These are mechanism that needs to be followed for this to be achieved. Host variables are the variables used for making the initial communication between the database and an application developed in any programming language. An example of the host variables are as follows: (Oracle8 2010, 93)

```
EXEC SQL begin declare section;
    int cno;
    varchar cname[31];
    varchar cname[31];
    int zip;
   char phone[13];
EXEC SQL end declare section;
```

2.2 Advantages of Database Management Systems

The following advantages provide what a database can do. (Connolly and Begg 1999, 26-27)

Data Redundancy: It controls the way data is stored in the database. Reducing redundancy is accomplished by creating a relationship between table objects that stores the data.

Consistency: This is accomplished when a data in the database is updated. This is done once the database has reduced redundancy from the database. This makes the update information to appear immediately in the database.

Data Sharing: Data in the database belong to the organisation which can be shared between users of the database. When more data are added to the database, the only thing the database administrator has to do is give access to the users.

Data integrities: These are rules known as constraints that the database is not permitted to ignore. Before data are stored, these constraints are checked if available. An example is that a password data cannot be less than six digits.

Improved security: This is a way of protecting and preventing access to the database from unauthorized users. These users include people that do not have certain privileges to access a particular file or data in the database.

2.3 Disadvantages

The following provides the disadvantages of using a database. (Connolly and Begg 1999, 29-30)

Increase in complexity: As the database grows, there is an increase in the complexity that makes for a good DBMS. Making the slightest mistake in the designing decisions can lead to consequences for the organisation.

Cost of a DBMS: The cost of buying and maintaining a DBMS varies significantly making it difficult for some organisations to make some tough decisions.

Size: Increase in data stored in a database increase the complexity and functionality of a DBMS. These increases include memory size. Figure 2 shows the database processing.

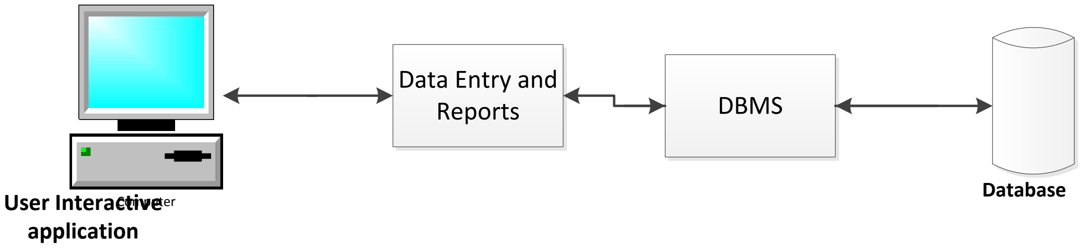


FIGURE 2. Database processing

2.4 Kinds of Database Management Systems

There are two common categories of Database Management Systems namely Commercial Database Management System and Open Source Management Systems. Table 1 shows the differences between them.

Table 1 Difference between Commercial and Open Source

Commercial Database Management System	**OpenSource Database Management System**
These are developed by organizations, and provided to the generally public at a certain price.	This means that the database can be developed and used by the everyone
The software code is hiding to the people using it, therefore it can only be improve by the vendor.	The software code is available to developers to improve it.
Example of this database systems are Microsoft SQL server, Oracle, Microsoft Access and IBM DB2.	Examples of this database systems are MySQL, NoSQL and PostgreSQL
This is expensive to purchase or maintain	It is cheap and easy to maintain it given the fact that the software code is available for manipulation to suit personal purposes

2.4.1 Microsoft Access

Microsoft Access is a multifunctional program which has sophisticated capabilities such as page layout, and formatted printing. It is a DBMS that combines the relational tables with a GUI and software-development tools. This is the platform that is used for this project. Table 2 shows some functionality of the database and figure 3 shows the datasheet view of the database.

TABLE 2. The Access Functionality

Data Organisation	Access creates a good format and views for creating and manipulating tables with data. E.g. the database view
Table joining and data extraction	This connects queries to tables using data relationships thereby creating temporary tables called record sets.
Data presentation	These involves creating, viewing and print report from a set of tables' data

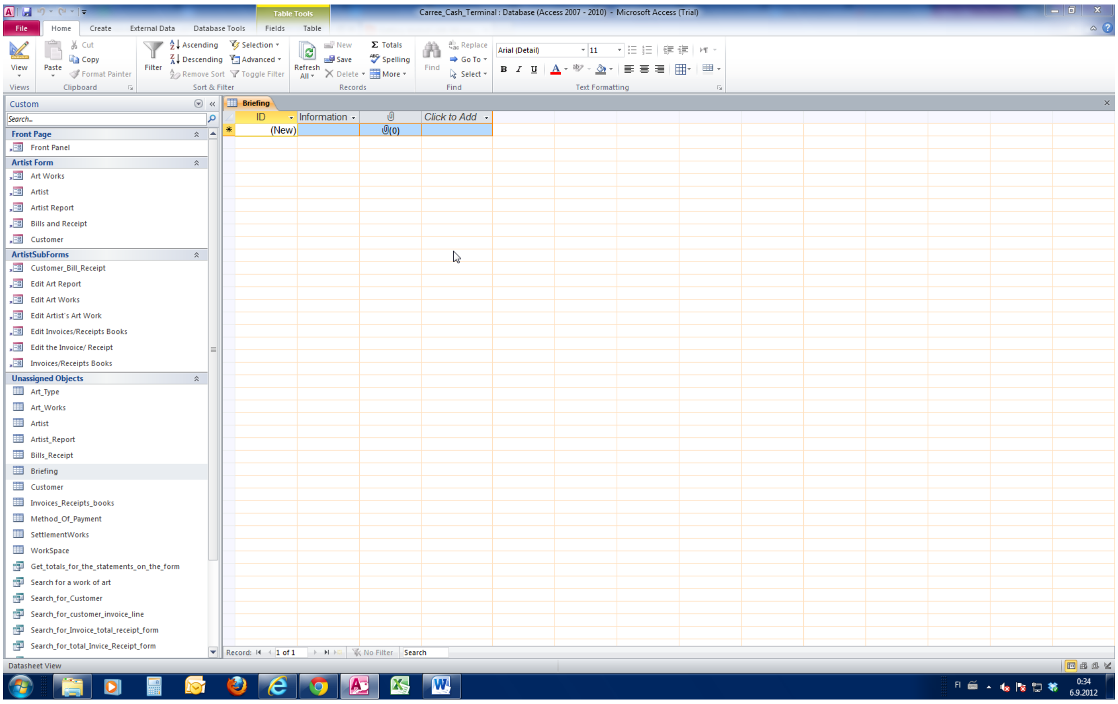


FIGURE 3. The datasheet view of Microsoft Access Database.

2.5 Programming Languages

Programming Languages are languages designed to provide a way of communicating with a computer or machine. The two types of programming languages include high-level that is machine independent and low-level that is machine dependent. An example of this is the assembly language. Next some programming languages are described in greater details (V Rajaraman 1998)

2.5.1 Java

Java is the most common programming language developed by Sun Microsystems. It is widely used due to some characteristics features like: (Carter 1997)

Portability: meaning that a single computer program that is written in Java can run on any operating system platform. Its libraries provide a way of host features like threading, networking and graphics.

2.5.2 C Sharp

C Sharp is a general-purpose programming language developed by Microsoft in 2001. Some of the goals of the language was that it should provide support for engineering principles such as arrays checking and be good for writing applications of various kinds like web, embedded systems etc. (Standard ECMA – 334, 2006)

2.5.3 Visual Basic .NET

Visual Basic .NET is an object oriented programming language designed mainly for Microsoft operating systems by Microsoft Corporation. Mainly used with .NET framework. (Michelbrink 2004)

2.6 Frameworks

Frameworks are software reusable design that contains a set of code libraries that provides functionality to a common set of applications. Most common software frameworks are developed using programming languages. Examples of these are ruby on rails framework developed with ruby programming language, Django framework developed with Python programming language and .Net framework developed with C# programming language. (Murray, Carrington and Strooper 2004)

2.6.1 .NET Framework

This was developed by Microsoft. It provides a comprehensive programming model for Microsoft platforms. Mainly used for designing Microsoft desktop applications. It is a software development tool that is covered about the operating system level. (Michelbrink 2004)

3 THE ART GALLERY AS CASE STUDY

The case study for this thesis describes an Art gallery that specialises in collecting work of art from artists and selling them to the customers. The company provides a way of interacting with the artist in a way that the customer can buy your favourite art item without knowing where the art work is from. These services provided by her Art Gallery Company include collecting art works, selling them to the buyers and paying the artists. The following information describes the data recorded, maintained and accessed by the art gallery to improve the maintaining and operation of the organisation

3.1 Data Requirements

Before creating a Cash register terminal for a gallery, there are few requirements that need to be considered. This requirement provides the bases of which the cash register terminal will be developed. This includes the Cash Handling section, Item Recording section, and Bank reference number generation.

3.1.1 Cash Handling

In every sales business, there is a need for cash handling section for store transactions. In this case, the transactions can involve selling items to customers, artist being paid for the art work sold and commission collected by the gallery.

3.1.2 Item Recording

The necessity of storing data in the database is very essential. In this case, art works are stored in the database for referencing. It makes it easier to check if an art work is sold or not.

3.1.3 Bank Reference Number

Bank Reference Number is numeric set of numbers for transfer clearance that can be issued by the invoicing party according to a standard. This is generated for a particular item like bills, receipts etc., for identifying a particular bill information from the other.

There is a standard that must be followed to generate this number. It includes the length of the number that must be between 2 and 20 digits and the reference number is calculated using the 7-3-1 method that allows the invoicing company to generate their own set of digits.

The 7-3-1 method comprises of a digits figure between 2 and 20 numeric characters length is calculated by multiplying 7,3,1,7,3,1 … from right to left. The value is the added up and subtracted from the nearest zero. The check digit is the difference from the result, then entered as the last digit in the reference number.

3.2 Transaction Requirements

These requirements are available for the effective management of the operator of the database. The required transactions include creating and maintaining records of the customers that purchase the art works, producing sales report for a particular period, printing receipt for customers when they purchase the art works and printing the artist sales report for the art work sold for referencing.

3.2.1 Data Modelling

Data modelling is described as the way of structuring and operating a database for a particular purpose. Structuring of data into various table models, data-types, and creating the relationships, which provides a way that the database can store, delete, insert, and modify data. (Shamkant B.Navathe 1992)

Figure 4 shows the structure of the tables created for Cash Register Terminal (Art Gallery)

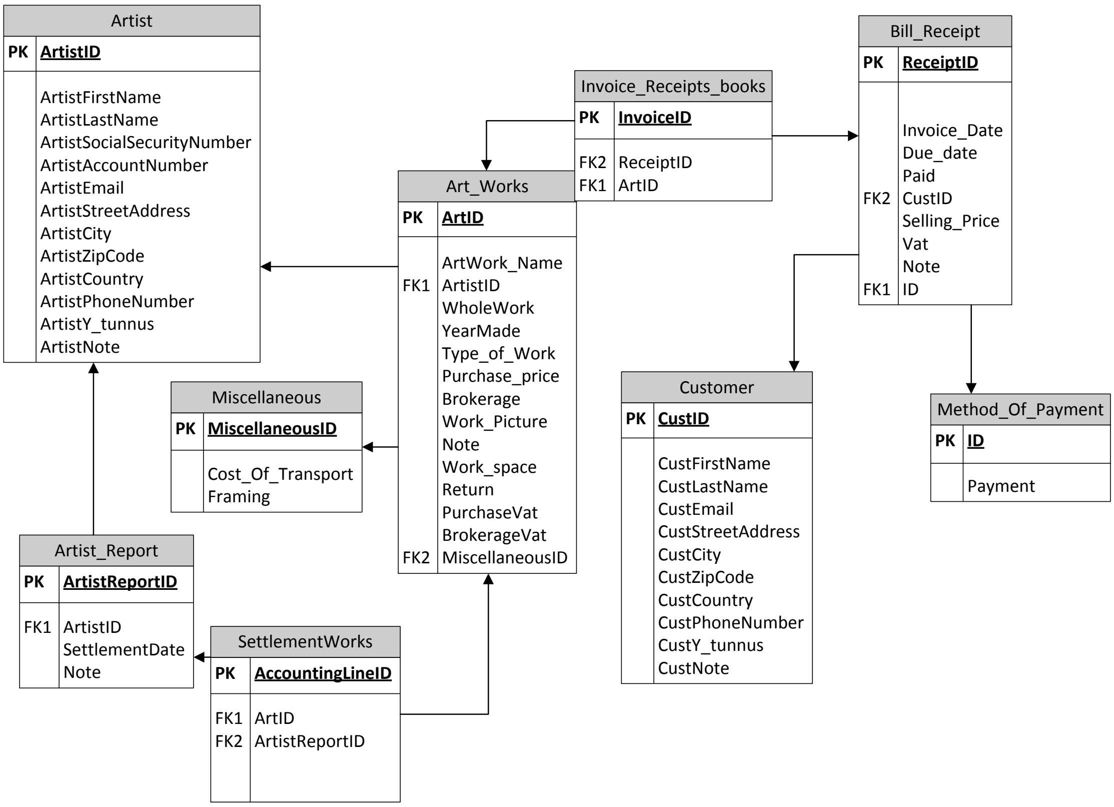


FIGURE 4. Database Entity Relationship Model

3.2.2 Table modelling

Creating the table where the data will be store is main part of developing a good database. At this point, the first question we asked earlier in the introduction will be answered. The database consists of the Artist information like Name, Street Address, the Art work information, Customers information, and Sales information. Figure 5 shows the structure flow of the cash register terminal.

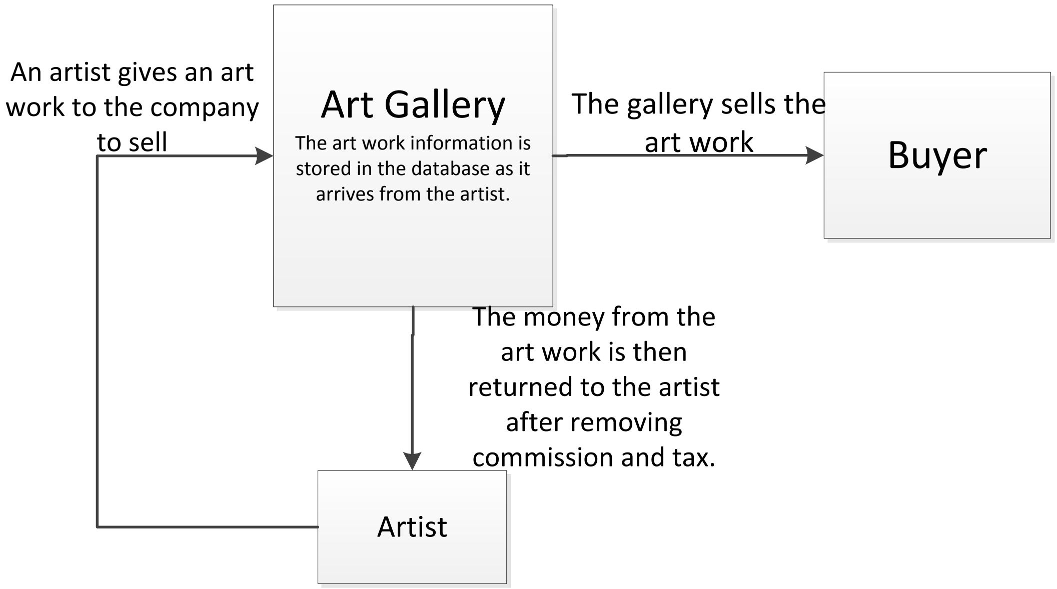

An artist gives an art work to the company to sell
Art Gallery
The art work information is stored in the database as it arrives from the artist.
The gallery sells the art work
Buyer
The money from the art work is then returned to the artist after removing commission and tax.
Artist

FIGURE 5. The structure flow of the cash register terminal

The art gallery system carries out a lot of functions. This includes the content of a table. The table is very important to the database. This content states how the data is stored in the database.

Some information to note is:
Primary key: This is the unique identifier of an item in the database. A table can only have one primary key. This unique identifier always consists of the ID key word
Foreign key: This is the referential constraint between two tables. A table can have one or more foreign keys.

These tables includes

TABLE 3. The tables and their characteristics

Table	**Characteristics**
Bills and Receipt	These stores the information for the invoice date for the art work sold.
Customers	This is where the customer information is stored for later referencing or purchase of artwork.
Artist	This stores the artist information.
Art work	This is where the art works data are stored when it arrives from artist.

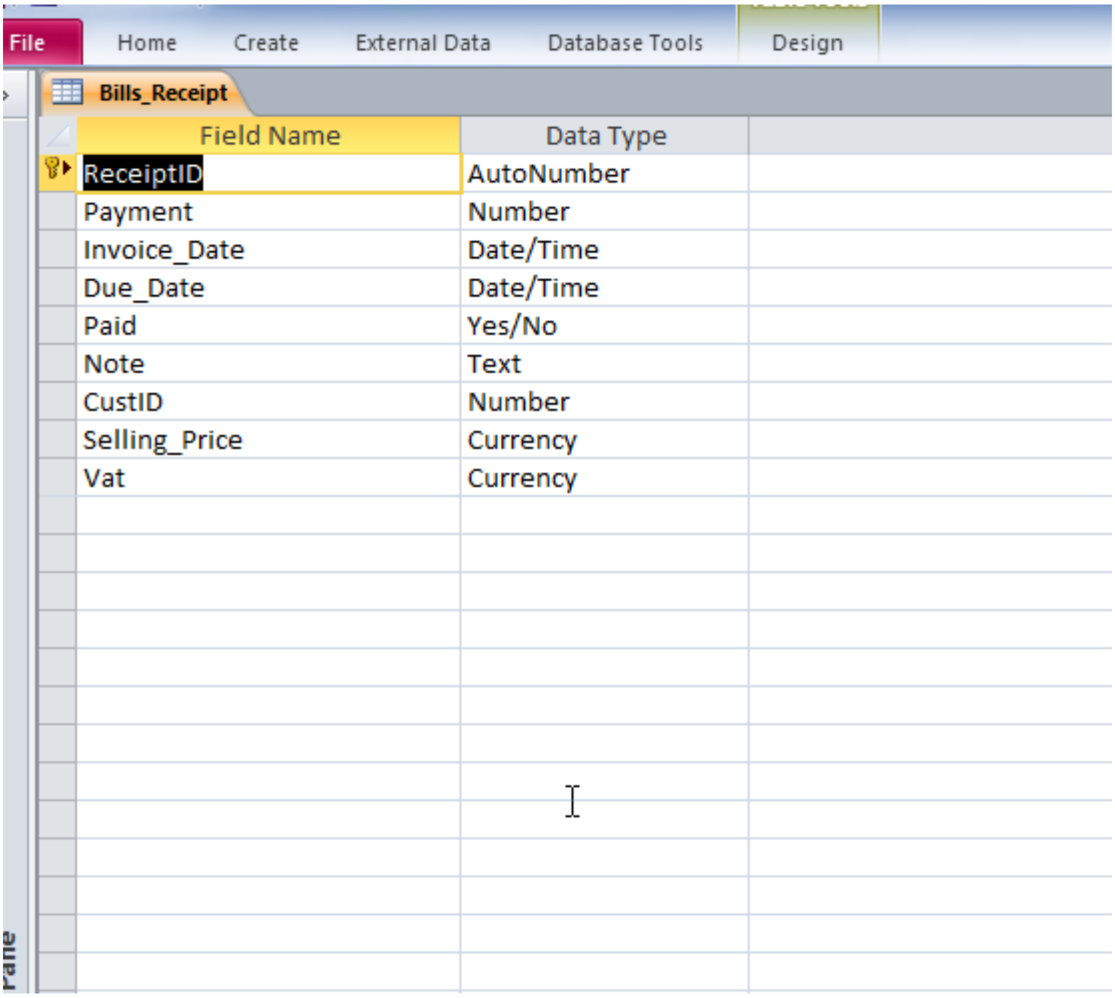
Artist Report	This stores the sales information that is sent to the artist as a sales report for art work sold.
Art Type	This is the list of the type of art work stored in the database. This includes painting, sculpture, graphics, art and design and other.
Method of Payment	This is a list of the payment type which includes cash, bills and debit payment.
Work space	This is a list of the art work availability. This includes gallery, reserved, sold, on loan and back. For example, if an artwork is still in store, then it is said to be in the gallery.

3.2.3.1 Bill and Receipt's Table

This consists of information that deals with the information about the art work sold to customers. Its field properties include payment type which is the collection of the method of payment table. This has a data type Number. List of the payment was gotten using the following SQL syntax.

```
SELECT Method_Of_Payment.ID, Method_of_Payment
FROM Method_Of_Payment;
```

Invoice Date that keeps records the date of the purchase of the artwork, due date which stores the due date for any artwork purchase on credit, Paid field, Selling price, and Vat. Figure 6 shows the design view of the bills and receipt table.



Field Name	Data Type
ReceiptID	AutoNumber
Payment	Number
Invoice_Date	Date/Time
Due_Date	Date/Time
Paid	Yes/No
Note	Text
CustID	Number
Selling_Price	Currency
Vat	Currency

FIGURE 6. The design view of the Bills and Receipt table

3.2.3.2 Customer Table

The customer’s table stores the information of each person that purchases an art work. Its field properties include First Name and Last Name, Email, Street Address, City, ZipCode, Country, Phone Number and Note.

3.2.3.3 Artist Table

Artist’s table stores artist information, which is similar to the customer’s table. It contains fields like name, address, Street Address, City, Zipcode, Country, Phone Number, Social Security number, and Account Number

3.2.3.4 Art Work Table

Art work stores the art work information. The fields’ includes Art Work Name, YearMade which stores the year the art work was made, purchase price, brokerage, work picture, vat and Type of Work which is a FK for the art-type table. Creating a combo box that was select from its Primary table using

```
SELECT Art_Type.ID, Art_Type.Art_Type
FROM Art_Type;
```

3.2.3.5 Art Work Report

Art work stores the art work report information. The fields includes, ArtistID which is the FK of the Artist table, Settlement Date, Commission sum and vat, Purchase Price sum and vat, Cost of Transport, Framing cost: Stores the cost of frame used to enclose an artwork.

4 USER INTERFACE DESIGN

User interface design is an important part of any software application for human interaction with the system. The user interface (UI) is considered to be good when it is easy to use by the staff or a novice who does not know anything about the development of the application. (Introducing User Interface Design)

Figure 7 shows the sketch for the front page of the application. This shows how the front end will look like when the application is completed.

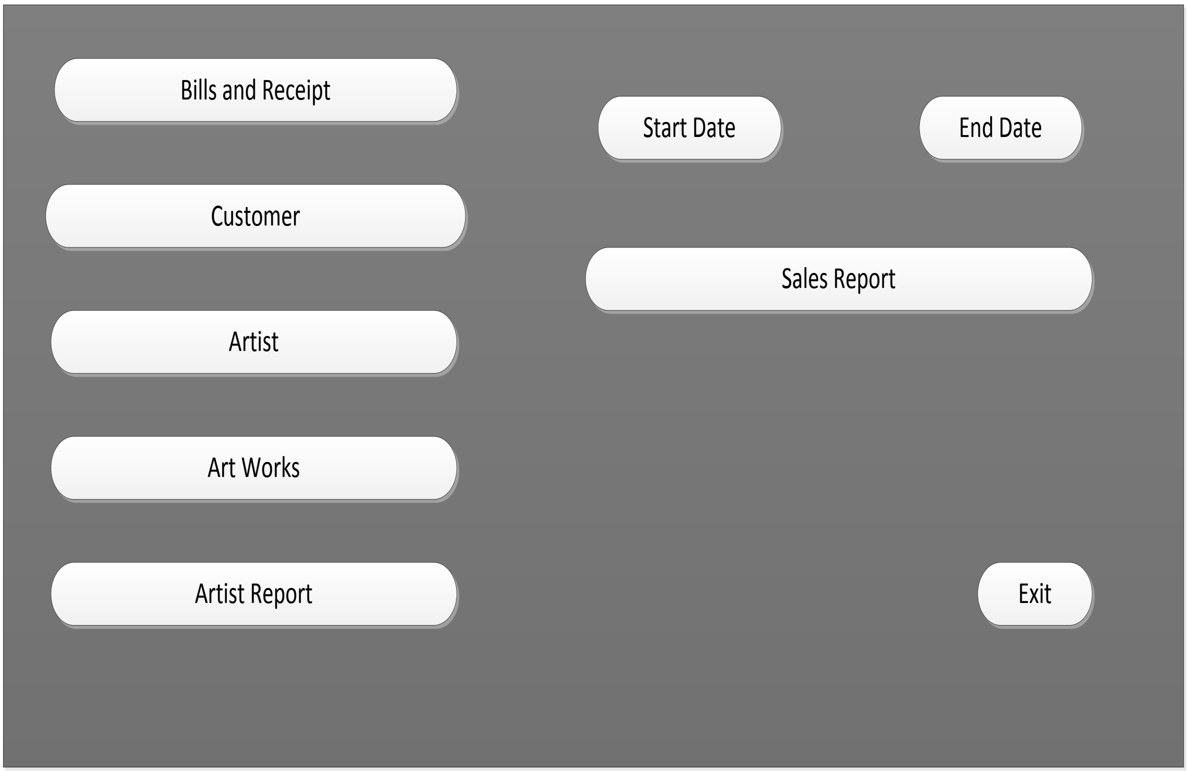


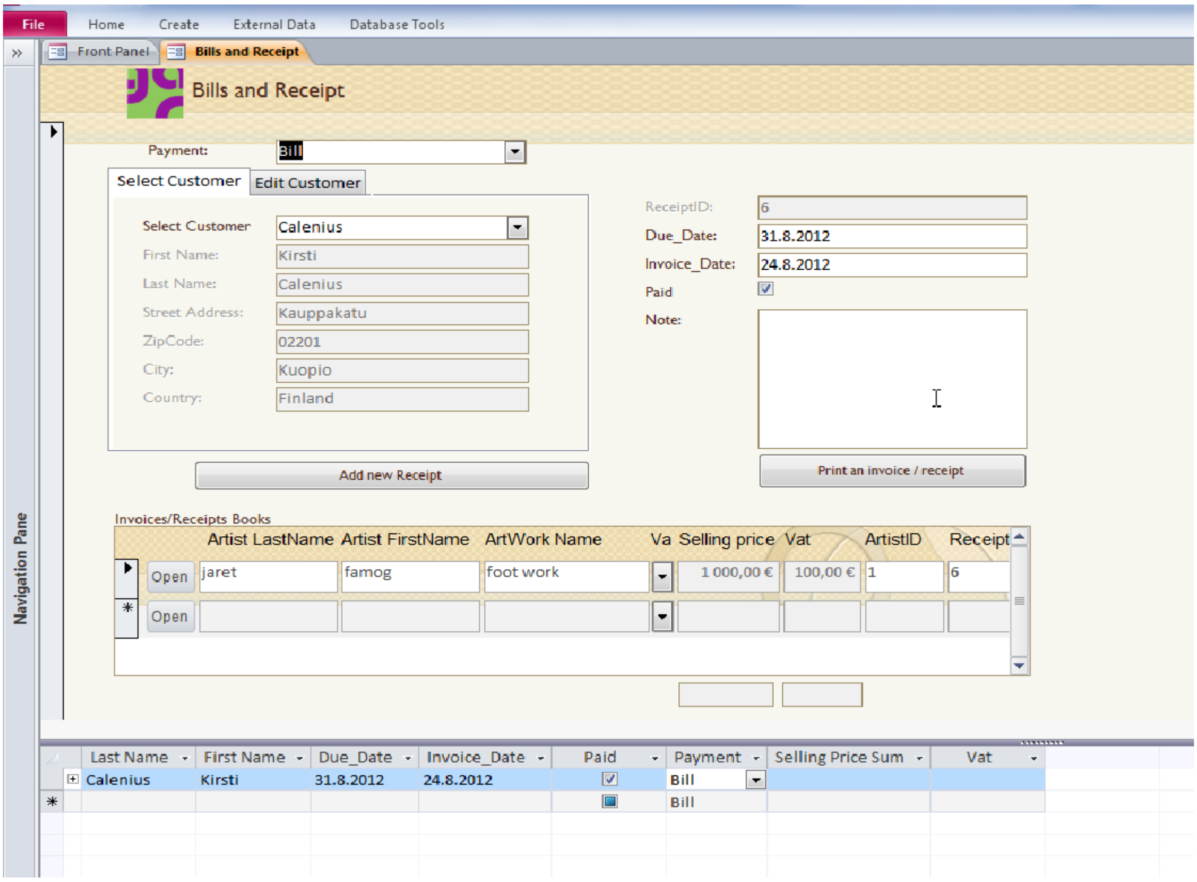
FIGURE 7. The welcome section of the application.

The welcome section consists of various buttons for accessing each section in the application. In the following sections, I will tell on each section, and how it was implemented.

4.1 Bill and Receipt User Interface

The bills and receipt user interface provides a good interface for recording the bills and receipt data. Information for printing, for sales records of art works sold. The Form was created using the following query:

```
SELECT Bills_Receipt.*, Customer.CustFirstName, Custom-
er.CustLastName, Customer.CustStreetAddress, Custom-
er.CustCity, Customer.CustZipCode, Customer.CustCountry,
Customer.CustY_tunnus
FROM Customer INNER JOIN Bills_Receipt ON Customer.CustID =
Bills_Receipt.CustID;
```



Bills and Receipt

Payment: Bill

Select Customer	Calenius
First Name:	Kirsti
Last Name:	Calenius
Street Address:	Kauppakatu
ZipCode:	02201
City:	Kuopio
Country:	Finland

ReceiptID:	6
Due_Date:	31.8.2012
Invoice_Date:	24.8.2012
Paid	☑
Note:	

Invoices/Receipts Books

	Artist LastName	Artist FirstName	ArtWork Name	Va	Selling price	Vat	ArtistID	Receipt
Open	jaret	famog	foot work		1 000,00 €	100,00 €	1	6
Open								

	Last Name	First Name	Due_Date	Invoice_Date	Paid	Payment	Selling Price Sum	Vat
⊞	Calenius	Kirsti	31.8.2012	24.8.2012	☑	Bill		
*					□	Bill		

FIGURE 8. Bills and Receipt UI

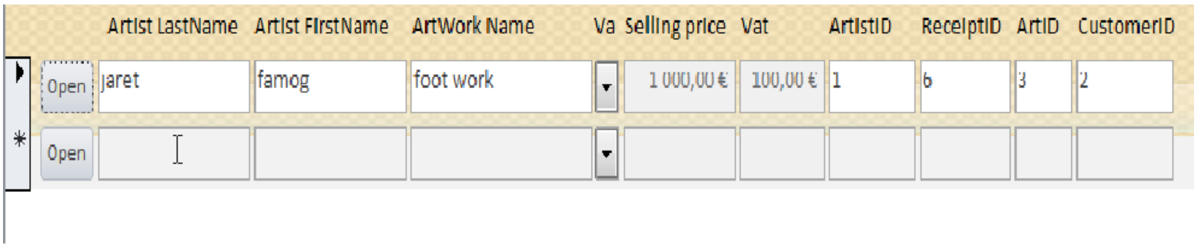
Figure 8 shows the user interface for the bills and receipt section. The form shows the user interface where the user can interact with the database. The payment type was created using a combo box. It consists of Bill, Debit Card and Cash column.

The customer information selection and editing was created using a tab control. The Select Customer is the field where the customer that is stored in the database can be view and reused if the customer purchases an art work some other time. This uses a combo box to lookup the customer table and return the information required. The following SQL query was used to get the data of the combo box.

```
SELECT Customer.CustID, Customer.CustLastName, Custom-
er.CustFirstName, Customer.CustCity
FROM Customer
ORDER BY Customer.CustLastName;
```

The Edit Customer is used to enter new customers or edit the current data collected from the customer table, the ReceiptID is used to uniquely identify each receipt. The due date column that provides the due date for the payment of the art work sold and Invoice date is the date the art work was sold. Before this information is added. The user will want to enter the information of the art work that the customer wants to buy. The form was created using the following query:

```
SELECT Invoices_Receipts_books.*, Art_Works.ArtWork_Name,
Art_Works.YearMade, Art_Works.Purchase_price,… (See Appendix A
number 1)
```

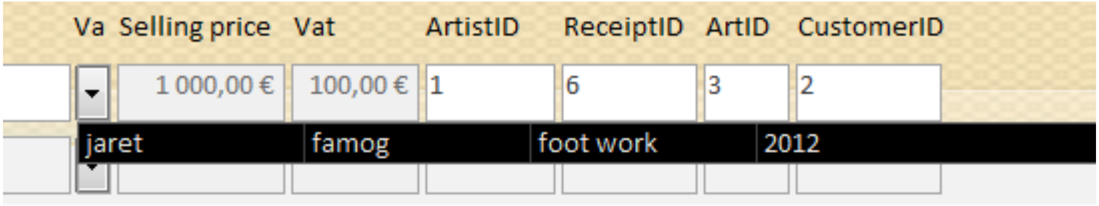


	Artist LastName	Artist FirstName	ArtWork Name	Va	Selling price	Vat	ArtistID	ReceiptID	ArtID	CustomerID
Open	jaret	famog	foot work		1 000,00 €	100,00 €	1	6	3	2
Open										

FIGURE 9. Art work UI

Figure 9 shows a view where the user can select the art work. The Va is a combo box that contain the art work information that is stored in the Art Work table. This was created using the following query:

```
SELECT Art_Works.ArtID, Artist.ArtistLastName, Art-
ist.ArtistFirstName, Art_Works.ArtWork_Name,
Art_Works.YearMade, … (See appendix A number 2)
```

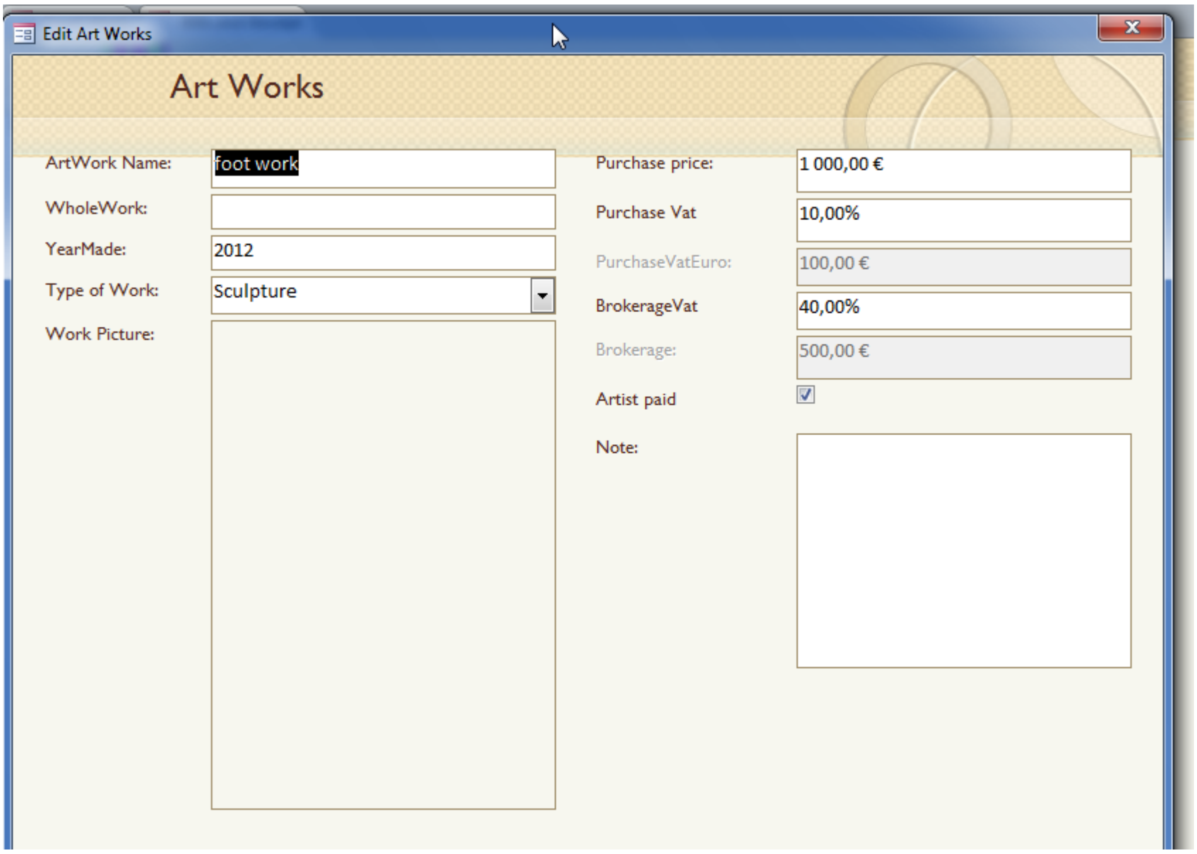


Va	Selling price	Vat	ArtistID	ReceiptID	ArtID	CustomerID
	1 000,00 €	100,00 €	1	6	3	2

jaret	famog	foot work	2012

FIGURE 10. The list of the current art work in the database

Figure 10 show the art work information. This art work will automatically populate the field from the database. The Open button is where the information can be edited and clicking this button will open the form on figure 11.



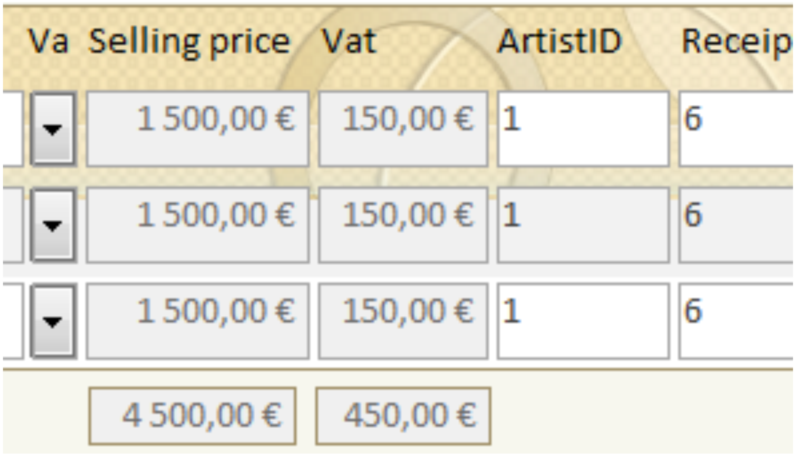
ArtWork Name:	foot work	Purchase price:	1 000,00 €
WholeWork:		Purchase Vat	10,00%
YearMade:	2012	PurchaseVatEuro:	100,00 €
Type of Work:	Sculpture	BrokerageVat	40,00%
Work Picture:		Brokerage:	500,00 €
		Artist paid	☑
		Note:	

FIGURE 11. Edit Art work UI

The open button opens the art work where the button was clicked for editing. The following information can be edited except for the purchasing Vat in Euro and Brokerage which are calculated using the following mathematical equation:

PurchaseVatEuro = PurchaseVat % * Purchase price

Brokerage = PurchaseVatEuro + Purchase price * BrokerageVat %



Va	Selling price	Vat	ArtistID	Receip
	1 500,00 €	150,00 €	1	6
	1 500,00 €	150,00 €	1	6
	1 500,00 €	150,00 €	1	6
	4 500,00 €	450,00 €		

FIGURE 12. The calculated total selling price and vat

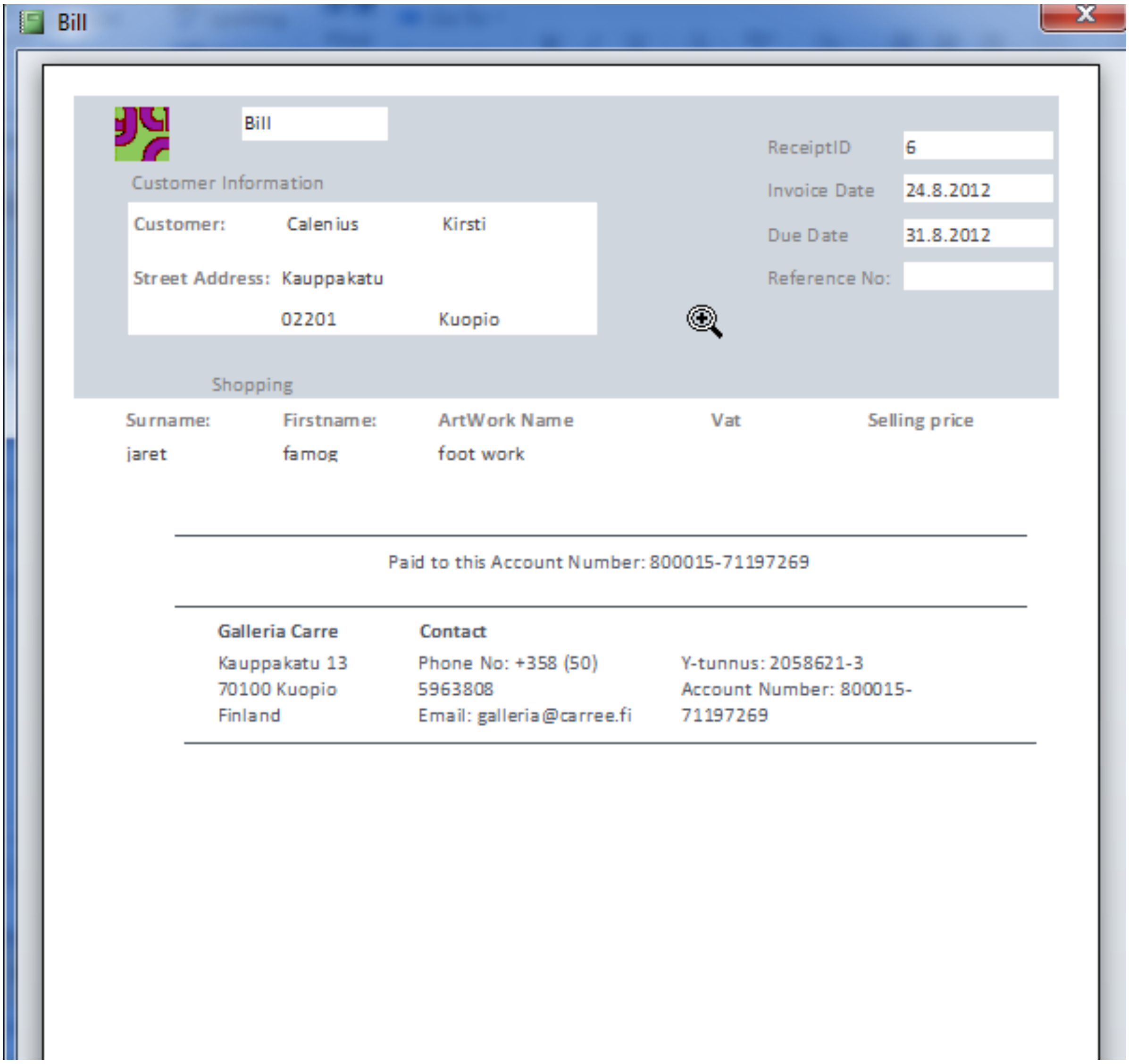
Calculate a certain column of a record can be done using DSum function. DSum function takes three parameters listed in the Table 4. Creating the using the following VB.NET code

```
Dim Sum As Variant
Sum = DSum (‘Purchase_price’;’Sum Query’; ‘ReceiptID’)
```

TABLE 4. Lists the parameters available for the DSum function.

Expression	This is the numerical values or the name of the column that needs to be calculated. In this case, it is the purchase price
Domain	This is the record name. This can be the table where the expression is listed. In this case, it is the Sum query of the sub form.
Criteria	In SQL, this is the WHERE clause that specify the record data, this field is optional. Which is the ReceiptID in this case.

After this information has been entered, the button (Add new Receipt) can be click to add the information to the database. The same applies to the print button which shows the preview on figure 13.



Bill

Bill

ReceiptID: 6

Invoice Date: 24.8.2012

Due Date: 31.8.2012

Reference No:

Customer Information

Customer: Calenius Kirsti

Street Address: Kauppakatu

02201 Kuopio

Shopping

Surname:	Firstname:	ArtWork Name	Vat	Selling price
jaret	famog	foot work		

Paid to this Account Number: 800015-71197269

Galleria Carre
Kauppakatu 13
70100 Kuopio
Finland

Contact
Phone No: +358 (50) 5963808
Email: galleria@carree.fi

Y-tunnus: 2058621-3
Account Number: 800015-71197269

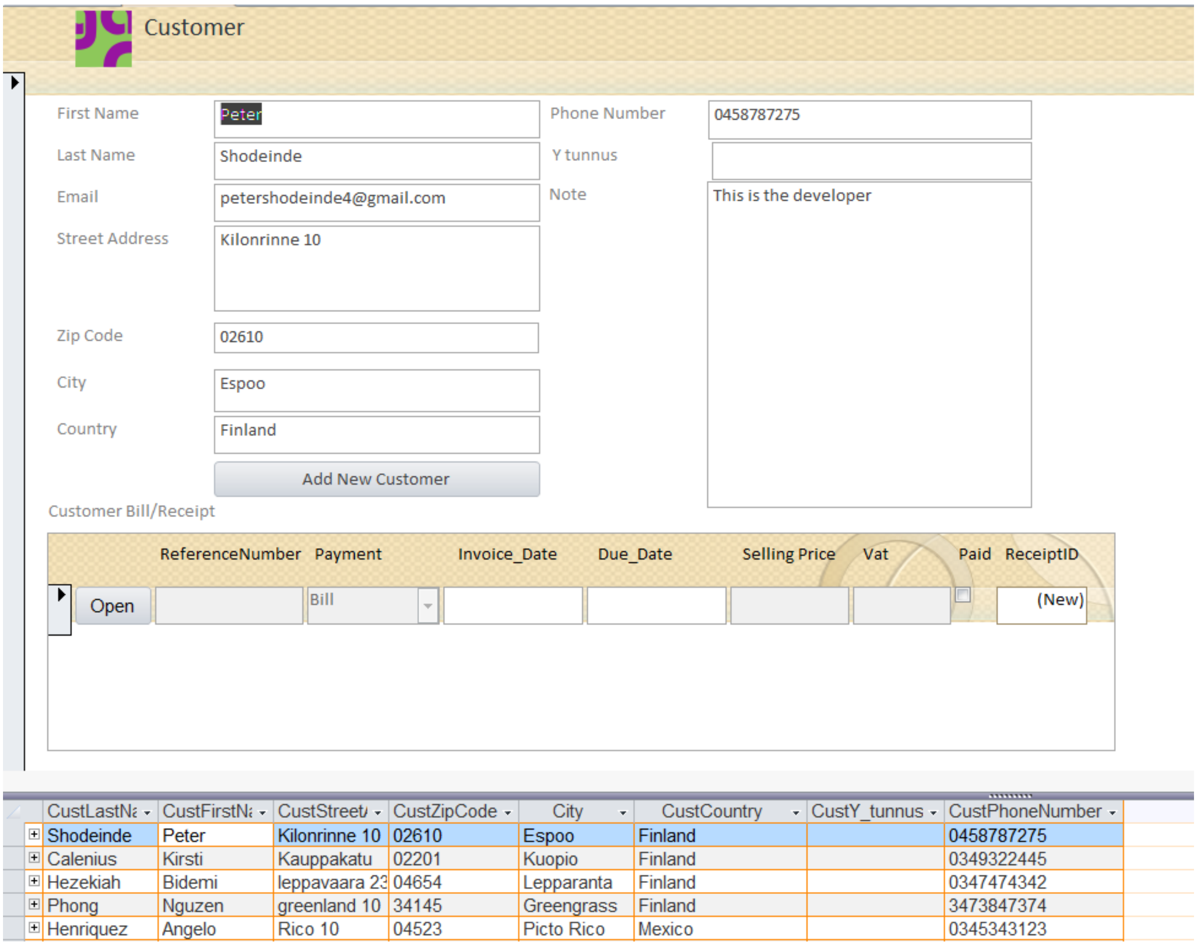
FIGURE 13. Preview of the receipt

This provides the customers with the necessary information about the art work purchased. The following query was used to create the following field in the report form.

```
SELECT Invoices_Receipts_books.*, Bills_Receipt.CustID,
Art_Works.ArtWork_Name, Artist.ArtistLastName, Art-
ist.ArtistFirstName,… (See Appendix A number 3)
```

4.2 Customer User Interface

Customer User Interface provides the user with the customer's information that has purchase the art work with the list of receipt information. Figure 14 shows the customer's user interface.



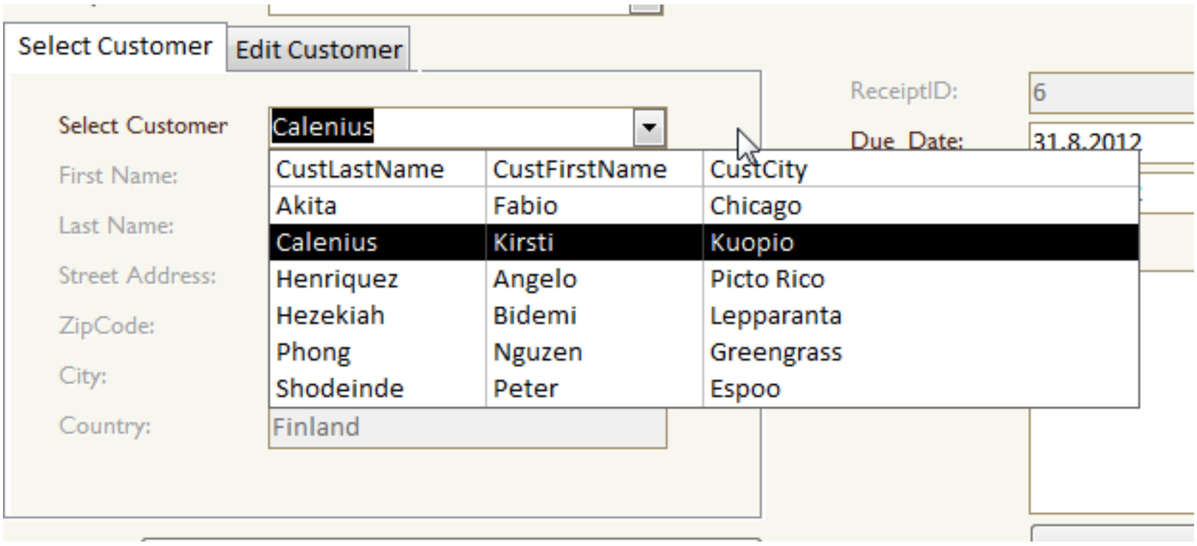
First Name	Peter	Phone Number	0458787275
Last Name	Shodeinde	Y tunnus	
Email	petershodeinde4@gmail.com	Note	This is the developer
Street Address	Kilonrinne 10		
Zip Code	02610		
City	Espoo		
Country	Finland		

	ReferenceNumber	Payment	Invoice_Date	Due_Date	Selling Price	Vat	Paid	ReceiptID
Open		Bill					□	(New)

CustLastNa	CustFirstNa	CustStreetA	CustZipCode	City	CustCountry	CustY_tunnus	CustPhoneNumber
Shodeinde	Peter	Kilonrinne 10	02610	Espoo	Finland		0458787275
Calenius	Kirsti	Kauppakatu	02201	Kuopio	Finland		0349322445
Hezekiah	Bidemi	leppavaara 23	04654	Lepparanta	Finland		0347474342
Phong	Nguzen	greenland 10	34145	Greengrass	Finland		3473847374
Henriquez	Angelo	Rico 10	04523	Picto Rico	Mexico		0345343123

FIGURE 14. Customer User Interface

The user enters the information of the customers with their respective receipt information. The UI was created from the customer table. Clicking the button (Add New Customer), adds the customers. Whenever there is any look up for Customers in the Select Customer tab in the Bill and Receipt UI, the information for the customer is added to the lookup. Figure 15 shows the lookup menu for the customer information available in the database.



Select Customer: Calenius

CustLastName	CustFirstName	CustCity
Akita	Fabio	Chicago
Calenius	Kirsti	Kuopio
Henriquez	Angelo	Picto Rico
Hezekiah	Bidemi	Lepparanta
Phong	Nguzen	Greengrass
Shodeinde	Peter	Espoo

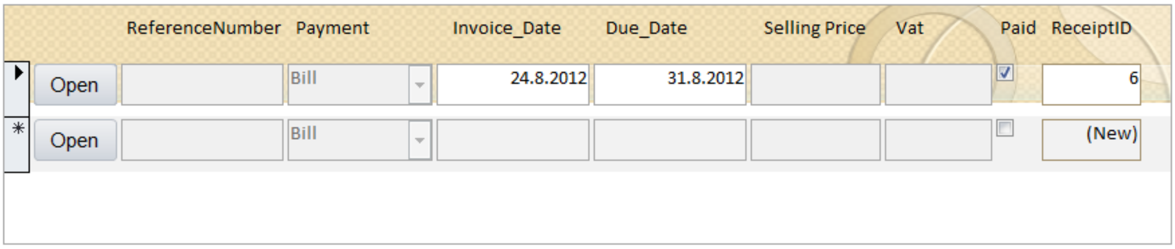
Country: Finland

ReceiptID: 6

Due Date: 31.8.2012

FIGURE 15. Bill and Receipt UI Lookup menu.

After a customer has purchase an art work, then the information for the receipt will be imported into the Bill and Report Sub form in the Customer UI.

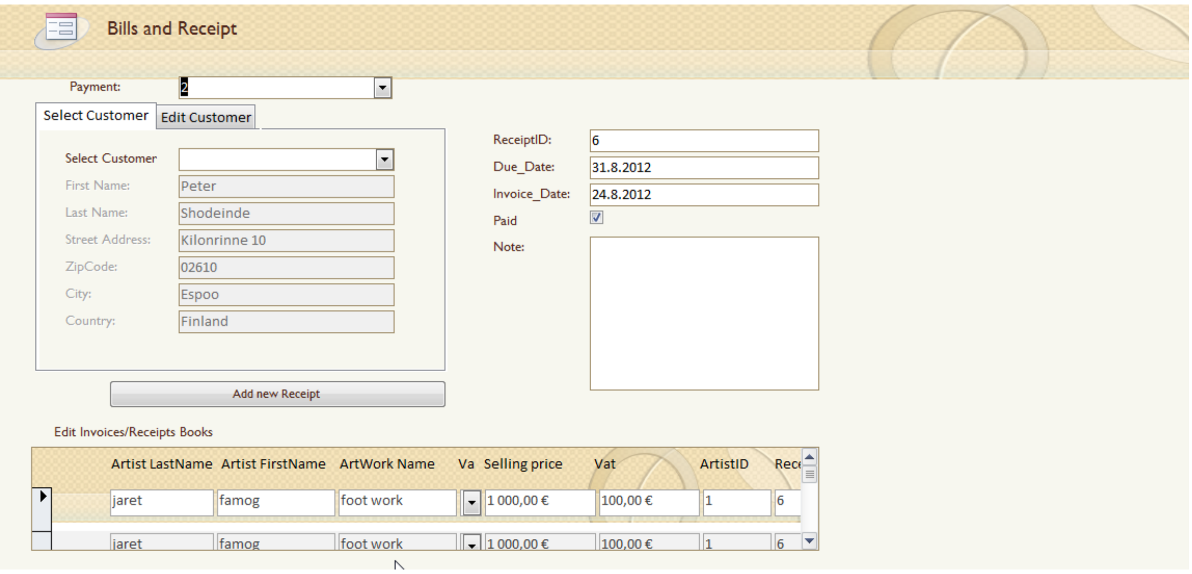


	ReferenceNumber	Payment	Invoice_Date	Due_Date	Selling Price	Vat	Paid	ReceiptID
Open		Bill	24.8.2012	31.8.2012			☑	6
Open		Bill					☐	(New)

FIGURE 16. Shows the Receipt UI in the Customer UI

The section adds the receipt for each customer. The information in the Figure 16 is the information of each receipt, with the reference number been generated.

The Reference Number is an important part of the receipt. As explained in Section 3.1.3, Creating this reference number, will generate a five digit number for each invoicing receipt for the customer. (See Appendix A number 6, to see code for the reference number). The open button shows the information of each art work purchase for each receipt. Figure 17 shows the edit view when the open button was clicked.



Bills and Receipt

Payment: 2

Select Customer	
First Name:	Peter
Last Name:	Shodeinde
Street Address:	Kilonrinne 10
ZipCode:	02610
City:	Espoo
Country:	Finland

ReceiptID:	6
Due_Date:	31.8.2012
Invoice_Date:	24.8.2012
Paid	☑
Note:	

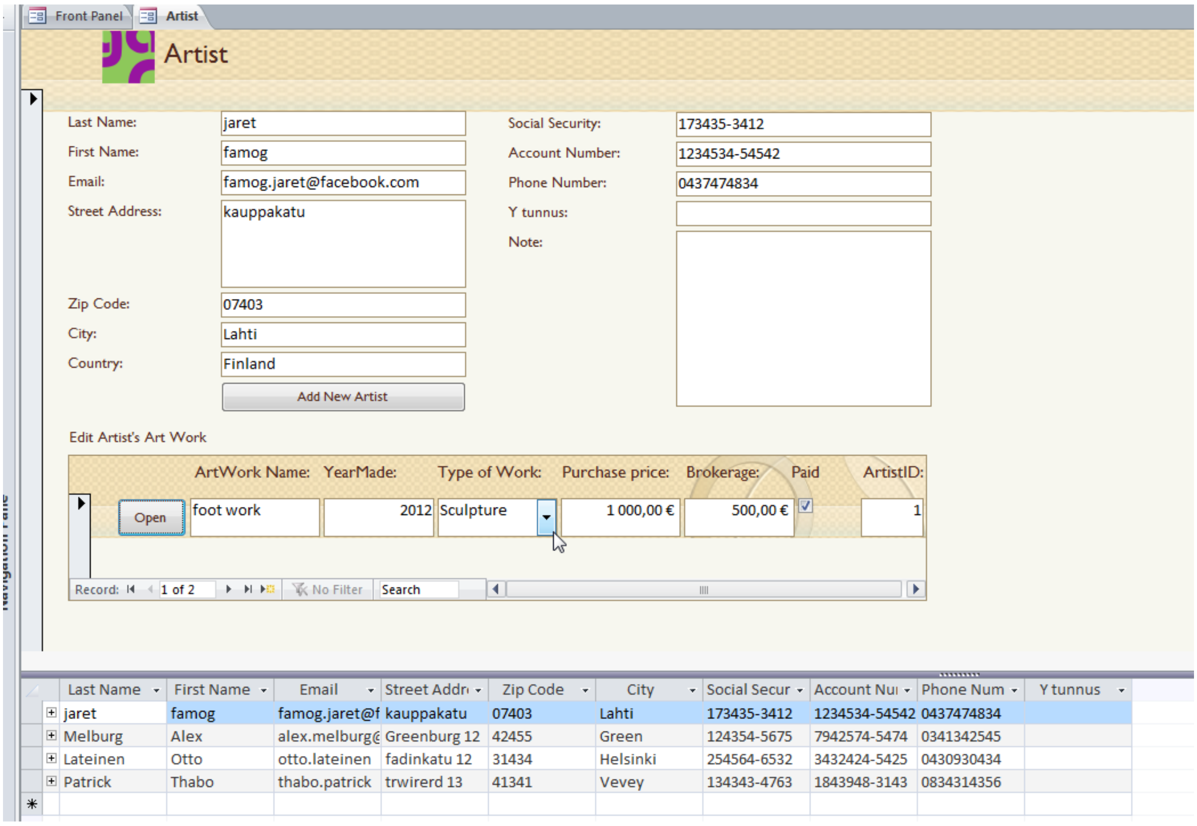
Edit Invoices/Receipts Books

Artist LastName	Artist FirstName	ArtWork Name	Va	Selling price	Vat	ArtistID	Rece
jaret	famog	foot work		1 000,00 €	100,00 €	1	6
jaret	famog	foot work		1 000,00 €	100,00 €	1	6

FIGURE 17. Show the edit view for the receipt.

4.3 Artist User Interface

Artist information needs to be stored as well, along with their art works. This makes it easier for the user to identify the owner of each art works in the gallery. Figure 18 shows the Artist User Interface.



Artist

Last Name:	jaret
First Name:	famog
Email:	famog.jaret@facebook.com
Street Address:	kauppakatu
Zip Code:	07403
City:	Lahti
Country:	Finland

Social Security:	173435-3412
Account Number:	1234534-54542
Phone Number:	0437474834
Y tunnus:	
Note:	

Edit Artist's Art Work

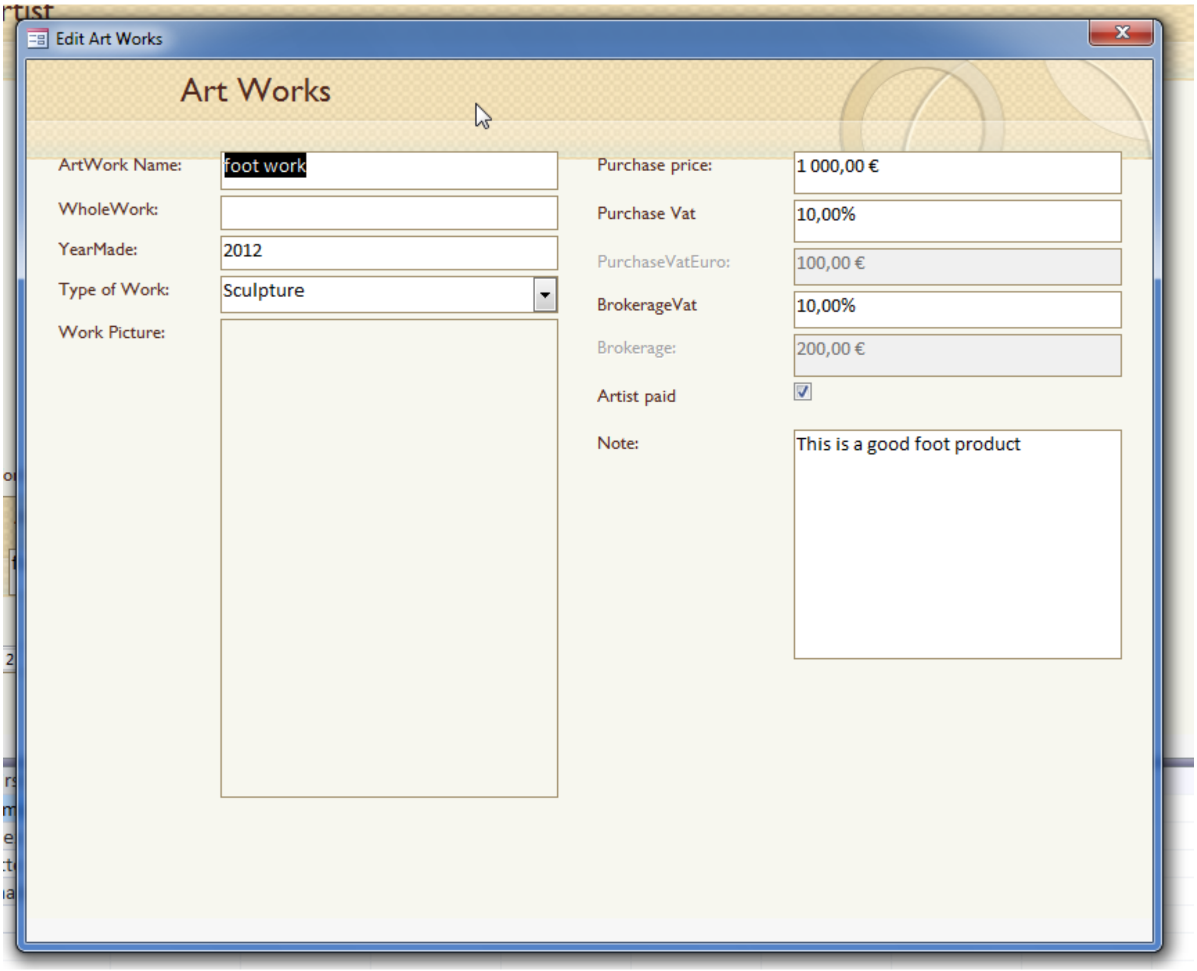
	ArtWork Name:	YearMade:	Type of Work:	Purchase price:	Brokerage:	Paid	ArtistID:
Open	foot work	2012	Sculpture	1 000,00 €	500,00 €	☑	1

Record: 1 of 2 No Filter Search

Last Name	First Name	Email	Street Addr	Zip Code	City	Social Secur	Account Nu	Phone Num	Y tunnus
jaret	famog	famog.jaret@f	kauppakatu	07403	Lahti	173435-3412	1234534-54542	0437474834	
Melburg	Alex	alex.melburg@	Greenburg 12	42455	Green	124354-5675	7942574-5474	0341342545	
Lateinen	Otto	otto.lateinen	fadinkatu 12	31434	Helsinki	254564-6532	3432424-5425	0430930434	
Patrick	Thabo	thabo.patrick	trwirerd 13	41341	Vevey	134343-4763	1843948-3143	0834314356	

FIGURE 18. Artist User Interface

The UI was created from the Artist table, The Edit Artist's Art work section provides the user with the information of the Artist's art works. The open button provides the user with a UI where each art work can be edited. This was created from the art work table. Figure 19 shows the Edit Art work UI.



ArtWork Name:	foot work	Purchase price:	1 000,00 €
WholeWork:		Purchase Vat	10,00%
YearMade:	2012	PurchaseVatEuro:	100,00 €
Type of Work:	Sculpture	BrokerageVat	10,00%
Work Picture:		Brokerage:	200,00 €
		Artist paid	☑
		Note:	This is a good foot product

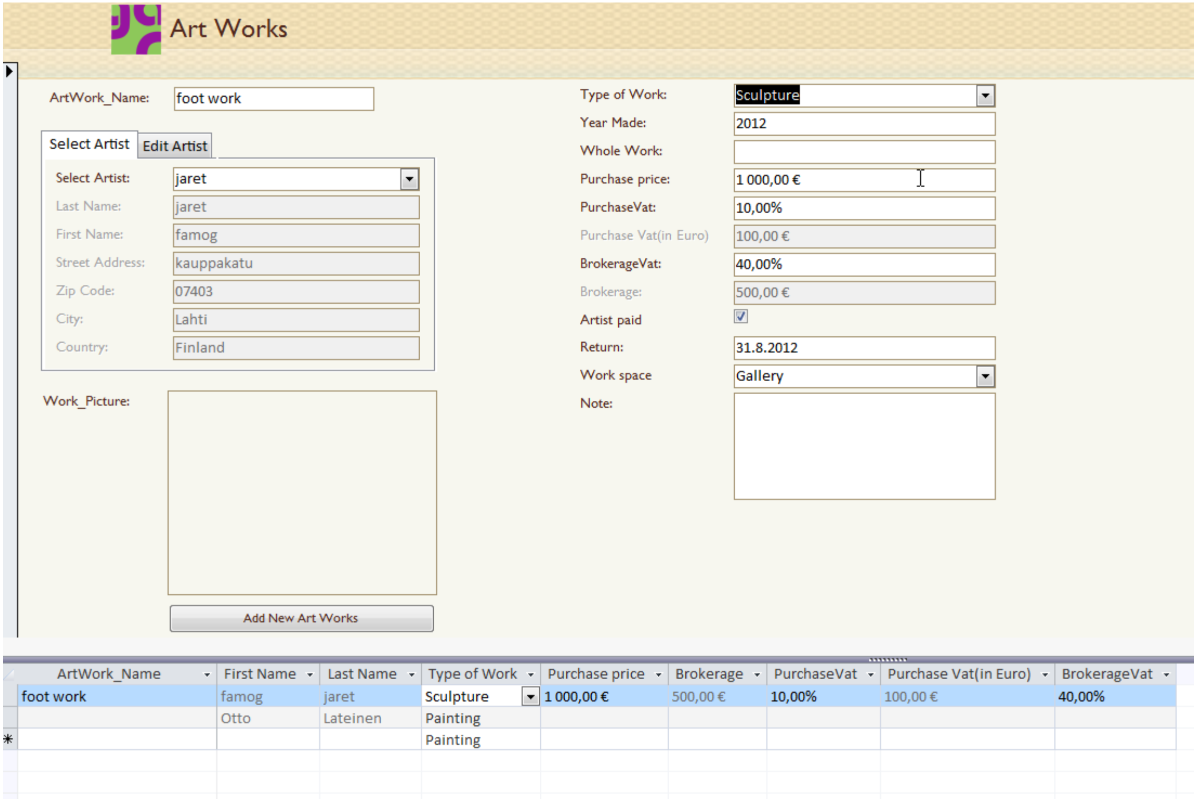
FIGURE 19. Shows a UI where art work can be edited.

4.4 Art Work User Interface

Art Work User Interface provides an interface for art work information like name, price and so on. This makes it easy to get the information of the art work. Figure 20 shows the art work UI. This was created using the query;

```
SELECT Art_Works.*, Artist.ArtistLastName, Art-
ist.ArtistFirstName, Artist.ArtistStreetAddress, Art-
ist.ArtistCity, Artist.ArtistZipCode, Artist.ArtistCountry
```

```
FROM Artist INNER JOIN Art_Works ON Artist.ArtistID =
Art_Works.ArtistID;
```



Art Works

ArtWork_Name:	foot work

Select Artist:	jaret
Last Name:	jaret
First Name:	famog
Street Address:	kauppakatu
Zip Code:	07403
City:	Lahti
Country:	Finland

Type of Work:	Sculpture
Year Made:	2012
Whole Work:	
Purchase price:	1 000,00 €
PurchaseVat:	10,00%
Purchase Vat(in Euro)	100,00 €
BrokerageVat:	40,00%
Brokerage:	500,00 €
Artist paid	☑
Return:	31.8.2012
Work space	Gallery
Note:	

	ArtWork_Name	First Name	Last Name	Type of Work	Purchase price	Brokerage	PurchaseVat	Purchase Vat(in Euro)	BrokerageVat
	foot work	famog	jaret	Sculpture	1 000,00 €	500,00 €	10,00%	100,00 €	40,00%
		Otto	Lateinen	Painting					
*				Painting					

FIGURE 20. Art Work User Interface

The select customer tab control was created using the same logic as the bills and receipt UI. Work picture section is an attachment control box for adding art work images if necessary. Providing the other information creates a new art work data ready to be sold.

4.5 Artist Report User Interface

Artist Report User Interface provides the user with a report to give to the artist for the sale of an art work or group of art works in a given period of time. This includes information of the art work and the price it was sold. These provides information like the settlement date, Artist account number etc. See Appendix 4 for the query for the Artist Report UI. Figure 21 shows the Artist Report GUI

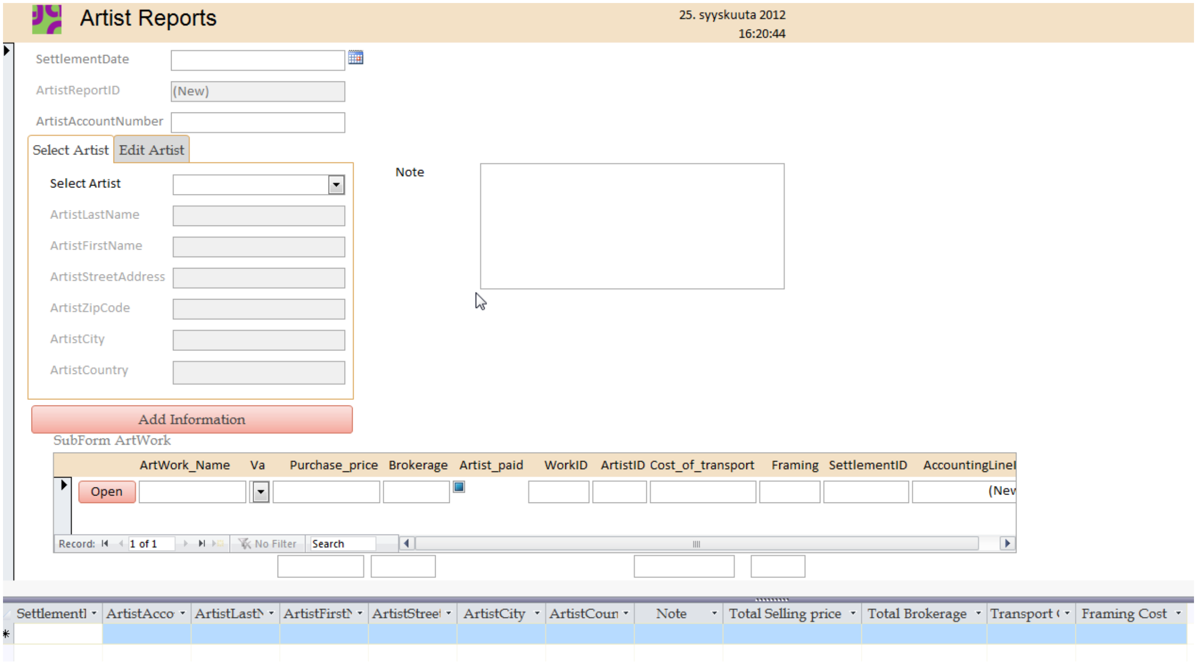


FIGURE 21. Artist Report UI

The Artist Report User Interface provides a good means of getting the records of the artist using the tab control box and using the sub form section for listing the art work sold for that particular artist.

The total selling price for all the art work was calculated using the DSum function in vb. This also applies to the brokerage, Cost of transport and framing cost, substituting various column names with the Purchase_price. See Appendix 5 for the query of the sub form section. figure 22 shows the sub-form for the Artist Report UI.

```
Dim Sum As Variant
Sum = DSum(‘Purchase_price’;’Sum Query’; ‘ReceiptID’)
```

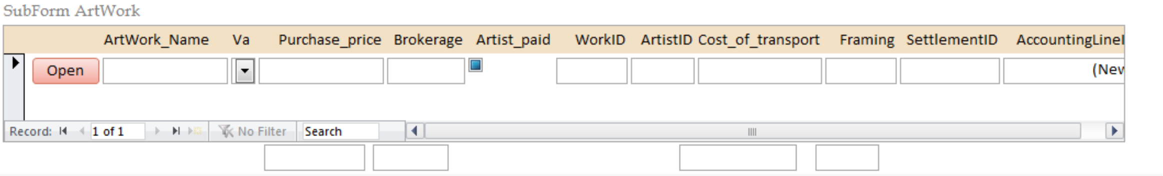


FIGURE 22. Shows the sub-form UI

5 CONCLUSION

This thesis focused on developing a cash terminal for an art gallery. It was implemented using a well-organised integrated application platform (Microsoft Access) and interactive programming language (VB.NET). The programming language was used to develop the calculation involved in the cash terminal.

Testing the program shows that the tool is suitable for storing cash related information. The structure of the database follows the object-oriented databases structure, which means that it is possible to add more structure to the database when there is need for it. This application can be developed and improved further by adding some functionality like sending receipts and report by email to customer and artist respectively, and giving notice to the user when an item is out of stock or remains one.

REFERENCES

Bank Bar Code Guide version 5.2 (2012) [e-journal]. [Accessed September 12, 2012]. Available from:
http://www.fkl.fi/en/material/publications/Publications/Bank_bar_code_guide_v052.pdf

Carter, P. (1997). *An Introduction to the Java Programming Language* [e-journal]. [September 7, 2012]. Available from:
http://www.cs.binghamton.edu/~guydosh/cs350/JavaPrimer.pdf

Connolly,T. and Begg,C., (2004). *Database Systems: A Practical Approach to Design, Implementation and Management (4th Edition)* [Wesley, A.].

Groff,J. & Weinberg,P. (1999). *SQL: The Complete Reference* [Electronic book].
Available from:
http://nadula.info/downloads/books/sql-the-complete-reference.9780072118452.1079.pdf

Introducing User Interface Design [Electronic book]. Available from:
http://www.elsevierdirect.com/companions/9780120884360/casestudies/Chapter_01.pdf

Michelbrink,P. (2004). *Visual Basic .Net tutorial 2004* [e-journal]. [Accessed in September, 5, 2004]. Available from:
http://www.cs.iusb.edu/internship/papers/pmichelbrink/VisualStack.pdf

Murray,L., Carrington,D., & Strooper,P., (2004). *An approach to specifying software frameworks.* Australia: School of Information Technology and Electrical Engineering
The University of Queensland.

Oracle 8. (2010).[Electronic book] Available from:
http://tinman.cs.gsu.edu/~raj/books/Oracle8-chapter-3.pdf

PRF Brown, (2002). *A brief history of modern RDBMS IT management* [e-journal]. [Accessed September 10, 2012]. Available from:
http://www.mountainman.com.au/software/history/it2.html

Rajaraman,V. (1998). *Programming languages* [e-journal]. [Accessed September 4, 2012] Available from:
http://www.ias.ac.in/resonance/Dec1998/pdf/Dec1998p43-54.pdf

Shamkant, B.N., (1992). *Evolution of Data Modeling for Databases. Communications of the ACM* [e-journal]. [Accessed September 6, 2012] Available from: http://www.cc.gatech.edu/~sham/classpapers/p112-navathe.pdf

Standard ECMA–334, (2006). *C# language Specification 2006* [e-journal]. [Accessed September 4, 2012]. Available from: http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-334.pdf

The reference number and the check digit (2009) [e-journal]. [Accessed September 11, 2012]. Available from: http://www.fkl.fi/en/material/publications/Publications/The_reference_number_and_the_check_digit.pdf

APPENDIX A: Query used in the development of the GUIs

1. Query for the art work section in the Customer GUI

```
SELECT Invoices_Receipts_books.*, Art_Works.ArtWork_Name,
Art_Works.YearMade, Art_Works.Purchase_price,
Art_Works.Brokerage, Art_Works.ArtistID, Artist.ArtistLastName,
Artist.ArtistFirstName, Bills_Receipt.CustID,
Art_Works.BrokerageVat, Art_Works.PurchaseVat
FROM Bills_Receipt INNER JOIN (Artist INNER JOIN (Art_Works IN-
NER JOIN Invoices_Receipts_books ON Art_Works.ArtID = Invoic-
es_Receipts_books.ArtID) ON Artist.ArtistID =
Art_Works.ArtistID) ON Bills_Receipt.ReceiptID = Invoic-
es_Receipts_books.ReceiptID;
```

2. *The query for the Va combo box in the art work section in the Customer GUI*

```
SELECT Art_Works.ArtID, Artist.ArtistLastName, Art-
ist.ArtistFirstName, Art_Works.ArtWork_Name,
Art_Works.YearMade, Art_Works.Type_of_Work, Artist.ArtistID
FROM Artist INNER JOIN Art_Works ON Artist.ArtistID =
Art_Works.ArtistID
ORDER BY Artist.ArtistLastName, Art_Works.ArtWork_Name;
```

3. The query for the report for printing the bills and receipt.

```
SELECT Invoices_Receipts_books.*, Bills_Receipt.CustID,
Art_Works.ArtWork_Name, Artist.ArtistLastName, Art-
ist.ArtistFirstName, Art_Works.Purchase_price,
Art_Works.Brokerage, Bills_Receipt.Invoice_Date,
Bills_Receipt.Due_Date, Bills_Receipt.Payment,
Art_Works.PurchaseVat, Art_Works.BrokerageVat
FROM Bills_Receipt INNER JOIN (Artist INNER JOIN (Art_Works IN-
NER JOIN Invoices_Receipts_books ON Art_Works.ArtID = Invoic-
es_Receipts_books.ArtID) ON Artist.ArtistID =
Art_Works.ArtistID) ON Bills_Receipt.ReceiptID = Invoic-
es_Receipts_books.ReceiptID;
```

4. The query for creating the Artist Report GUI

```
SELECT Artist_Report.*, Artist.ArtistFirstName, Artist.ArtistLastName, Artist.ArtistStreetAddress, Artist.ArtistCity, Artist.ArtistCountry, Artist.ArtistZipCode,
Artist.ArtistAccountNumber
FROM Artist INNER JOIN Artist_Report ON Artist.ArtistID = Artist_Report.ArtistID;
```

5. The query for creating the Artist Report Sub form GUI

```
SELECT SettlementWorks.*, Art_Works.ArtWork_Name,
Art_Works.YearMade, Art_Works.Purchase_price,
Art_Works.Brokerage, Art_Works.PurchaseVat,
Art_Works.BrokerageVat, Art_Works.ArtistID,
Art_Works.Artist_paid, Miscellaneous.Cost_of_transport, Miscellaneous.Framing, Art_Works.MiscellaneousID
FROM Miscellaneous INNER JOIN (Artist_Report INNER JOIN
(Art_Works INNER JOIN SettlementWorks ON Art_Works.ArtID = SettlementWorks.WorkID) ON Artist_Report.ArtistReportID = SettlementWorks.SettlementID) ON Miscellaneous.MiscellaneousID =
Art_Works.MiscellaneousID;
```

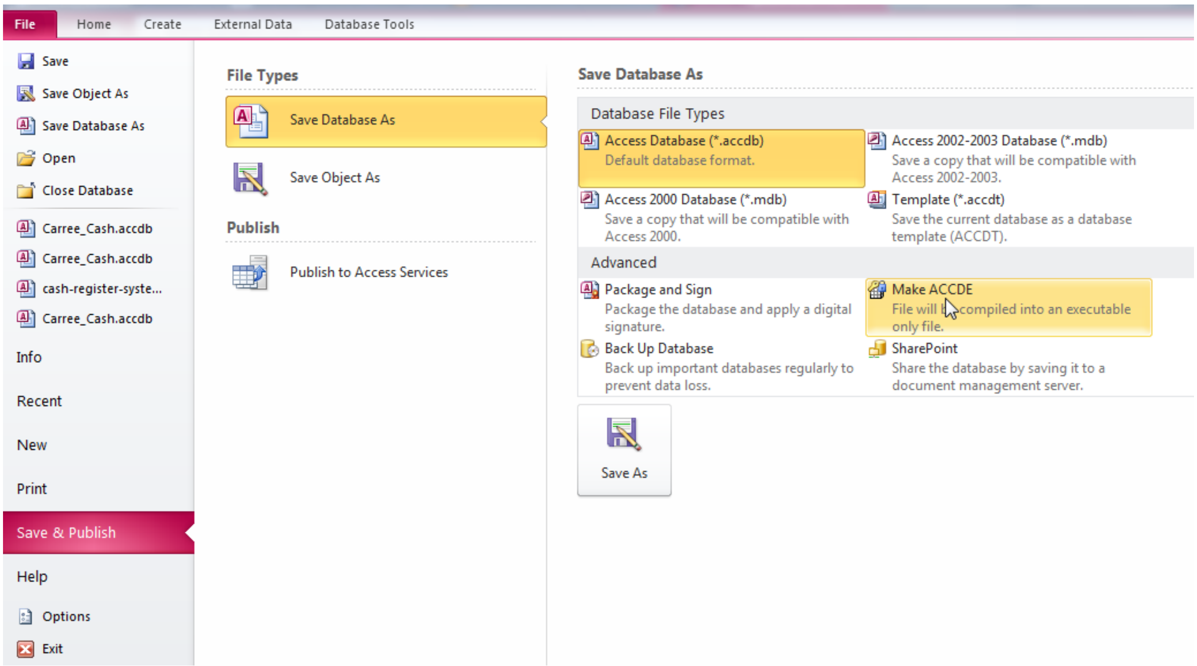
6. Generating a bank reference number for invoicing.

```
Function NewReferenceNumber(InvoiceID)
Dim FormatNumberZero As Integer
Dim InvoiceNumber As Integer
Dim ContractNumber As Variant
Dim ReferenceNumBasic As Variant
Dim CheckSum, i, c  As Variant

InvoiceNumber = FormatNumberZero(InvoiceID, 8)
ReferenceNumBasic = ContractNumber & "0" & InvoiceNumber
CheckSum = 0

For i = 1 To Len(ReferenceNumBasic)

    CurNumber = CVal(Mid(ReferenceNumBasic,
(Len(ReferenceNumBasic) + 1 - i), 1))
If i Mod 3 = 1 Then
    CheckSum = CheckSum + CurNumber * 7
ElseIf i Mod 3 = 2 Then
```

```
    CheckSum = CheckSum + CurNumber * 3
ElseIf i Mod 3 = 0 Then
    CheckSum = CheckSum + CurNumber * 1
End If
Next
    CheckNum = 10 - (CheckSum Mod 10)
    If CheckNum = 10 Then
        CheckNum = 0
    End If
        NewReferenceNumber = ReferenceNumBasic & CheckNum
End Function
```

APPENDIX B. Packaging the Cash register for Testing and Usage

1. Converting the application into an executable only file



Access Database (*.accdb)
Default database format.

Access 2002-2003 Database (*.mdb)
Save a copy that will be compatible with Access 2002-2003.

Access 2000 Database (*.mdb)
Save a copy that will be compatible with Access 2000.

Template (*.accdt)
Save the current database as a database template (ACCDT).

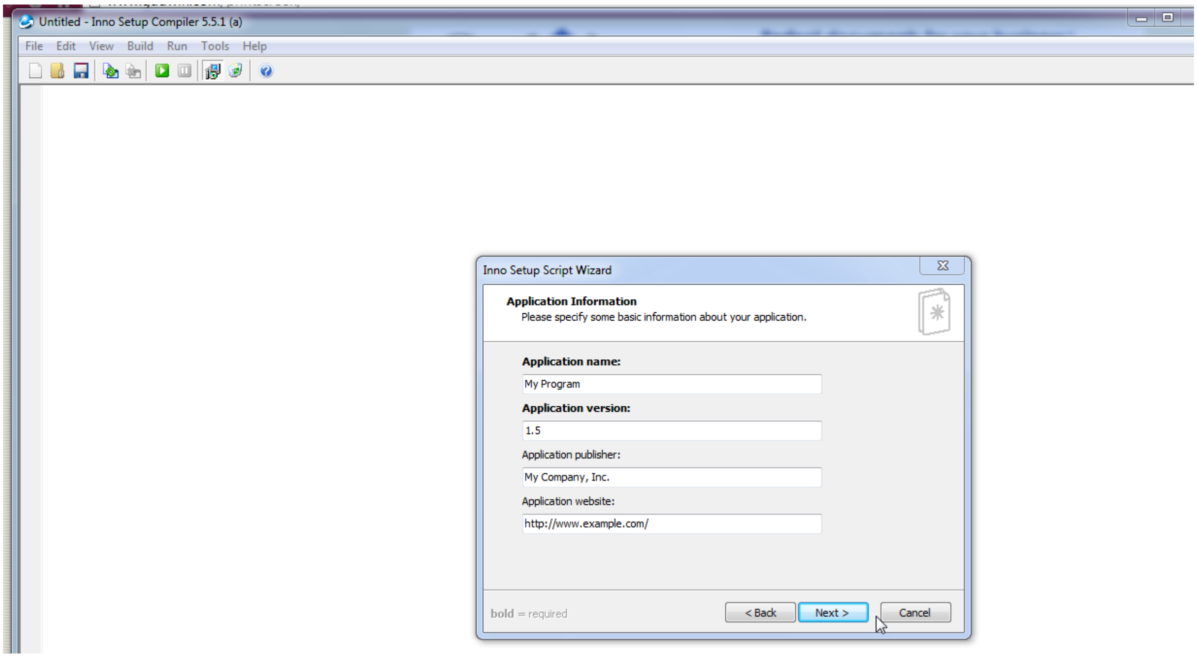
Package and Sign
Package the database and apply a digital signature.

Make ACCDE
File will b compiled into an executable only file.

Back Up Database
Back up important databases regularly to prevent data loss.

SharePoint
Share the database by saving it to a document management server.

2. Using Inno Setup Compiler Wizard to convert it into an installation file



Application Information
Please specify some basic information about your application.

Application name:
My Program

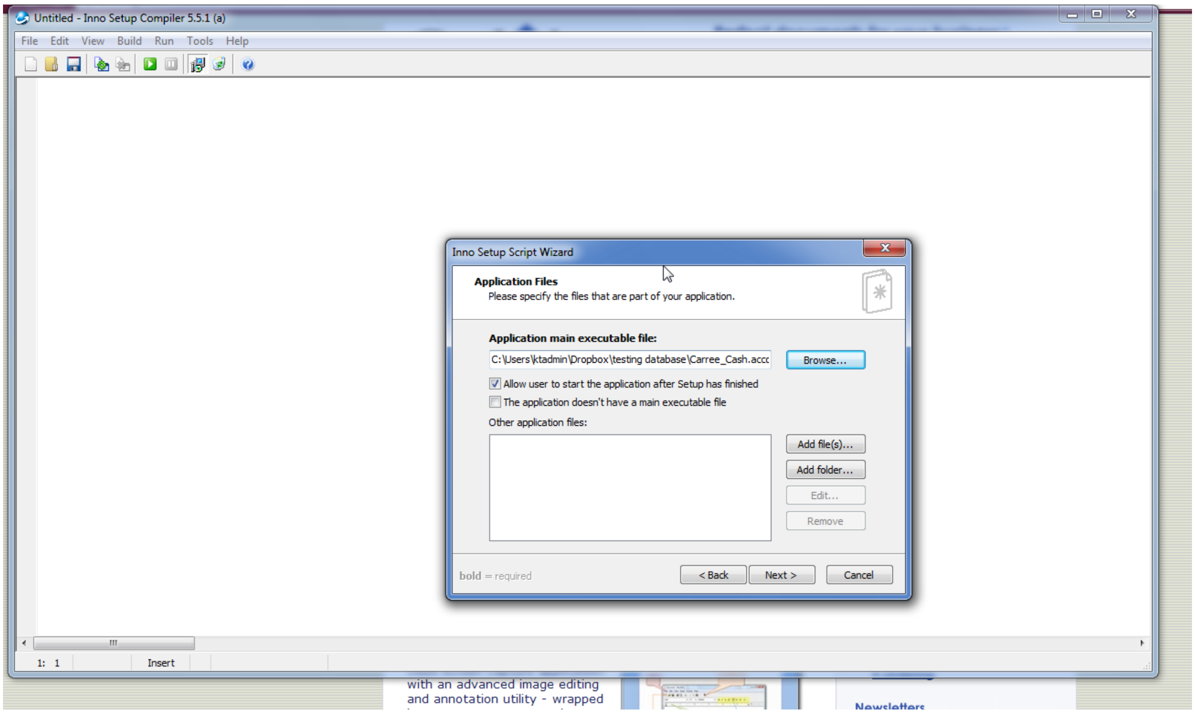
Application version:
1.5

Application publisher:
My Company, Inc.

Application website:
http://www.example.com/

bold = required

3. Inputting the executable only file into the script wizard.



Application Files
Please specify the files that are part of your application.

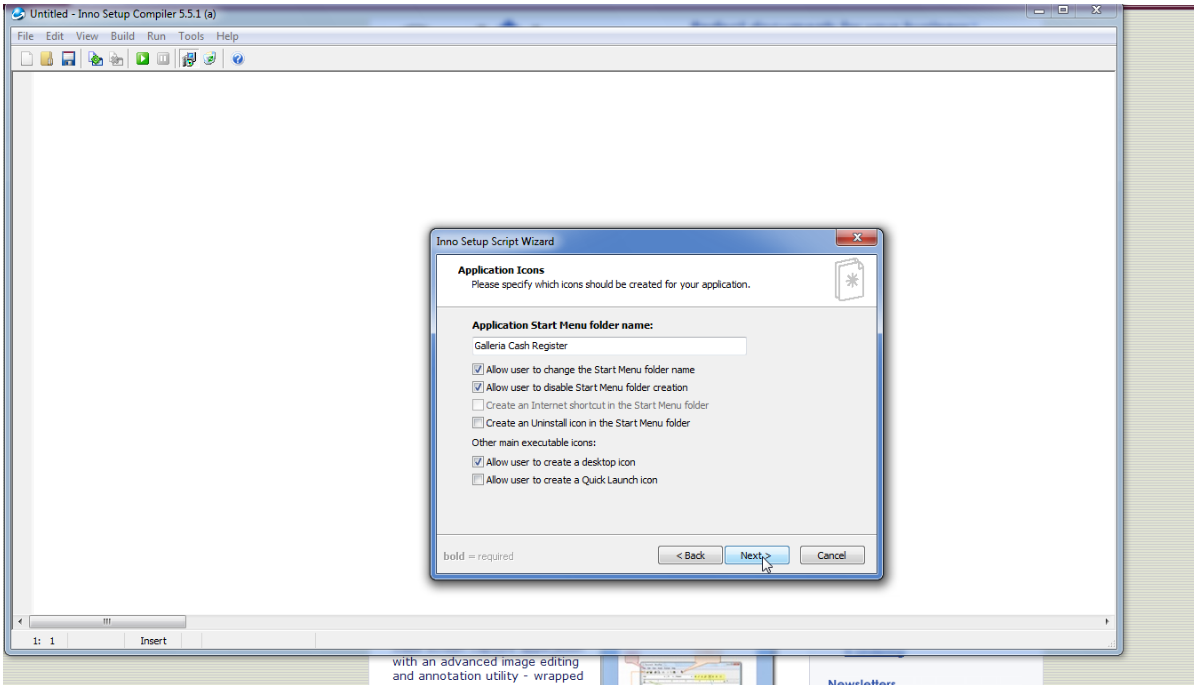
Application main executable file:
C:\Users\ktadmin\Dropbox\testing database\Carree_Cash.accc

☑ Allow user to start the application after Setup has finished
□ The application doesn't have a main executable file

Other application files:

bold = required

4. Creating permission for the user



Untitled - Inno Setup Compiler 5.5.1 (a)
File Edit View Build Run Tools Help
Inno Setup Script Wizard
< Back
Next >
Cancel
1: 1
Insert

Application Icons
Please specify which icons should be created for your application.

Application Start Menu folder name:
Galleria Cash Register

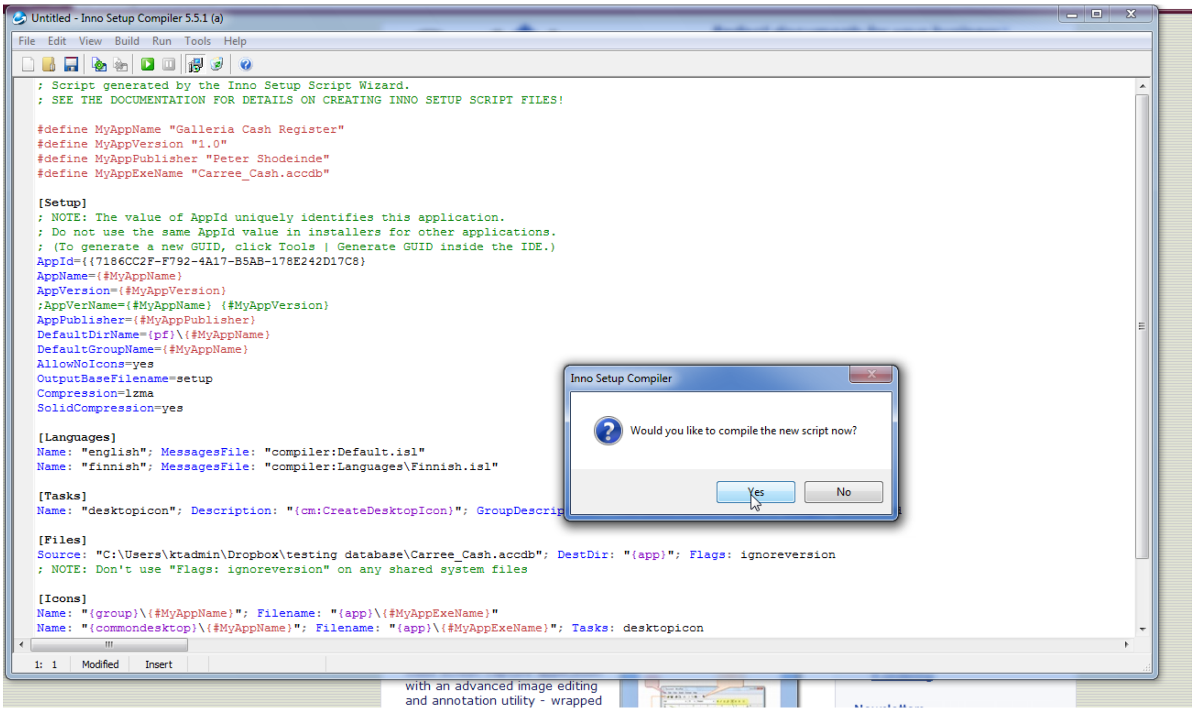
☑ Allow user to change the Start Menu folder name
☑ Allow user to disable Start Menu folder creation
□ Create an Internet shortcut in the Start Menu folder
□ Create an Uninstall icon in the Start Menu folder

Other main executable icons:

☑ Allow user to create a desktop icon
□ Allow user to create a Quick Launch icon

bold = required

5. Compiling the file to create the installable file.


Untitled - Inno Setup Compiler 5.5.1 (a)
File Edit View Build Run Tools Help
1: 1 Modified Insert
with an advanced image editing
and annotation utility - wrapped

```
; Script generated by the Inno Setup Script Wizard.
; SEE THE DOCUMENTATION FOR DETAILS ON CREATING INNO SETUP SCRIPT FILES!

#define MyAppName "Galleria Cash Register"
#define MyAppVersion "1.0"
#define MyAppPublisher "Peter Shodeinde"
#define MyAppExeName "Carree_Cash.accdb"

[Setup]
; NOTE: The value of AppId uniquely identifies this application.
; Do not use the same AppId value in installers for other applications.
; (To generate a new GUID, click Tools | Generate GUID inside the IDE.)
AppId={{7186CC2F-F792-4A17-B5AB-178E242D17C8}
AppName={#MyAppName}
AppVersion={#MyAppVersion}
;AppVerName={#MyAppName} {#MyAppVersion}
AppPublisher={#MyAppPublisher}
DefaultDirName={pf}\{#MyAppName}
DefaultGroupName={#MyAppName}
AllowNoIcons=yes
OutputBaseFilename=setup
Compression=lzma
SolidCompression=yes

[Languages]
Name: "english"; MessagesFile: "compiler:Default.isl"
Name: "finnish"; MessagesFile: "compiler:Languages\Finnish.isl"

[Tasks]
Name: "desktopicon"; Description: "{cm:CreateDesktopIcon}"; GroupDescrip

[Files]
Source: "C:\Users\ktadmin\Dropbox\testing database\Carree_Cash.accdb"; DestDir: "{app}"; Flags: ignoreversion
; NOTE: Don't use "Flags: ignoreversion" on any shared system files

[Icons]
Name: "{group}\{#MyAppName}"; Filename: "{app}\{#MyAppExeName}"
Name: "{commondesktop}\{#MyAppName}"; Filename: "{app}\{#MyAppExeName}"; Tasks: desktopicon
```

Inno Setup Compiler

Would you like to compile the new script now?