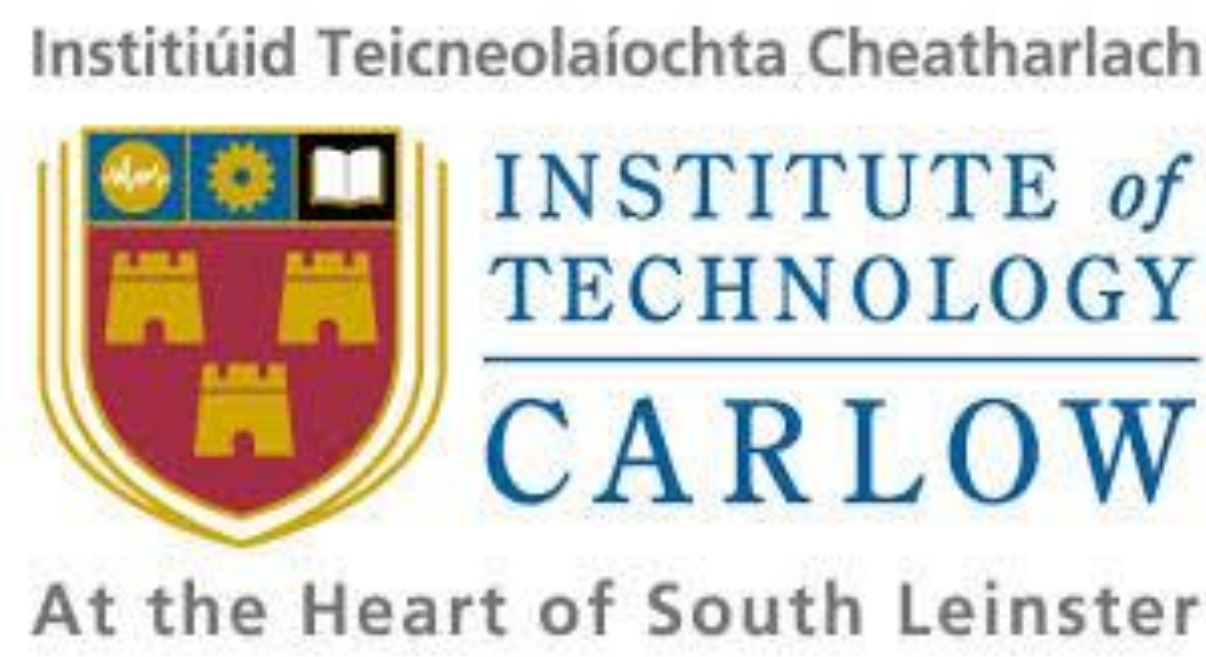

Final Year Project

Design Document

Project Title: ClickNWin

Student: Geoffrey Atkinson

Student Number: C00184861

Project Supervisor: Greg Doyle

Date: 05/04/2017

Table of Contents

1.0 Introduction

This document is the design documentation for the ClickNWin project. The design document will contain the high and low level design elements for the application. There will be a listing of the project's detailed use cases giving a low level look at the application's functionality. There will also be a description of the UI for the web application which will include screenshots of potential designs for the different pages required in the application and the navigation sequence for the site.

2.0 Architecture

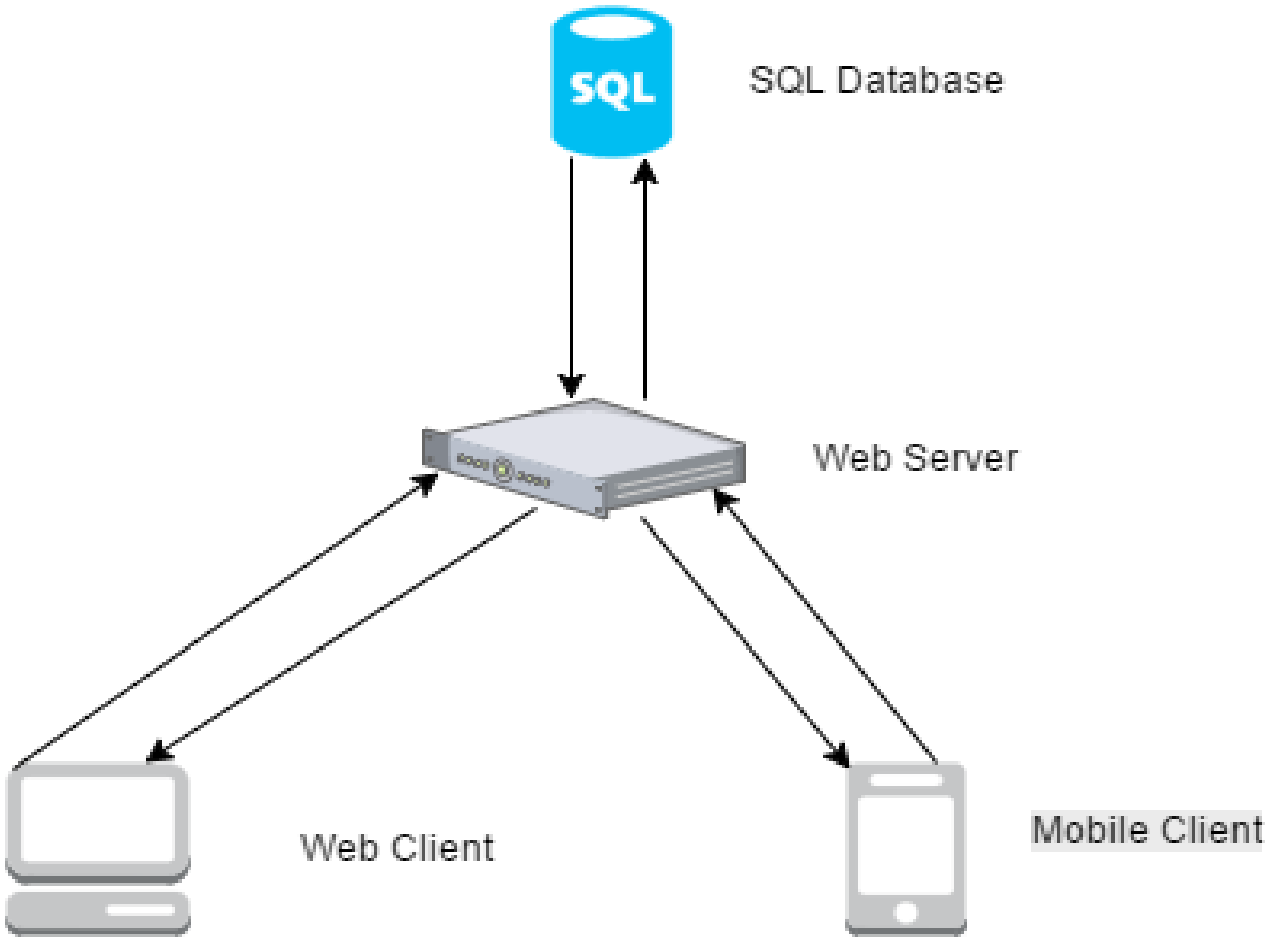

Fig 1. System Architecture

3.0 Database Model

By its nature, ClickNWin will have to store data. The database to be used by ClickNWin will be MySQL. The database will store user data, admin data, scratch cards, payment cards and the information on the different types of scratch cards. Other types of data may be required as work on the application progresses and these will be documented as appropriate. Attached below is the current working model of what the ClickNWin database will look like. Items with PK beside them are the table's primary key. Items with FK are foreign keys that link to another table.

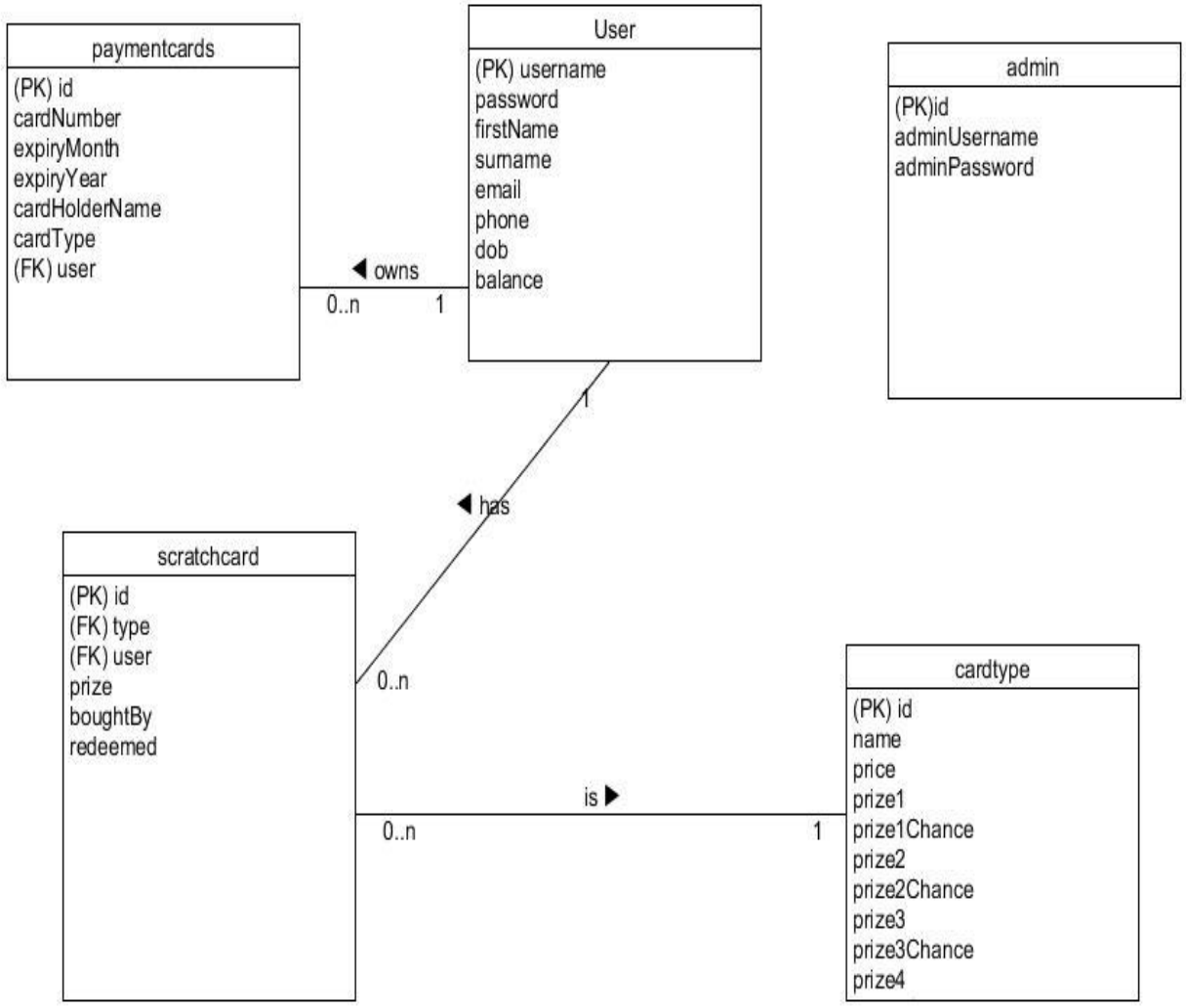

Fig 2. ClickNWin database model

4.0 Detailed Use Cases

4.1 Log In

Actors: Player

Description: This use case begins when a player wants to log into the web or mobile application.

Main Success Scenario

1. The player arrives at the application homepage.
2. The player selects the log in option on the homepage.
3. The system displays the login form.
4. The player inputs and submits their username and password.
5. The system validates the login and returns the login success screen.

Alternatives

4a. The player inputs an incorrect username or password.

1. The system displays an error message informing the user they have incorrectly entered their details.
2. The player is returned to step three.

4.2 Process Login

Actors: System

Description: This use case begins when the client sends a login in validation request to the server.

Main Success Scenario

1. The client receives a login request from a player which contains a user name and password.
2. The client sends the username and password to the server.
3. The server checks if the submitted username exists in the users table in the database.

4. The server checks if the submitted password matches the password stored in the user record.
5. The server sends a response back to the client informing it that the login credentials are valid.

Alternatives

3a. The submitted username does not exist in the user table.

1. The server returns a response to the client informing it that the login credentials are invalid.

4a. The submitted password does not match the stored password in the user record.

1. The server returns a response to the client informing it that the login credentials are invalid.

4.3 Generate Scratch Card

Actors: Player

Description: This use case begins after a player has bought a scratch card.

Main Success Scenario

1. The client sends a message to the server to generate a new scratch card.
2. The server runs the probability algorithm to decide if the card is a winner.
3. The server creates a record for the new card in the database.
4. The new card record is linked to the user and saved to the database.
5. The server returns a message that the card has been created and is waiting to be redeemed.

4.4 Use Scratch Card

Actors: Player

Description: This use case begins when a player wishes to redeem a scratch card they have previously bought.

Main Success Scenario

1. The player selects a card from the “my cards” list.
2. The system asks for confirmation of the action.
3. The card details are retrieved from the database.
4. A graphic for a card is displayed on screen with a grey area on the left side that has the prize amount underneath.
5. The user clicks on the grey area.
6. The prize amount is displayed to the user.
7. The client sends the winning prize amount and card details to the server.
8. The server checks to see if the card has already been redeemed.
9. The prize amount is added to the clients balance in the database.
10. The card is marked as redeemed and its record updated in the database.
11. The server returns a message to the client that the prize has been redeemed.
12. The client displays the message to the player.

Alternatives

8a. The card has already been redeemed

1. The server returns an error message to the client informing it that the card has already been redeemed.
2. The client displays the message to the player

4.5 Create User

Actors: Player

Description: This use case begins when a player wishes to create a new user account in the application.

Main Success Scenario

1. The player arrives at the application home page.
2. The player selects the register link.
3. The client displays the registration form.
4. The player fills out the registration form.

5. The player submits the registration form.
6. The client sends the registration form to the server.
7. The server creates a new record for the user with the submitted details
8. The new record is submitted to the database.
9. The server returns a confirmation message to the client.

Alternatives

5a. The user enters data in the wrong format in one of the registration fields.

1. The user is kept on the registration form.
2. An error message is displayed to the user under the incorrect field.
3. The user corrects the error and resubmits.

4.6 Buy Scratch Card

Actors: Player

Description: This use case begins when the user wishes to buy a scratch card.

Main Success Scenario

1. The user selects the option to buy a scratch card from the homepage.
2. The user selects the type of card they wish to buy.
3. The user is asked for confirmation of the purchase.
4. The type of card and cost amount are sent to the server.
5. The server subtracts the cost amount from the user's balance.
6. The user's record is saved.

Alternatives

3a. The user does not have enough money in their account to buy a card.

1. An error message is displayed to the user explaining that they do not have enough funds to complete the transaction.
2. A link to the top up funds screen is displayed.

4.7 Redeem Balance

Actors: Player, PayPal

Description: This use case starts when a player wishes to redeem their current account balance to their PayPal account.

Main Success Scenario

1. The player selects the option to redeem their current funds.
2. The client displays the redeem form.
3. The user inputs how much they wish to redeem and their PayPal email address.
4. The user submits the redeem form.
5. The client submits the form to the server
6. The server uses the PayPal API to process the payment to the user's account.
7. The server returns a confirmation of success.
8. The client displays a success message to the user.

Alternatives

4a. The user tries to redeem more than they have available to redeem.

1. An error message is displayed informing the user they must enter an amount lower than or equal to their balance.
2. The user is returned to the redeem form.

4.8 Modify Prizes and Chances

Actors: Admin

Description: This use case begins when a logged in administrator wishes to modify the prizes on offer and/or the chances on each card to win one of those prizes.

Main Success Scenario

1. The admin selects the option to modify a game.
2. The client sends a message to the server to retrieve the selected game properties.
3. The server returns the current properties to the client.

4. The client displays the properties in a form.
5. The admin modifies the properties as desired. Prize chances should be floating point number between zero and one. They must add up to one. Prizes must be natural numbers greater than zero.
6. The admin submits the form.
7. The client returns the form to the server.
8. The server stores the properties in the database.

Alternatives

6a. The admin submits prize chances that do not add up to 1.

1. The admin is kept on the form.
2. An error message is displayed telling the admin to change the prize chances and displaying the total they currently add up to.
3. The admin corrects and resubmits.

6b. One of the prizes is set at an amount no allowed

1. The client displays a message warning the admin that a prize is being set at prohibited amount
2. The admin will be asked for confirmation of the amount.
3. The admin can confirm the amount and continue the form submission or cancel and go back to change the prize amount.

4.9 Generate Record

Actors: Web Client

Description: This use case begins when the web client sends a request to the server to retrieve and send back information from the database.

Main Success Scenario

1. The client sends a message to the server requesting information.
2. The server creates a request to the database to retrieve the appropriate data.
3. The database returns the data to the server.
4. The server makes a request to an external web server to retrieve the encryption key.

5. The encryption key is returned to the server.
6. The data retrieved is decrypted.
7. The data is returned to the client.

Alternatives

2b. The data requested does not exist.

1. The server returns a message to the client that the requested data does not exist.
2. The client displays the message.

4.10 Store Data

Actors: Web Client

Description: This use case begins when the client sends data to the server for storage in the database.

Main Success Scenario

1. The client sends data to the server.
2. The server requests the encryption key from an external web server.
3. The encryption key is returned to the server.
4. The data is encrypted.
5. The data is stored in the database.
6. A confirmation message is sent to the client.
7. The client displays the confirmation message.

4.11 Top Up Funds

Actors: Player, PayPal

Description: This use case begins when a user wishes to add funds to their account balance in the application.

Main Success Scenario

1. The player selects the top up funds option.

2. The client displays the top up funds form.
3. The user selects what card they are using for the payment.
4. The top up amount and security code for the card are entered.
5. The user submits the form.
6. The information is sent to the PayPal system for processing.
7. Payment confirmation is sent by PayPal to the server.
8. The server adds the appropriate amount to the user's account balance.
9. A payment confirmation message is sent to the client.
10. The client displays the confirmation message.

Alternatives

6a. There are not sufficient funds on the card to make the appropriate payment.

1. PayPal sends an error message to the server informing it the transaction could not be completed.
2. The server sends the error message to the client.
3. The client displays the error message.

6b. The user enters an incorrect security code for the card.

1. PayPal sends a message to the server informing it that the card was declined.
2. The server returns a message to the client informing it that the card code is wrong.
3. The client displays the message to the user.
4. The user re-enters the code and resubmits the payment.

4.12 Add Payment Card

Actors: Player

Description: This use case begins when a player wishes to add a new payment card to their account.

Main Success Scenario

1. The player selects the option to add a card to their account.
2. The client displays the add card form.
3. The player selects the card type.

4. The player inputs the card number and expiry date.
5. The player submits the form.
6. The client carries out validation checks on the data.
7. The client sends the data to the server.
8. The server stores the data in the database.
9. A confirmation message is sent back to the client.
10. The client displays the confirmation message.

Alternatives

6a. The card number is invalid.

1. The client stops the form submission.
2. The client displays a message to the user that card number is invalid.
3. The user corrects the area and resubmits the form.

6b. The expiry date entered has already passed.

1. The client stops the form submission.
2. The client displays a message to the user that the date entered is invalid.
3. The user corrects the error and resubmits the form.

4.13 View Cards

Actors: Player

Description: This use case begins when a user wishes to view all the unredeemed cards they have in their account.

Main Success Scenario

1. The user selects the view cards option.
2. The client sends a message to the server to retrieve the list of cards for that user.
3. The server returns a list of the user's cards that have not been redeemed.
4. The client displays the list of cards with relevant information and a link to redeem the card.

4.14 Add New Admin

Actors: Admin

Description: This use case begins when an admin wishes to add a new admin to the system.

Main Success Scenario

1. The admin selects the option to add a new admin
2. The client displays the add new admin form
3. The admin enters a new username and password for the new admin
4. The admin submits the form.
5. The client sends the form data to the server.

Alternatives

2a. The admin enters an admin username already in use

    1. An error message is displayed on the form that the entered name is already chosen and to select another.
    2. The admin selects an unchosen name and submits the form.

4.15 Add New Game

Actors: Admin

Description: This use case begins when an admin wishes to add a new scratch card game to the system.

Main Success Scenario

1. The admin selects the option to add a new game.
2. The client displays the new game form
3. The admin fills out the necessary details for the game.
4. The admin submits the form
5. Form data is sent to the server

Alternatives

3a. The new game has the same name as an existing game.

1. The client displays a message that the name is already in use.
2. The admin selects a new unchosen name and submits the form.

3b. The admin selects a new prize or chance that is outside the acceptable range.

1. The client displays an error message informing the admin that the value is outside the range.
2. The admin selects a new value and submits the form.

4.16 Top Up Funds(PayPal)

Actors: Player, PayPal

Description: This use case begins when a user wishes to add funds to their account balance in the application using the PayPal store

Main Success Scenario

1. The user selects the PayPal option on the top up form.
2. The user enters the amount they wish to top up by.
3. The user submits the form.
4. The data is sent to the server.
5. The server makes a call to PayPal API sending the required data.
6. PayPal returns a URL.
7. The client is redirected to the URL which opens the PayPal website
8. The user confirms the payment
9. The client is redirected back to the application
10. A page is displayed with the amount the user topped up by and their transaction ID

Alternatives

8a. The user cancels the payment on the PayPal website

1. The user is redirected back to the application homepage.

5.0 Navigation Sequence

The navigation sequence for ClickNWin will be as straightforward as possible to prevent user confusion. When users first arrive at the site, they will be greeted by the site's main home page for non-logged in users.

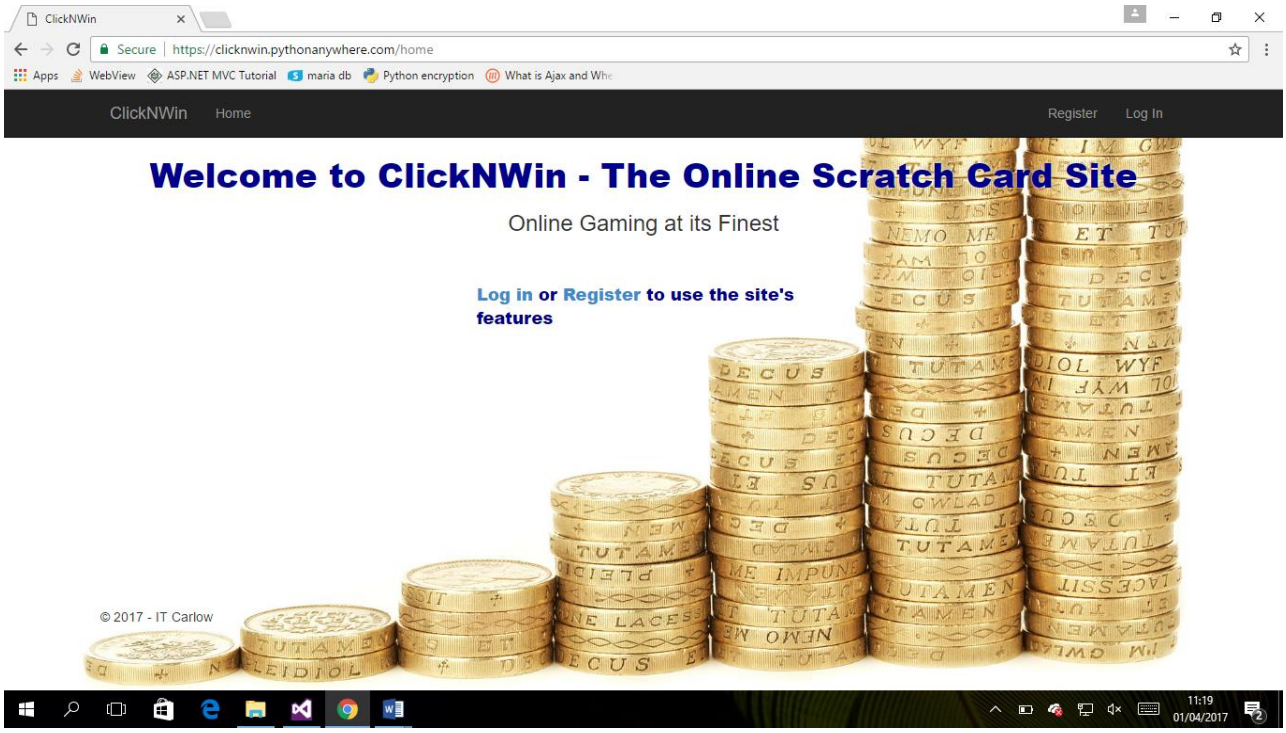

Fig 3. ClickNWin Homepage

After arriving at this page users will have the option to either login or register as a new user. After choosing an option, they will then be brought to the main logged in homepage where they will have access to the various functionalities of the application as well as a display in the navigation bar showing their current balance with the site.

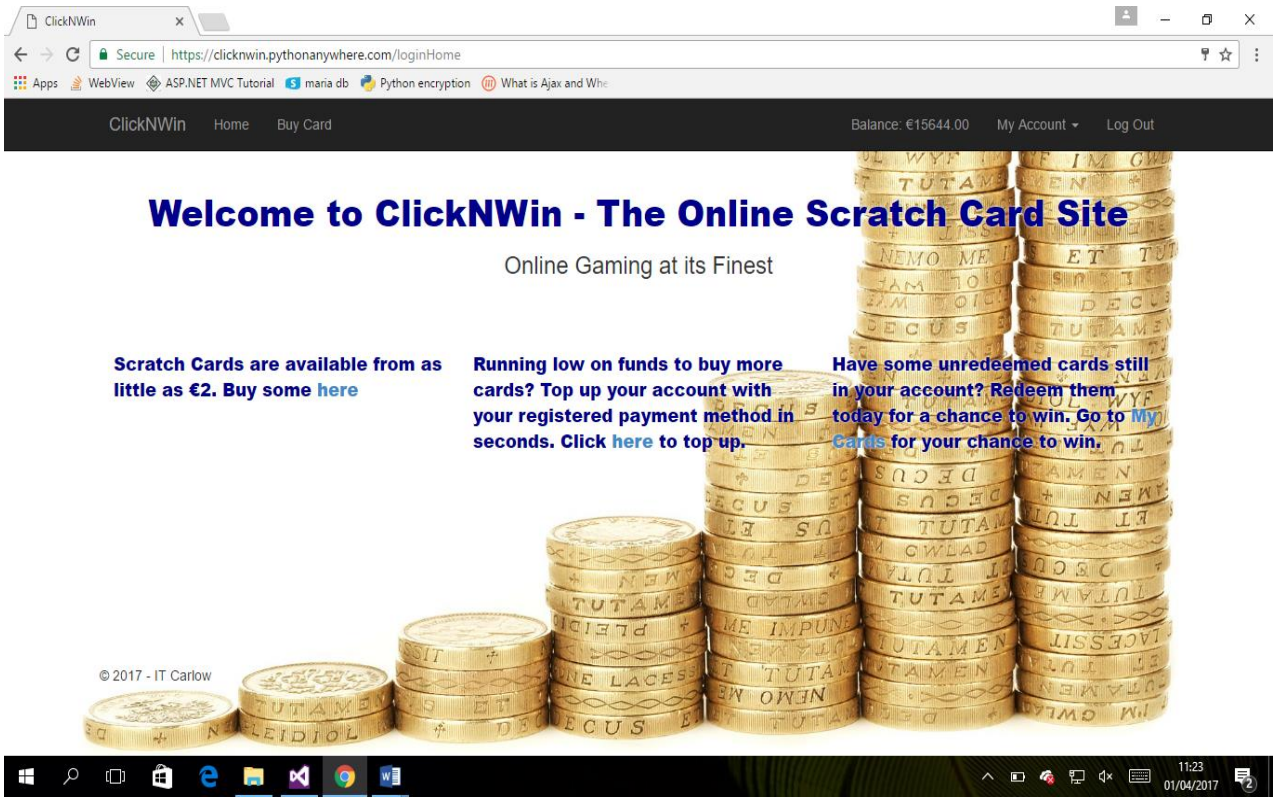

Fig 4. ClickNWin homepage for logged in users

The main paragraphs on the screen provide access to the more common areas of ClickNWin while the top navigation bar provides access to all functionalities. Users can choose to top up their balance, buy some scratch cards or redeem previously purchased cards. The following images show the design for the scratch cards before and after redemption.

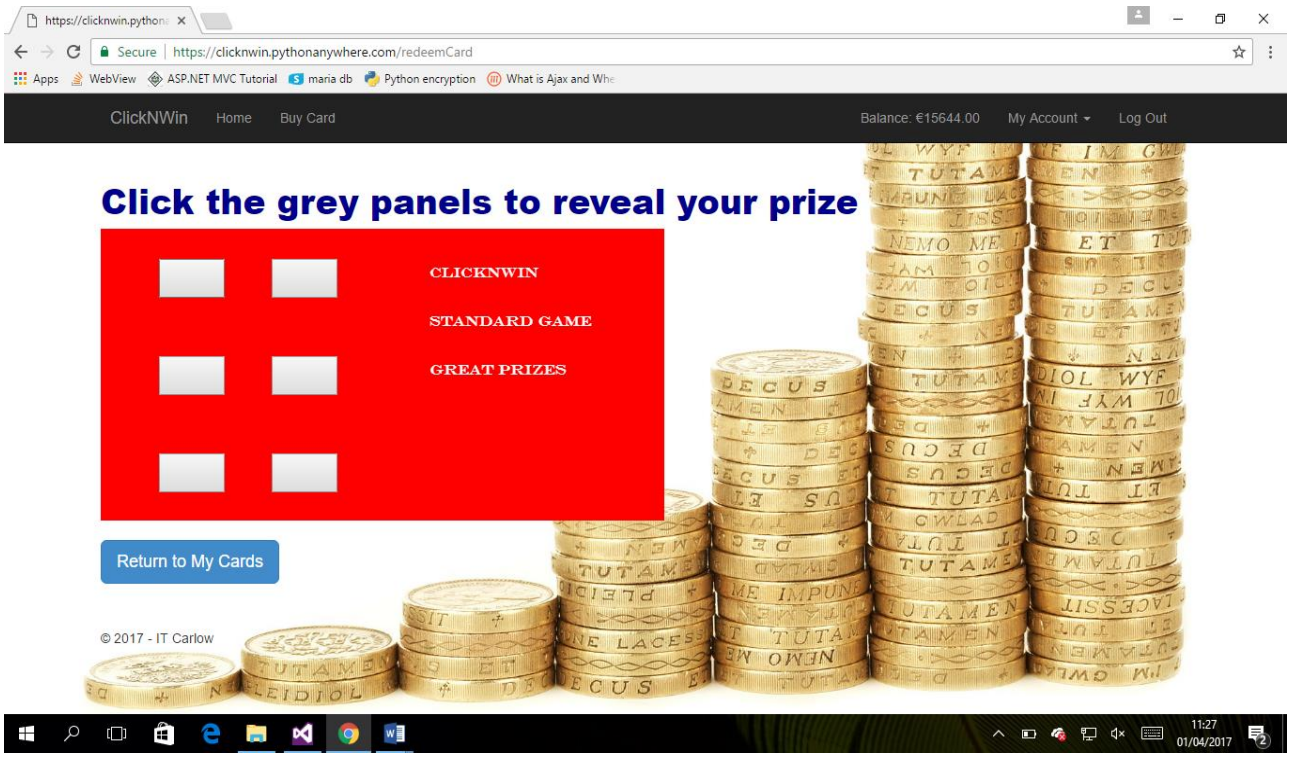

Fig 5. Scratch card about to be redeemed

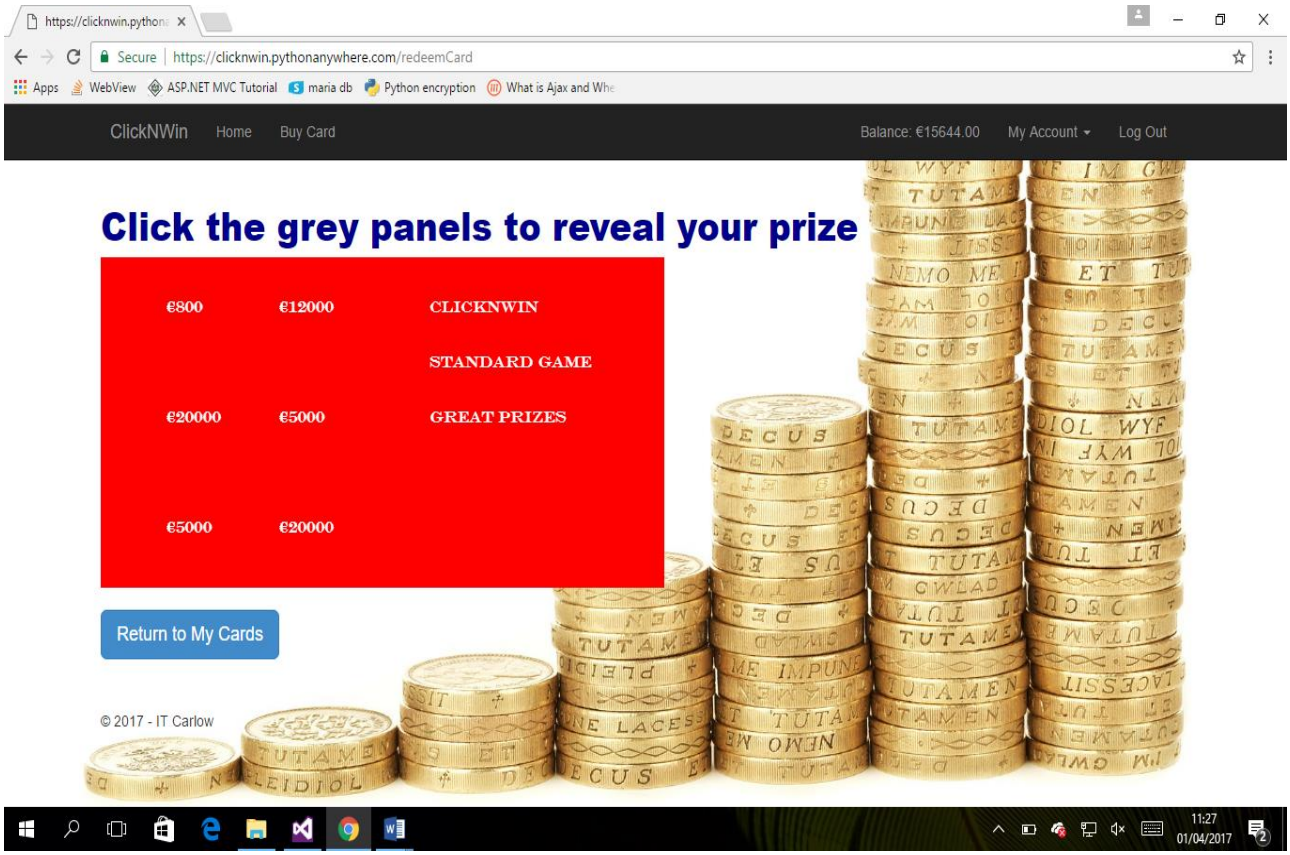

Fig 6. Redeemed Scratch Card

While logged in, users will not have access to the non-logged in homepage, registration form or login page. Any attempt to access them while logged in will result in the client redirecting the user to the logged in homepage. The same is true for non-logged in users trying to get to one of the internal pages like buying cards. They will be redirected to the non-logged in homepage.

There are no internal site links to the admin functionality pages. Admins need to know these links before they can access them. Users who are not logged in as admins will be redirected to the admin login page before they can access any admin functionality. They will need to have an admin username and password before they can log into the admin pages. Once logged in, admins will then be able to modify existing games as seen in the below screenshot.

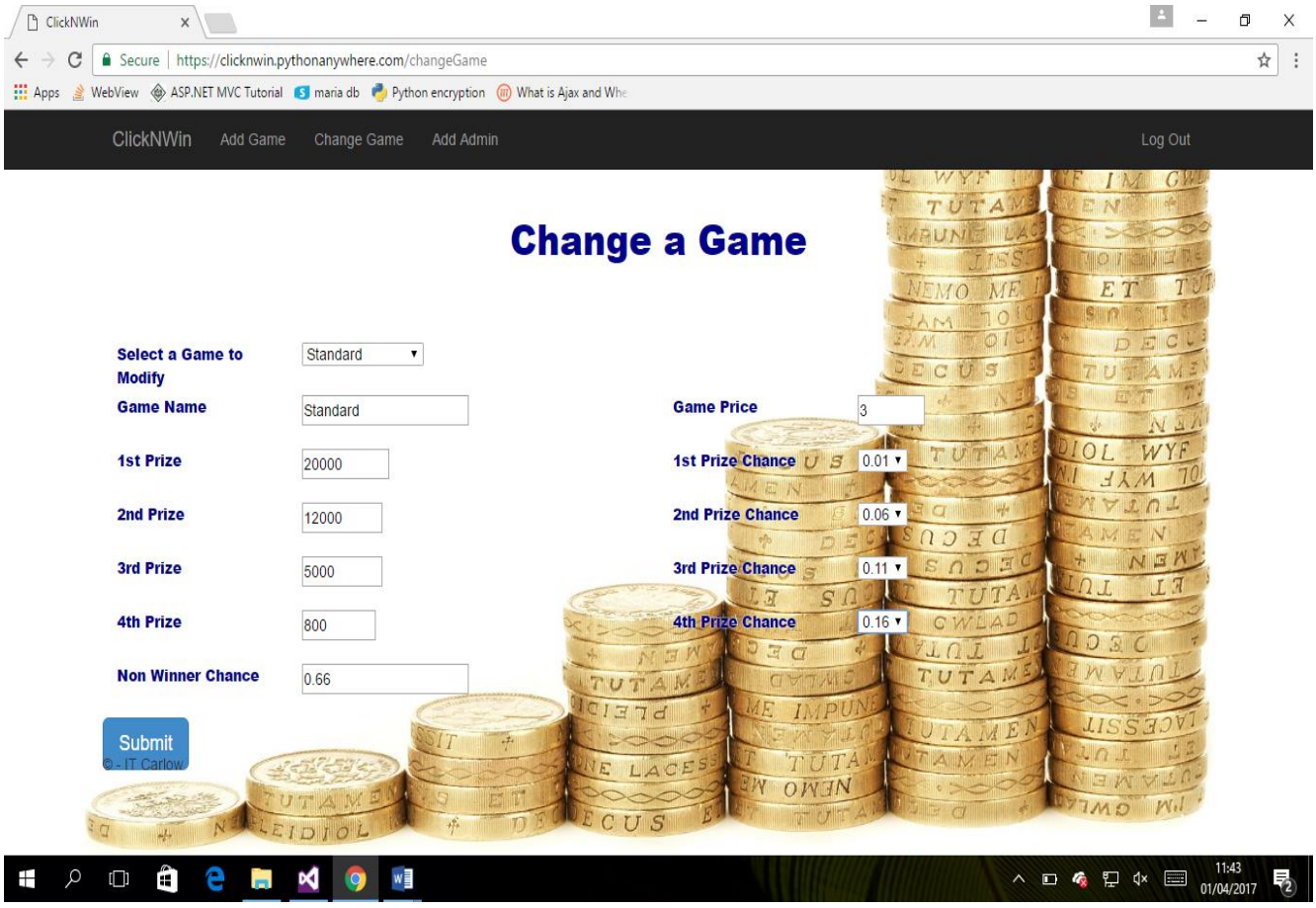

Fig 7. Modify Game Page

6.0 Security Requirements

6.1 AES Encryption

Due to the sensitive nature of some of the data ClickNWin will be handling, some security measures will be put in place to protect this data from malicious attackers. Firstly, all data persistently stored by ClickNWin will be encrypted. This will be achieved by using Advanced Encryption Standard(AES) algorithm with a 128 bit key. AES will be implemented in ClickNWin with the use of the pycrypto library. The pycrypto library contains several encryption algorithms including AES and is free and open source for anyone to use [1]. In each database function that inserts or updates data in the ClickNWin database, the pycrypto library will be used to encrypt the data before it goes into the database. In any functions where data is retrieved, once the data has been fetched from the database, pycrypto will be used to decrypt it. This will ensure the confidentiality of all data stored by ClickNWin.

6.2 HTTPS Implementation

Transmissions made between the client browser and the ClickNWin server could also be vulnerable to attack as the standard protocol for transmitting data between them transmits all data in plain text. This would happen before the data was encrypted on the ClickNWin server or when decrypted data is being sent from the server to the client. To secure these transmissions, ClickNWin will use the Hyper Text Transmission Protocol Secure(HTTPS) to encrypt all data being sent between the client and server. The Flask_SSLify library will be used to achieve this. This library can be implemented in a Flask application to enable it to use HTTPS provided the application is not in debug mode [2]. All testing of the HTTPS protocol will be done when the application is deployed as it will not work when running on a local machine.

References

[1] – Python Package Index. 2012. Pycrypto 2.6. [ONLINE]. Available at: https://pypi.python.org/pypi/pycrypto/2.6. [Accessed 30 March 2017].

[2] - Python Package Index. 2015. Flask-SSLify 0.1.5. [ONLINE]. Available at: https://pypi.python.org/pypi/Flask-SSLify/0.1.5. [Accessed 30 March 2017].