LottoSYS

Submitted by: Darren Moriarty

T00168162

Software Engineering – Stage 2

Date Submitted: 02 / 05 / 2017

**Table of Contents**

[1. Introduction/overview 4](#_Toc469051960)

[2. Functional Components 5](#_Toc469051961)

[3. User Requirements 6](#_Toc469051962)

[3.1. LottoSYS Performs Customer Administration 6](#_Toc469051963)

[3.2. LottoSYS will allow for sale transactions 6](#_Toc469051964)

[3.3. LottoSYS will manage prizes and winnings 6](#_Toc469051965)

[4. System Requirements 7](#_Toc469051966)

[4.1. System Level Use Case Diagram 7](#_Toc469051967)

[4.2. Manage Customers 8](#_Toc469051968)

[4.2.1. New Customer 8](#_Toc469051969)

[4.2.2. Update Customer 12](#_Toc469051970)

[4.2.3. De-register Customer 16](#_Toc469051971)

[4.2.4. List Customers 19](#_Toc469051972)

[4.2.5. Customer Profile 22](#_Toc469051973)

[4.3. Process Sales 25](#_Toc469051974)

[4.3.1. Sell Ticket 25](#_Toc469051975)

[4.3.2. Sales Analysis 31](#_Toc469051976)

[4.4. Manage Prizes 33](#_Toc469051977)

[4.4.1. Run Draw 33](#_Toc469051978)

[4.4.2 Check Tickets 36](#_Toc469051979)

[4.4.3 Lotto Number Analysis 39](#_Toc469051980)

[5. System Model 41](#_Toc469051981)

[5.1. Level-0 DFD 41](#_Toc469051982)

[5.2. Level-1 DFD 42](#_Toc469051983)

[5.3. Level-2 DFD (Process P1: Manage Customer) 43](#_Toc469051984)

[5.4. Level-2 DFD (Process P2: Process Sales) 44](#_Toc469051985)

[5.5. Level-2 DFD (Process P3: Manage Prizes) 45](#_Toc469051986)

[6. Data Model (Class Diagram) 46](#_Toc469051987)

[6.1. Class Diagram 46](#_Toc469051988)

[6.2. Relational Schema 47](#_Toc469051989)

[6.3. Database Schema 51](#_Toc469051990)

[7. Conclusion 52](#_Toc469051991)

[8. Appendices 53](#_Toc469051992)

[8.1. Appendix A – Title 53](#_Toc469051993)

[8.2. Appendix B – Title 53](#_Toc469051994)

# Introduction/overview

LottoSYS is a software application designed to be used by a lottery organisation. The system is designed to be used by the lottery administrator

The system will allow new customers to be added to the system, updating of existing customer details, deregistration of existing customers, customer details listings and generation of customer profiles.

The system will allow for tickets to be sold to customers and for an analysis of the ticket sales.

The Manage Prizes section will run the weekly system draw, this draw will generate 6 numbers, the system will check customer’s tickets to see if a prize has been won. The system will issue prizes and generate a lotto number analysis.

The LottoSYS will be a simple and as user friendly as possible so that any system administrator will be able to navigate the system intuitively.

# Functional Components

# User Requirements

The following user requirements have been outlined for the system (LottoSys):

## LottoSYS Performs Customer Administration

* + 1. *LottoSYS will register a new customer*
    2. *LottoSYS will allow customer details to be updated*
    3. *LottoSYS will allow customer details to be withdrawn*
    4. *LottoSYS will list customer details*
    5. *LottoSYS will generate a customer’s profile upon request*

## LottoSYS will allow for sale transactions

* + 1. *LottoSYS will allow tickets to be sold*
    2. *LottoSYS will generate a sales analysis*

## LottoSYS will manage prizes and winnings

* + 1. *LottoSYS will run a lotto draw weekly*
    2. *LottoSYS will inform winners and pay prize*
    3. *LottoSYS will show Winning numbers analysis and Statistics*

# System Requirements

The system requirements relating to the user requirements identified in section 3 are defined below:

## System Level Use Case Diagram

This section describes the LottoSYS system at a system level

Lotto SYS

Administrator

Customer

## Manage Customers

The system has a number of requirements in relation to the management of customer data. The registration of new customers, the updating of customer details, deregistration of a customer, a list of all the customers on the system and generating a customer profile.

### New Customer

This function registers a new customer on the system.

Customer

Admin

<< Include >>

<< Extends >>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **New Customer** | |
| **Use Case Id** | 1.1 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | This function registers a new customer’s details in the system. | |
| **Preconditions** | The customer must complete and sign the appropriate registration form. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the create New Customer function  **Step 8:** The administrator enters the following required details:   * Title * Surname * Forename * DOB * PPSN * Address Line 1 * Address Line 2 * Town * County * Country * Country of Residence * Nationality * Gender * Phone * E-mail   **Step 9:** The administrator confirms customer is to be registered | **Step 2:** The system retrieves details of customer titles from the **Titles File** and loads on UI  **Step 3:** The system retrieves details of customer nationalities from the **Nationality File** and loads on UI  **Step 4:** The system retrieves details of customer gender from the **Gender File** and loads on UI  **Step 5:** The system retrieves details of customer county from the **County File** and loads on UI  **Step 6:** The system retrieves details of customer country from the **Country File** and loads on UI  **Step 7:** The system displays the UI and prompts for the customer’s surname  **Step 10:** The system validates the customer details:   * PPSN must not be already registered * Email address must not be already registered * All fields except for Gender and Contact Phone no. must be entered * DOB must not be in the future * Customer must be over the age of 18 * Name must not have numeric characters   **Step 11:** The system generates and assigns a unique customer identification number  **Step 12:** The current date/time stamp is assigned as the registration date  **Step 13:** The system assigns the default value of ‘Active’ as the customer status  **Step 14:** Initialise the customer’s account balance to zero  **Step 15:** The system saves the customer details in the **Customer File**  **Step 16:** The system displays a confirmation message  **Step 17:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Customer under 18 years of age** | Administrator | **Step 9:** Business rule broken  **Step 10:** The system displays an error message with choice to either re-enter details at step 8 or cancel registration |
| **Invalid Data Entered** |  | **Step 9:** Data entered incorrectly / invalid data  **Step 10:** The system displays an appropriate error message and returns to step 8 |
| **Conclusions** | Customer is registered and may now purchase lotto ticket(s) | |
| **Post conditions** |  | |
| **Business Rules** | Customer must be over 18 years old | |
| **Implementation Constraints** |  | |

### Update Customer

This function updates an existing customer on the system.

Admin

<< Include >>

<< Extends >>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Customer** | |
| **Use Case Id** | 1.2 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | This function updates an existing customer’s details in the system. | |
| **Preconditions** | Customer must provide deed pole documentation if they wish to update their name | |
| **Trigger** | This use case is invoked when a customer wants to update their existing account | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the Update Customer function  **Step 8:** The administrator enters the customers surname  **Step 10:** The administrator selects the customer to update  **Step 11:** The administrator changes the customer details:   * Title * Surname * Forename * DOB * PPSN * Address Line 1 * Address Line 2 * Town * County * Country * Nationality * Country of Residence * Gender * Phone * E-mail   **Step 12:** The administrator confirms the data for the customer that is to be updated | **Step 2:** The system retrieves details of customer titles from the **Titles File** and loads on UI  **Step 3:** The system retrieves details of customer nationalities from the **Nationality File** and loads on UI  **Step 4:** The system retrieves details of customer gender from the **Gender File** and loads on UI  **Step 5:** The system retrieves details of customer county from the **County File** and loads on UI  **Step 6:** The system retrieves details of customer country from the **Country File** and loads on UI  **Step 7:** The system displays the UI and prompts for the customer’s surname  **Step 9:** The system retrieves a summary of customers from the **Customer File** for the surname entered with the status of ‘*Active*’ and displays on UI  **Step 13:** The system validates the customer details:   * Customer must be over the age of 18 * Name must not have any numeric characters * PPSN must not be already registered * Email address must not be already registered * All fields except for Gender, Contact Phone must be entered * DOB must not be in the future   **Step 14:** The system updates the updated customer details in the **Customer File**  **Step 15:** The system displays a confirmation message  **Step 16:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | **Step 14:** Data entered incorrectly / invalid data  **Step 15:** The system displays an appropriate error message and returns to step 12 |
| **Conclusions** | Customer’s details have been updated |  |
| **Post conditions** |  | |
| **Business Rules** | The customer must be already registered and have a status of ‘*Active*’ to update their profile | |
| **Implementation Constraints** |  | |

### De-register Customer

This function De – Registers a customer from the system

Customer

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **De-register Customer** | |
| **Use Case Id** | 1.3 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | This function changes the customer’s status from ‘Active’ to ‘De-Registered’ | |
| **Preconditions** | The customer must complete and sign the appropriate de-registration form.  A proof of death certificate must be provided in case of deceased person | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the De-Register Customer function  **Step 3:** The administrator enters the customers surname  **Step 5:** Administrator selects the customer to de-register  **Step 7:** The administrator selects the appropriate radio button | **Step 2:** The system loads the UI  **Step 4:** The system retrieves a summary of customer details from the **Customer File** for the surname entered with the status of ‘*Active*’ and displays on UI  **Step 6:** The system retrieves allthe relevantdetails for selected customer from **Customer File** and displays on UI for viewing only  **Step 8:** The system updates the customer’s status to ‘de-registered’ in the **Customer File**  **Step 9:**  The system saves the customer’s customerID and the de-registration date to the **De-Reg File**  **Step 10:** The system saves the date that the customer is de-registered in the **De-Reg File**  **Step 11:** The system displays a confirmation message  **Step 12:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Neither of radio buttons selected** |  | **Step 9:** Both radio buttons were left unselected  **Step 6:** return to step 8 |
| **Conclusions** | Customer is de-registered and my no longer hold an account | |
| **Post conditions** |  | |
| **Business Rules** | Only customers with the status of ‘Active’ may be de-registered | |
| **Implementation Constraints** |  | |

### List Customers

This function generates a listing of active customers

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **List Customers** | |
| **Use Case Id** | 1.4 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | This function lists all the customers in the system | |
| **Preconditions** |  | |
| **Trigger** | The administrator wants to list all customers in the system | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the List Customers function  **Step 3:** The administrator enters the customers surname  **Step 5:** The administrator selects the desired a filter option from the following categories   * All * Winners * Withdrawn   **Step 8:** Administrator has the option to print the specified list | **Step 2:** The system loads the UI  **Step 4:** The system retrieves a summary active customers from the **Customer File** ordered by CustomerID in ascending order  **Step 6:** The system retrieves the tickets from the **Ticket File** for the customers that have won a prize  **Step 7:** System filters the customer listings accordingly  **Step 9:** The system creates a print file with the selected data and sends it to specified printer  **Step 10:** The system displays a confirmation message after print is complete  **Step 11:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Conclusions** | Customer files are listed | |
| **Post conditions** | No edit can be performed on customer data | |
| **Business Rules** | Customer files may only be viewed from this function, neither altered or deleted | |
| **Implementation Constraints** |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| Customers sorted by CustomerID | | | |
| ID | Forename | Surname | Balance |
| 56 | John | Doe | €500 |
| 93 | John | Smith | €2,500 |
| 667 | Anne | Shea | €1 |

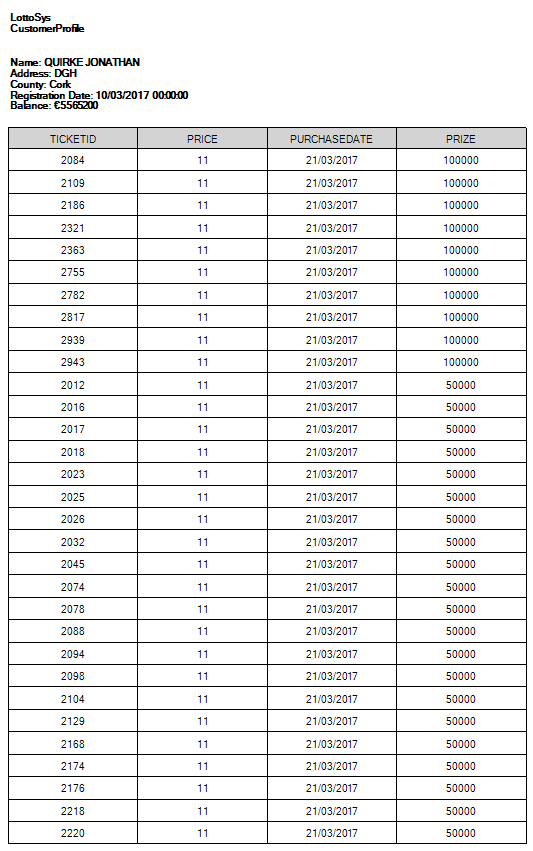
### Customer Profile

This function creates a printable file with a full customer profile

Customer

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Customer Profile** | |
| **Use Case Id** | 1.5 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | This function creates a printable file with a full customer profile | |
| **Preconditions** |  | |
| **Trigger** | This use case is invoked when a customer wants all their account details | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the Customer Profile function  **Step 3:** The administrator enters the customer’s surname  **Step 5:** The administrator selects the required customer  **Step 7:** The administrator has the option to print the customer’s details | **Step 2:** The system displays the UI  **Step 4:** The system retrieves a summary of customer details from the **Customer File** with matching surnames  **Step 6:** The system retrieves the details for the selected customer from the **Customer File, Prizes File** and the **Tickets File** and displays the UI  **Step 8:** The system creates a printable file of the customer’s details and sends to printer  **Step 9:** The system displays a confirmation message once the file has been printed  **Step 10:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Conclusions** | Customer’s details have been printed |  |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |



## Process Sales

The system has a number of requirements in relation to the processing of ticket sales. The selling of tickets, and the analysis of the ticket sales.

### Sell Ticket

This function sells a customer tickets. The customer can choose between 1 to 5 panels per ticket. A purchased ticket is only active in one draw, the draw subsequent the purchase date. After the draw the ticket is no longer valid. Tickets cannot be purchased after 5 o clock on a Tuesday which is the day of the draw

Customer

Admin

<< Include >>

|  |
| --- |
| **Activity Diagram: Sell Ticket** |

|  |  |
| --- | --- |
| Administrator | System |

Display UI

Invoke Sell Ticket

.

Enter Surname

Retrieve all active members with surname

Select Member

Retrieve selected member details

.

Generate Lines

Select Line Amount

.

More Lines?

Y

N

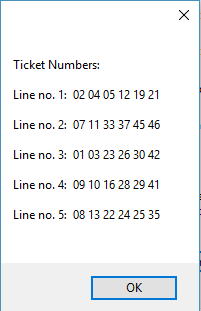
Display Confirmation Message

Save Ticket Details to Tickets File

Set Purchase Date

Save panels to Panel File

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Sell Ticket** | |
| **Use Case Id** | 2.1 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** | Customer | |
| **Description** | This function sells a customer tickets, each new tickets’ prizeFlag is set to ‘NO’ | |
| **Preconditions** | The customer must be a registered customer and have a status of active | |
| **Trigger** | This use case is invoked when a customer wants to purchase a ticket | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the Sell Tickets function  **Step 3:** The administrator enters the customer’s surname  **Step 5:** The administrator selects the correct customer from the listings  **Step 7:** The administrator selects the amount of panels the customer decides to purchase | **Step 2:** The system displays the UI and prompts customer details  **Step 4:** The system retrieves a summary of customer details from the **Customer File**  **Step 6:** The system retrieves the customer’s details from the **Customer File** and prompts for the amount of panels to be printed on the ticket being sold  **Step 8:** The system generates a ticket with the unique TicketID, the customer’s CustomerID, purchase date set to the system date and saves the details to the **Tickets File**  **Step 9:** The system saves the total ticket price to the **Tickets File**,  **Step 10:** The systemsets the prize flag to ‘No’ for the current ticket being saved in the **Tickets File**.  **Step 11:** The systemsaves the details to the **Tickets File**  **Step 12:** The system saves the panel numbers, the generated PanelId and the TicketId to the **Panels File**  **Step 13:** The system displays a confirmation message with the ticket numbers  **Step 14:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | Customer has purchased ticket(s) | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

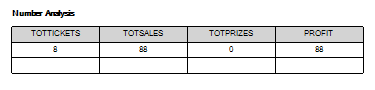


### Sales Analysis

This function gives a sales analysis

Administrator

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Sales Analysis** | |
| **Use Case Id** | 2.2 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | This function gives a sales analysis | |
| **Preconditions** | The Administrator must enter the correct Username and Password to use this function | |
| **Trigger** | This use case is invoked when the administrator invokes the sales analysis function | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the Sales Analysis function  **Step 3:** The administrator enters the username and password  **Step 5:** The administrator picks a start date for the analysis  **Step 6:** The manager picks an end date for the analysis  **Step 9:** The administrator selects the print button | **Step 2:** The system displays the UI and prompts for username and password  **Step 4:** The system validates the username and password  **Step 7:** The system retrieves the total of all ticket sales and total prizes between the two dates from the **Tickets File** and the **Prizes File**  **Step 8:** The system generates a sales analysis on the retrieved data and loads the UI  **Step 10:** The system creates a print file with the listed data and sends it to specified printer  **Step 11:** The system displays a confirmation message after print is complete  **Step 12:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Username and or password incorrect** | **manager** | **Step 4:** Username and or password incorrect  **Step 5:** return to step 3 |
| **Conclusions** |  | |
| **Post conditions** |  | |
| **Business Rules** | These file are read only | |
| **Implementation Constraints** |  | |



## Manage Prizes

The system has a number of requirements in relation to the management of prizes. The execution of the weekly lotto draw, the checking of tickets and the analysis of draw numbers.

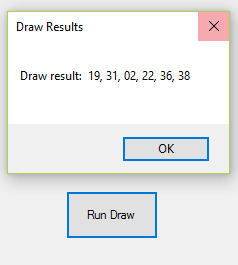
### Run Draw

This function Runs the systems weekly draw. The draw is run every Tuesday at 19:00. The system will generate six random numbers in the range of 1 to 47.

Admin

<< Include >>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Run Draw** | |
| **Use Case Id** | 3.1 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | This function Runs the system draw | |
| **Preconditions** |  | |
| **Trigger** | The administrator invokes the Run Draw function | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the Run Draw function  **Step 3:** The administrator selects the run draw button | **Step 4:** The System generates six random numbers between 1 and 47  **Step 5:** The system validates the numbers, confirming there are no duplicate numbers generated in the draw  **Step 6:** The system saves the winning numbers and the draw date in the **Draw File.**  **Step 7:** The system updates the **Number Analysis File** with the draw number occurrences  **Step 8:** The system displays the draw numbers on the UI  **Step 9:** The system displays a confirmation message and disables the run draw button for seven days  **Step 10:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Conclusions** | Lotto numbers have been saved to the **Draw File** | |
| **Post conditions** | Winnings can be collected if there are winners  Draw cannot be run for another seven days | |
| **Business Rules** |  | |
| **Implementation Constraints** | This function can only be run on the day of the draw which is Tuesday after 17:00 | |



### 4.4.2 Check Tickets

This function checks tickets entered in the most recently run draw and identifies those tickets which have won a prize. A ticket wins a prize if there are 6, 5, 4 or 3 matching numbers on the ticket. The prize amounts are fixed and are shown in the table below.

|  |  |  |
| --- | --- | --- |
| Prize structure | | |
| 1 | Match 6 | €1,000,000 |
| 2 | Match 5 | €500,000 |
| 3 | Match 4 | €100,000 |
| 4 | Match 3 | €10,000 |

Admin

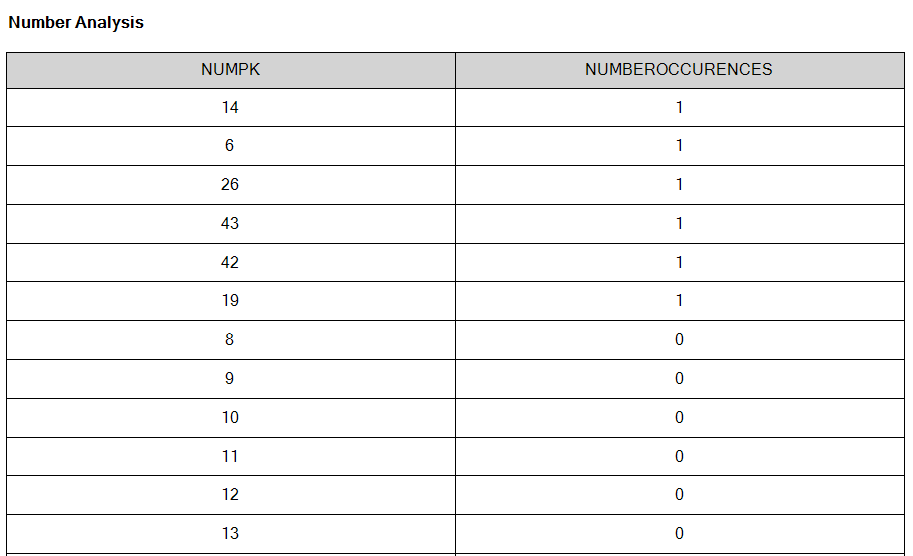
|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check Tickets** | |
| **Use Case Id** | 4.2 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | This function checks for winning tickets for the previously run draw | |
| **Preconditions** |  | |
| **Trigger** | The administrator invokes the Check Tickets function | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the Check Tickets function  **Step 11:** After all the tickets have been checked, the administrator can sort the winning tickets  **Step 13:** The administrator can choose to print the list of winning tickets | **Step 2:** The System retrieves a list of the tickets from the **Tickets File** that were purchased within seven days’ prior of the current drawand displays the UI  **Step 3:** The system retrieves the corresponding panels from the **Panels File**  **Step 4:** The system loops through each panel to compare the tickets numbers with the draw numbers  **Step 5:** The system detects a winning ticket and retrieves the corresponding prize amount from the **Prize Structure File**  **Step 6:** The system updates the customer’s balance with the prize amount in the **Customer File**  **Step 7:** The system sets the winning tickets prizeFlag to ‘Yes’ in the **Tickets File**  **Step 8:**  The system saves the TicketID, prize mount and the draw date in the **Prizes File**  **Step 9:** The system e-mails the customer to confirm the payment has been processed  **Step 10:** If more tickets are to be checked return to step 4  **Step 12:** The system creates a print file and sends it to the local printer  **Step 14:** The system displays a confirmation message  **Step 15:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** |  | |
| **Post conditions** | Each ticket that has been checked is no longer eligible for entry in future draws | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### 4.4.3 Lotto Number Analysis

This function shows a listing draw number analysis

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Lotto Number Analysis** | |
| **Use Case Id** | 3.4 | |
| **Priority** | High | |
| **Source** | Administrator | |
| **Primary Business Actor** | Administrator | |
| **Other Participating Actors** |  | |
| **Description** | This function shows a listing draw number analysis | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The administrator starts the Lotto Number Analysis function  **Step 3:** The administrator can select to print the details | **Step 2:** The system lists the most frequently occurring numbers along with the least frequently occurring numbers in list boxes from the **Number Analysis File** and loads the UI  **Step 4:** The system creates a print file and sends it to the selected printer  **Step 5:** The system displays a confirmation message  **Step 6:** The UI is cleared and reset |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Conclusions** |  | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |



# System Model

*The following dataflow diagrams have been produced for the system*:

## Level-0 DFD

Customer

LottoSYS

Purchase Request

D6

De-Reg File

## Level-1 DFD

Customer Information

Titles Dropdown

De-Reg Information

Nationality Dropdown Dropdown

D1

Titles File

Gender Dropdown

Customer

Customer Profile

D2

Nationality File

P1

Manage Customers

Customer Prizes

County Dropdown

D3

Gender File

D9

Prizes File

Country Dropdown

D4

County File

Customer details

D5

Country File

Customer Tickets

D7

Customer File

Customer details

D9

Prizes File

Ticket details

D8

Tickets File

P2

Process Sales

Customer

Panel Numbers

Customer Balance

D7

Customer File

Draw Analysis

D13

NumberAnalysis

Panel Numbers

Prize Flag

Ticket

Draw Numbers

D12

PrizeStructure File

Prize structure

D11

Draw File

Prize Amounts

D9

Prizes File

P3

Manage Prizes

Ticket Request

D10

Panels File

## Level-2 DFD (Process P1: Manage Customer)

Titles Dropdown

Titles Dropdown

Nationality Dropdown

Nationality Dropdown

D1

Titles File

P1.1

New Customer

D5

Country File

D2

Nationality File

D3

Gender File

D4

County File

Gender Dropdown

Gender Dropdown

P1.2

Update Customer

County Dropdown

County Dropdown

Country Dropdown

Country Dropdown

Existing Customer Details

D7

Customer File

Updated Customer Details

New Customer Details

Customer’s Tickets

Customer’s Prizes

Customer Details

D9

Prizes File

De-Registered Account

D6

De-Reg File

D8

Tickets File

Customer Details

De-Registered Account

Customer Details

De-Register Request

Profile Request

Customer Profile

New Customer Details

Updated Customer Details

P1.4

List Customer

D7

Customer File

P1.3

De-Register Customer

Customer

P1.5

Customer Profile

## Level-2 DFD (Process P2: Process Sales)

Panel Numbers

D10

Panels File

Customer Details

D7

Customer File

P2.2

Sales Analysis

Ticket Request

Sold Ticket Details

Sold Ticket Details

Customer

D8

Tickets File

P2.1

Sell Ticket

Prize Amounts

D9

Prizes File

Ticket

## Level-2 DFD (Process P3: Manage Prizes)

Draw Numbers

Draw Numbers

D13

NumberAnalysis

Draw Numbers

Draw Date



Updated Panel

Panel Numbers

D10

Panels File

Updated Ticket

Update Balance

D7

Customer File

D8

Tickets File

Winning Ticket

D9

Prizes File

P3.3

Lotto Number Analysis

Prize Amount

Customer ID

Draw Numbers

D12

PrizeStructure File

P3.2

Check Tickets

D11

Draw File

P3.1

Run Draw

# Data Model (Class Diagram)

In the diagram below, the UML class diagrams have been outlined. It shows the relationships between the tables. The database consists of 13 tables, this includes the weak entity table.

## Class Diagram

## Database Schema

**Relation:** Customer

Attributes:

CustomerID numeric (6),

Title char(4) NOT NULL,

Surname char(25) NOT NULL,

Forename char(25) NOT NULL,

DOB date NOT NULL,

PPSN char(8) NOT NULL,

AddressLine1 varchar(40) NOT NULL,

AddressLine2 varchar(40) NOT NULL,

Town varchar(40) NOT NULL,

County varchar(35) NOT NULL,

Country varchar(35) NOT NULL,

Nationality varchar(40) NOT NULL,

Gender char(6),

Phone char(15),

Email char(40) NOT NULL,

Balance decimal(10,2) DEFUALT 0,

Customer-Status char(10) NOT NULL,

Reg-Date date NOT NULL

**Primary Key:** UserID

**Relation:** Ticket

Attributes:

TicketID char(12) ,

CustomerId char(6) NOT NULL,

PurchaseDate date NOT NULL,

Time date/time NOT NULL,

Price decimal(4,2) NOT NULL,

PrizeFlag char(4) NOT NULL

**Primary Key:** TicketID

**Foreign Key:** CustomerID REFERENCES Customer

**Relation:** Panel

Attributes:

PanelID char(12) NOT NULL,

TicketID char(12) NOT NULL,

Number1 numeric(2) NOT NULL,

Number2 numeric(2) NOT NULL,

Number3 numeric(2) NOT NULL,

Number4 numeric(2) NOT NULL,

Number5 numeric(2) NOT NULL,

Number6 numeric(2) NOT NULL,

**Primary Key:** PanelID, TicketID

**Foreign Key:** TicketID REFERENCES Ticket

**Relation:** Draw

Attributes:

DrawDate date NOT NULL,

Number1 numeric (2) NOT NULL,

Number2 numeric (2) NOT NULL,

Number3 numeric (2) NOT NULL,

Number4 numeric (2) NOT NULL,

Number5 numeric (2) NOT NULL,

Number6 numeric (2) NOT NULL

**Primary Key:** DrawDate

**Relation:** PrizeStructure

Attributes:

DrawDate date NOT NULL,

Match6 numeric(7) NOT NULL,

Match5 numeric(6) NOT NULL,

Match4 numeric(5) NOT NULL,

Match3 numeric(4) NOT NULL

**Primary Key:** DrawDate

**Relation:** De-Reg

Attributes:

CustomerID numeric (6),

De-RegDate date NOT NULL

**Primary Key:** CustomerID

**Foreign Key:** CustomerID REFERENCES Customer

**Relation:** Prizes

Attributes:

DrawDate date NOT NULL,

TicketID char(12) NOT NULL,

PanelID char(12) NOT NULL,

PrizeAmount numeric(7) NOT NULL

**Primary Key:** TicketID, PanelID

**Foreign Key:** TicketID REFERENCES Ticket

**Relation:** NumberAnalysis

Attributes:

NumPk numeric(2) ,

NumberOccurences numeric(5) DEFAULT 0

**Primary Key:** NumPk

**Relation:** Title

Attributes:

Titles char(5) NOT NULL,

**Primary Key:** Titles

**Relation:** Gender

Attributes:

Genders char(6) NOT NULL,

**Primary Key:** Genders

**Relation:** Nationality

Attributes:

Nationalities varchar(50) NOT NULL,

**Primary Key:** Nationalities

**Relation:** County

Attributes:

Counties varchar(25) NOT NULL,

**Primary Key:** Counties

**Relation:** Country

Attributes:

Countries varchar(40) NOT NULL,

**Primary Key:** Countries

## Relational Schema

Customer(CustomerId, Title, Surname, Forename, DOB, PPSN, AddressLine1, AddressLine2,

Town, County, Country, Nationality, Gender, Phone, Email, Balance,

CustomerStatus, Reg-Date)

Ticket(TicketId, CustomerId, PurchaseDate, Time, Price, PrizeFlag)

Panel(PanelId, TicketId, Number1, Number2, Number3, Number4, Number5, Number6)

Draw(DrawDate, Number1, Number2, Number3, Number4, Number5, Number6)

PrizeStructure(DrawDate, Match6, Match5, Match4, Match3)

De-Reg(CustomerId, De-RegDate)

Prizes(TicketId, PanelID, PrizeAmount, DrawDate)

NumberAnalysis(NumPk, NumberOccurances)

Title(Titles)

Gender(Genders)

Nationality(Nationalities)

**County**(Counties)

**Country**(Countries)

# Conclusion

**Aim**

The aim of this project is to design a functional Lottery system.

**Conclusion**

In this document is the design for a computerised lottery system. The key aspects of the lottery system are in the functionality implemented to allow managing customers, purchasing tickets and the processing of prizes. This is a fully functional piece of software that has been planned very carefully and implemented with rigorous precision. By following the specified implementation, any programmer should have the required information to design this piece of software.

# Appendices

## Appendix A – Title

## Appendix B – Title

Might include:

* **Lookup / Reference tables**
* **Sample reports / Listings**