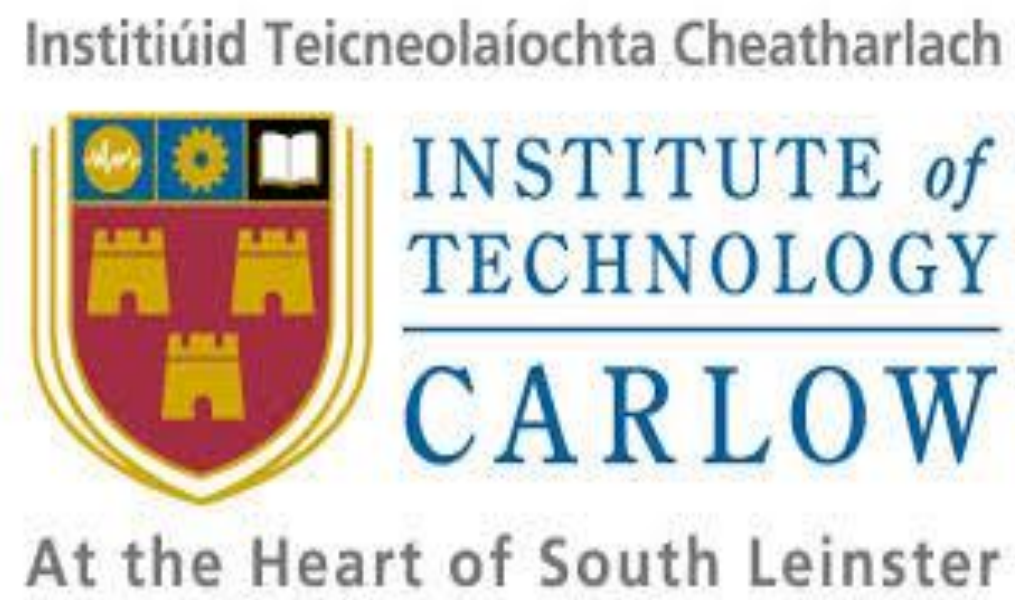

Final Year Project

Technical Manual

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 technical manual for the ClickNWin project. It will contain the installation instructions for the application and its dependencies, links to training videos for using the application and a copy of all the user written code from the project.

2.0 Installation Instructions

ClickNWin should be set up in a Python 3.5 environment. The project directory structure should not be changed while setting it up. To run ClickNWin, locally or on a cloud provider, the file runserver.py should be ran. Before the application can be ran, the following dependency libraries should be installed into the python environment that ClickNWin is being ran in using pip or your method of choice:

1. flask
2. paypalrestsdk
3. pycrypto
4. DBcm
5. flask_sslify

ClickNWin is also running in the cloud on the Python cloud provider PythonAnywhere. Its web address is clicknwin.pythonanywhere.com.

3.0 Training

The following training videos have been created for the purpose of showing user how to use the ClickNWin website and for training administrators for ClickNWin in how to perform their basic tasks on the admin section of the site.

1. User Training Video: https://www.youtube.com/watch?v=YpyrZ872x1w
2. Admin Training Video: https://www.youtube.com/watch?v=MhgjPhm6eMI

4.0 Project Code

The following is the project code for the ClickNWin application. In the case of files where code was auto generated, a note will be made in the document what was changed or updated by the project team. The GitHub repository for this project can be found at https://github.com/geoffo5/ClickNWin.

4.1 Python Code

4.1.1 Views.py

```
"""Routes and views for the flask application"""

from datetime import datetime
from functools import wraps
from flask import render_template, session, request, redirect, flash
from ClickNWin import app, database, utils


def isLoggedIn(func):#decorator to ensure only logged in users can access certain
pages
    @wraps(func)
    def wrapped_function(*args, **kwargs):
        if 'isLoggedIn' in session:
            return func(*args, **kwargs)
        return redirect('/home')
    return wrapped_function

def keepToLogin(func):#decorator to keep logged in users away from certain pages
    @wraps(func)
    def wrapped_function(*args, **kwargs):
        if 'isLoggedIn' in session:
            return redirect('/loginHome')
        return func(*args, **kwargs)
    return wrapped_function


@app.route('/')
@app.route('/home', methods=['GET'])
@keepToLogin
def home():
    return render_template('index.html', title='ClickNWin', year=datetime.now().year)

@app.route('/register')
@keepToLogin
def register():
    return render_template('register.html', title='ClickNWin', year =
datetime.now().year)

@app.route('/login')
@keepToLogin
def login():
```

```
    return render_template('login.html', title='ClickNWin', year =
datetime.now().year)

@app.route('/loginHome', methods=['POST', 'GET'])
@isLoggedIn
def loginHome():
    return render_template('loginHome.html', title='ClickNWin', year =
datetime.now().year, balance=database.getBalance(session['user']))

@app.route('/myCards', methods=['GET'])
@isLoggedIn
def myCards():
    userCards = database.getCards(session['user'])
    for i in userCards:
        i[1] = i[1][0:i[1].find('.')]
        i[1] = datetime.strptime(i[1], "%Y-%m-%d %H:%M:%S").strftime("%d-%m-%y %H:%M")

    return render_template('myCards.html',title='ClickNWin', year =
datetime.now().year, balance=database.getBalance(session['user']), cards = userCards)

@app.route('/registered', methods=['POST', 'GET'])
@keepToLogin
def registered():
    user = {'username':'', 'password':'', 'firstname':'', 'lastname':'', 'email':'',
'phone':'', 'dob':''}
    for k,v in request.form.items():
        user[k] = request.form[k]
    user['balance'] = '0.00'
    database.addUser(user)
    return render_template('registered.html',title='ClickNWin', year =
datetime.now().year)

@app.route('/loggedIn', methods=['POST'])
@keepToLogin
def loggedIn():
    username = request.form['username']
    password = request.form['password']
    success = database.login(username, password)

    if success:
        session['isLoggedIn'] = True
        session['user'] = request.form['username']
        return redirect('/loginHome')
    flash("Your username or password was incorrect.  Please try again","error")
    return redirect('/login')

@app.route('/addPaymentCard', methods=['POST', 'GET'])
@isLoggedIn
def addPaymentCard():
    return render_template('addPaymentCard.html', year = datetime.now().year,
balance=database.getBalance(session['user']))

@app.route('/logout', methods=['GET'])
def logout():
    session.pop('isLoggedIn')
    session.pop('user')
    return redirect('/home')

@app.route('/cardAdded', methods=['GET', 'POST'])
@isLoggedIn
def cardAdded():
```

```
    card = {'cardType':'', 'cardNumber':'', 'expiryMonths':0, 'expiryYears':0,
'cardFirstName':'', 'cardSurname':'', 'user': session['user']}
    for k,v in request.form.items():
        card[k] = request.form[k]

    database.addPaymentCard(card)
    return redirect('/loginHome')

@app.route('/buyCards', methods=['GET'])
@isLoggedIn
def buyCards():
    cards = database.getCardTypes()
    cardTypes = []
    for i in cards:
        cardTypes.append(i[0])
    return render_template('buyCard.html', year = datetime.now().year, cards =
cardTypes, balance=database.getBalance(session['user']))

@app.route('/cardsBought', methods=['POST', 'GET'])
@isLoggedIn
def cardsBought():
    sCards = {}

    if request.form['selectedUser'] == "":
        sCards['user'] = session['user']
    else:
        sCards['user'] = request.form['selectedUser']
    sCards['type'] = request.form['types']
    sCards['quantity'] = request.form['quantity']
    sCards['boughtBy'] = session['user']
    sCards['boughtOn'] = str(datetime.now())
    utils.newCards(sCards)
    price = request.form['price']
    database.reduceBalance(session['user'], price)
    return redirect("/myCards")

@app.route('/redeemCard', methods=['POST', 'GET'])
@isLoggedIn
def redeemCard():
    for i in request.form.items():
        id = i[0]
    card = database.getCard(id)
    if not card:
        return redirect('/myCards')
    return render_template('redeemCard.html', card=id, year = datetime.now().year,
balance=database.getBalance(session['user']))

@app.route('/topUp', methods=['GET'])
@isLoggedIn
def topUp():
    paymentCards = database.getPaymentCards(session['user'])
    formatCards = []
    index = 0
    for i in paymentCards:
        temp = ''
        formatCards.append({'id': i[0]})
        temp = i[1][12:]
        formatCards[index]['endNo'] = temp
        formatCards[index]['cardType'] = i[2]
        index = index + 1
    return render_template('topUp.html',payCards = formatCards, year =
datetime.now().year, balance=database.getBalance(session['user']))
```

```
@app.route('/addFunds', methods=['POST'])
@isLoggedIn
def addFunds():
    amount = request.form['amount']
    if request.form['payBy'] == "cardPay":
        cardID = request.form['card']
        cvv = request.form['cvv']
        data = utils.processCardPayment(session['user'], cardID, amount, cvv)
        if data:
            return render_template('fundsAdded.html',data = data, year =
datetime.now().year, balance=database.getBalance(session['user']))
    elif request.form['payBy'] == "paypal":
       data = utils.processPaypalPayment(session['user'], amount)
       if data:
            session['transactionID'] = data[1]
            session['amount'] = amount
            return redirect(data[0])
    flash("Payment Error.  Please check your details and try again", "error")
    return redirect('/topUp')


@app.route('/redeemBalance', methods=['GET'])
@isLoggedIn
def redeemBalance():
    return render_template('redeemBalance.html', year = datetime.now().year,
balance=database.getBalance(session['user']))

@app.route('/balanceRedeemed', methods=['POST'])
@isLoggedIn
def balanceRedeemed():
    amount = request.form['amount']
    email = request.form['email']
    password = request.form['password']
    success = database.login(session['user'], password)
    if not success:
        flash("Incorrect password.  Try again", "error")
        return redirect('/redeemBalance')
    amount = utils.formatCurrency(amount)
    if float(amount) > float(database.getBalance(session['user'])):
        flash("you do not have enough funds in your balance.", "error")
        return redirect('/redeemBalance')
    payoutSuccess = paypalAPI.balanceRedeem(email,amount)
    if payoutSuccess:
        database.reduceBalance(session['user'], amount)
        flash("Your payout was successful.  The requested funds will be available in
your account shortly.")
        return render_template('loginHome.html', year = datetime.now().year,
balance=database.getBalance(session['user']))
    else:
        flash("Payout Error.  Please check your details and try again", "error")
        return redirect('/redeemBalance')

@app.route('/paypalStoreReturn')
@isLoggedIn
def paypalStoreReturn():
    data = {}
    data['transactionID'] = session['transactionID']
    session['transactionID'] = ""
    data['user'] = session['user']
    data['amount'] = session['amount']
    session['amount'] = ""
```

```
    database.addFunds(data['user'], data['amount'])
    return render_template('paypalStoreReturn.html',data = data, year =
datetime.now().year, balance=database.getBalance(session['user']))

@app.route('/terms', methods=['GET'])
def terms():
    return render_template('terms.html', title='ClickNWin')
```

4.1.2 Admin.py

```
"""Routes and views for admin functions"""

from datetime import datetime
from functools import wraps
from flask import render_template, session, request, redirect, flash
from ClickNWin import app, database

def isAdmin(func):#decorator to ensure only logged in admins can access admin pages
    @wraps(func)
    def wrapped_function(*args, **kwargs):
        if 'isAdmin' in session:
            return func(*args, **kwargs)
        return redirect('/adminLogin')
    return wrapped_function


@app.route('/adminLogin')
def adminLogin():
    return render_template('adminLogin.html', title="ClickNWin")

@app.route('/adminLoggedIn', methods=['POST'])
def adminLoggedIn():
    admin = {}
    admin['user'] = request.form['username']
    admin['password'] = request.form['password']
    success = database.adminLogin(admin)
    if success:
        session['isAdmin'] = True
        session['admin'] = admin['user']
        return redirect('adminHome')
    flash("Username or password is incorrect.  Please try again")
    return redirect('/adminLogin')

@app.route('/adminLogout', methods=['GET'])
def adminLogout():
    session.pop('isAdmin')
    session.pop('admin')
    return redirect('/adminLogin')

@app.route('/adminHome')
@isAdmin
def adminHome():
    return render_template('adminHome.html', title='ClickNWin')

@app.route('/addAdmin', methods=['GET'])
@isAdmin
def addAdmin():
    return render_template('addAdmin.html', title='ClickNWin')
```

```
@app.route('/adminAdded', methods=['POST'])
@isAdmin
def adminAdded():
    admin = {}
    admin['username'] = request.form['username']
    admin['password'] = request.form['password']
    database.addAdmin(admin)
    return redirect('/adminHome')

@app.route('/addNewGame', methods=['GET'])
@isAdmin
def addNewGame():
    return render_template('addNewGame.html', title='ClickNWin')

@app.route('/newGameAdded', methods=['POST'])
@isAdmin
def newGameAdded():
    gameName = request.form['gameName']
    newGame = {'gameName':'', 'gamePrice':'', 'prize1':'', 'prize1Chance':'',
'prize2':'', 'prize2Chance':'', 'prize3':'', 'prize3Chance':'', 'prize4':'',
'prize4Chance':'', 'noWin':''}
    for k,v in request.form.items():
        newGame[k] = request.form[k]
    database.addCardType(newGame)
    flash("New Game " + gameName + " successfuly added")
    return redirect('/adminHome')

@app.route('/changeGame', methods=['GET'])
@isAdmin
def changeGame():
    games = database.getCardTypes()
    return render_template('changeGame.html', title='ClickNWin', games=games)

@app.route('/gameChanged', methods=['POST'])
@isAdmin
def gameChanged():
    gameName = request.form['gameName']
    changeGame = {'gameName':'', 'gamePrice':'', 'prize1':'', 'prize1Chance':'',
'prize2':'', 'prize2Chance':'', 'prize3':'', 'prize3Chance':'', 'prize4':'',
'prize4Chance':'', 'noWin':''}
    for k,v in request.form.items():
       changeGame[k] = request.form[k]
    database.modifyGame(changeGame)
    flash("Game " + gameName + " has been successfuly modified")
    return redirect('/adminHome')
```

4.1.3 API.py

```
from datetime import datetime
from functools import wraps
from flask import render_template, session, request, redirect, json, jsonify
from ClickNWin import app, database, utils, views, paypalAPI
"""AJAX API calls for the Flask Application"""


@app.route('/getCard', methods=['POST'])
def getCard():
    id = request.form['id']
    card = database.getCard(id)
    prizes = database.getCardPrizes(card[0])
    cardInfo = card + list(prizes[0])
```

```
    panels = utils.createPanelArray(cardInfo)
    return jsonify(card = panels)

@app.route('/checkUser', methods=['POST'])
def checkUser():
    user = request.form['user']
    exists = database.checkUsername(user)
    return jsonify(exists=exists)

@app.route('/getCardPrice', methods=['POST'])
def getCardPrice():
    type = request.form['type']
    price = database.getPrice(type)
    return jsonify(price=price)

@app.route('/cardRedeemed', methods=['POST'])
def cardRedeemed():
    id = request.form['id']
    card = database.getCard(id)
    prize = card[1]
    database.redeemCard(id)
    if prize:
        database.addFunds(session['user'], prize)
    return jsonify(prize = prize)

@app.route('/checkAdmin', methods=['POST'])
def checkAdmin():
    user = request.form['user']
    exists = database.checkAdmin(user)
    return jsonify(exists=exists)

@app.route('/getCardType', methods=['POST'])
def getCardType():
    id = request.form['id']
    cardType = database.getCard(id)
    return jsonify(cardType = cardType[0])

@app.route('/checkGame', methods=['POST'])
def checkGame():
    exists = False
    game = request.form['game']
    games = database.getCardTypes()
    for i in games:
        if game.lower() == i[0].lower():
            exists = True
    return jsonify(exists=exists)

@app.route('/getGame', methods=['POST'])
def getGame():
    game = request.form['game']
    data = database.getPrizes(game)
    cardData = data[0]
    return jsonify(data=cardData)
```

4.1.4 Database.py

```
import DBcm
import decimal
import base64
```

```
from ClickNWin import encrypt

"""Contains functions for inserting, updating and retrieving data from the MySQL
database"""

config = {
    'host': 'ClickNWin.mysql.pythonanywhere-services.com',
    'user': 'ClickNWin',
    'password': 'itcarlow',
    'database': 'ClickNWin$ClickNWin',
}

def checkUsername(username):
    username = encrypt.encrypt(username)

    _SQL = """SELECT username FROM users WHERE username = %s;"""
    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (username, ))
        user = database.fetchall()
    if len(user):
        return True
    return False


def addUser(user):
    duplicate = checkUsername(user['username'])
    if duplicate:
        return False

    for k in user:
        user[k] = encrypt.encrypt(user[k])

    _SQL = """INSERT INTO users
            (username, password, firstname, lastname, email, phone, birthdate,
balance)
            values
            (%s, %s, %s, %s, %s, %s, %s, %s)"""
    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (user['username'], user['password'], user['firstname'],
user['lastname'], user['email'],
                        user['phone'], user['dob'], user['balance']))

def login(username, password):
    username = encrypt.encrypt(username)

    _SQL = """SELECT username,password FROM users WHERE username = %s;"""
    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (username, ))
        user = database.fetchall()

    if not len(user):
        return False

    decUser = []
    decUser.append(user[0][0])
    decUser.append(encrypt.decrypt(user[0][1]))
    if decUser[1] != password:
        return False

    return True
```

```
def getBalance(username):#used to display the users balance at the top of each screen
    username = encrypt.encrypt(username)
    _SQL = """SELECT balance FROM users WHERE username = %s;"""
    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (username, ))
        balance = database.fetchall()
    decBalance = encrypt.decrypt(balance[0][0])
    numBal = '{:.2f}'.format(float(decBalance))
    return numBal

def addPaymentCard(card):
    for k in card:
        card[k] = encrypt.encrypt(card[k])

    _SQL = """INSERT INTO paymentcards
            (cardNumber, expiryMonth, expiryYear, cardType, holderFirstName,
holderSurname, user)
            values
            (%s, %s, %s, %s, %s, %s, %s)"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL,
(card['cardNumber'],card['expiryMonths'],card['expiryYears'],
card['cardType'],card['cardFirstName'],card['cardSurname'], card['user']))

def getCardTypes():
    decCards = []
    temp = []
    _SQL = """SELECT name, price FROM cardtypes"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL)
        cards = database.fetchall()
    for i in cards:
        temp = []
        for d in i:
            temp.append(encrypt.decrypt(d))
        decCards.append(temp)

    return decCards

def getPrizes(name):
    name = encrypt.encrypt(name)
    decPrizes = []
    temp = []
    _SQL = """SELECT * FROM cardtypes WHERE name = %s;"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (name, ))
        prizes = database.fetchall()
    for i in prizes:
        temp = []
        for d in i[1:]:
            temp.append(encrypt.decrypt(d))
        decPrizes.append(temp)

    return decPrizes

def getPrice(name):
    decPrice = []
    name = encrypt.encrypt(name)
    _SQL = """SELECT price FROM cardtypes WHERE name = %s;"""
```

```
    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (name, ))
        price = database.fetchall()
    decPrice.append(encrypt.decrypt(price[0][0]))
    return decPrice

def addScratchCard(card):
    for k in card:
        card[k] = encrypt.encrypt(card[k])

    _SQL = """INSERT INTO scratchcards
            (user, prize, type, boughtBy, boughtOn)
            values
            (%s, %s, %s, %s, %s)"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (card['user'], card['prize'], card['type'],
card['boughtBy'], card['boughtOn']))

def reduceBalance(user, amount):
    balance = float(getBalance(user))
    user = encrypt.encrypt(user)
    if amount[0] == '€':
        newBal = balance - float(amount[1:])
    else:
        newBal = balance - float(amount)
    newBal = encrypt.encrypt(str(newBal))

    _SQL = """UPDATE users SET balance = %s WHERE username = %s;"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL,(newBal, user))

def getCards(user):
    cards = []
    temp = []
    user = encrypt.encrypt(user)

    _SQL = """SELECT id, boughtOn, type, boughtBy FROM scratchcards WHERE user = %s
     AND redeemed = 0"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL,(user,))
        userCards = database.fetchall()

    for i in userCards:
        temp.append(i[0])
        for d in i[1:]:
            temp.append(encrypt.decrypt(d))
        cards.append(temp)
        temp = []
    return cards
def getCard(id):
    decCard = []
    _SQL = """SELECT type, prize FROM scratchcards WHERE id = %s AND redeemed = 0"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL,(id, ))
        card = database.fetchone()
    if not card:
        return card
```

```
    for i in card:
        decCard.append(encrypt.decrypt(i))
    return decCard

def redeemCard(id):
    _SQL = """UPDATE scratchcards SET redeemed = 1 WHERE id = %s"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (int(id),))

def addFunds(user,prize):
    balance = float(getBalance(user))
    user = encrypt.encrypt(user)
    newBal = balance + float(prize)
    newBal = str(newBal)
    newBal = encrypt.encrypt(newBal)

    _SQL = """UPDATE users SET balance = %s WHERE username = %s"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (newBal, user))

def getCardPrizes(name):
    temp = []
    decPrizes = []
    name = encrypt.encrypt(name)

    _SQL = """SELECT prize1, prize2, prize3, prize4 FROM cardtypes WHERE name = %s"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (name, ))
        prizes = database.fetchall()
    for i in prizes:
        temp = []
        for d in i:
            temp.append(encrypt.decrypt(d))
        decPrizes.append(temp)
    return decPrizes


def getPaymentCards(user):
    decCards = []
    temp = []
    user = encrypt.encrypt(user)
    _SQL = """SELECT id, cardNumber, cardType FROM paymentcards WHERE user = %s"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL,(user,))
        paymentCards = database.fetchall()

    for i in paymentCards:
        temp.append(i[0])
        for d in i[1:]:
            temp.append(encrypt.decrypt(d))
        decCards.append(temp)
        temp = []

    return decCards

def getPaymentCard(id):
    decCard = []
    temp = []
```

```
    _SQL = """SELECT id, cardNumber, expiryMonth, expiryYear, holderFirstName,
cardType, holderSurname from paymentcards WHERE id = %s""".format(id = id)

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL,(id,))
        paymentCard = database.fetchall()

    for i in paymentCard:
        temp.append(i[0])
        for d in i[1:]:
            temp.append(encrypt.decrypt(d))
        decCard.append(temp)
        temp = []
    return decCard

def adminLogin(admin):
    decUser = []
    admin['user'] = encrypt.encrypt(admin['user'])

    _SQL = """SELECT adminUsername, adminPassword FROM admin WHERE adminUsername =
%s"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (admin['user'], ))
        user = database.fetchone()
    if not len(user):
        return False

    for i in range(0,2):
        decUser.append(encrypt.decrypt(user[i]))

    if admin['password'] != decUser[1]:
        return False
    return True

def addAdmin(admin):
    for i in admin:
        admin[i] = encrypt.encrypt(admin[i])
    _SQL = """INSERT INTO admin
            (adminUsername, adminPassword)
            values
            (%s, %s)"""
    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (admin['username'], admin['password']))

def checkAdmin(user):
    user = encrypt.encrypt(user)
    _SQL = """SELECT adminUsername FROM admin WHERE adminUsername = %s"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (user, ))
        row = database.fetchone()
    if row:
        return True
    return False

def addCardType(newGame):
    for k in newGame:
        newGame[k] = encrypt.encrypt(newGame[k])

    _SQL = """INSERT INTO cardtypes
```

```
            (name, price, prize1, prize1chance, prize2, prize2chance, prize3,
prize3chance, prize4, prize4chance)
            values
            (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (newGame['gameName'], newGame['gamePrice'],
newGame['prize1'], newGame['prize1Chance'], newGame['prize2'],
newGame['prize2Chance'], newGame['prize3'], newGame['prize3Chance'],
newGame['prize4'], newGame['prize4Chance']))

def modifyGame(changeGame):
    for k in changeGame:
        changeGame[k] = encrypt.encrypt(changeGame[k])

    _SQL = """UPDATE cardtypes SET price=%s, prize1=%s, prize1chance=%s, prize2=%s,
prize2chance=%s, prize3=%s, prize3chance=%s, prize4=%s, prize4chance=%s WHERE name =
%s"""

    with DBcm.UseDatabase(config) as database:
        database.execute(_SQL, (changeGame['gamePrice'], changeGame['prize1'],
changeGame['prize1Chance'], changeGame['prize2'], changeGame['prize2Chance'],
changeGame['prize3'], changeGame['prize3Chance'], changeGame['prize4'],
changeGame['prize4Chance'], changeGame['gameName']))
```

4.1.5 Utils.py

```
from ClickNWin import database, paypalAPI
import random
import operator
"""Utility methods for creating scratch cards and creating panels for their design"""
def newCards(cards):#Creates new cards, runs a cumulatitive probability algorithm to
determine if they are winners and adds them to the database
    prizes = database.getPrizes(cards['type'])
    chances  = {}
    chances[prizes[0][2]] = float(prizes[0][3])
    chances[prizes[0][4]] = float(prizes[0][5])
    chances[prizes[0][6]] = float(prizes[0][7])
    chances[prizes[0][8]] = float(prizes[0][9])

    for i in range(0, int(cards['quantity'])):
        card = {}
        chance = random.uniform(0, 1)
        cumulative = 0.0
        prize = ""
        for k,v in sorted(chances.items(), key=operator.itemgetter(1)):
            cumulative += v
            if chance < cumulative:
                prize = k
                break
        card['prize'] = prize
        card['user'] = cards['user']
        card['boughtBy'] = cards['boughtBy']
        card['type'] = cards['type']
        card['boughtOn'] = cards['boughtOn']
        database.addScratchCard(card)
```

```
def createPanelArray(card):#creates a list of prizes to be displayed on the card based
on whether the card is a winner or not
    panels = []
    prizes = card[2:]
    while len(panels) < 6:
        if card[1] and card[1] not in panels:
            for i in range(0, 3):
                panels.append(card[1])
        pick = random.randint(2,5)
        if panels.count(card[pick]) < 2:
            panels.append(card[pick])
    return panels

def formatCurrency(amount):#formats the given number to look like a currency value
    point = False
    count = -1
    if amount[0] == '.':
        amount = '0' + amount
        for c in amount:
            if c == '.':
                point = True
            if point:
                count = count + 1
        if count == 1:
            amount = amount + '0'
    return amount

def processCardPayment(user, cardID, amount, cvv):#processes paypal payments using
stored credit cards
    card = database.getPaymentCard(int(cardID))
    card  = card[0]
    amount = formatCurrency(amount)
    paymentSuccess = paypalAPI.topUp(card, amount, cvv)
    if paymentSuccess:
        database.addFunds(user, amount)
        data = {}
        data['user'] = user
        data['amount'] = amount
        data['cardNo'] = card[1][12:]
        return data
    else:
        return False

def processPaypalPayment(user, amount):#processes paypal payments made through the
paypal store
    amount = formatCurrency(amount)
    data = paypalAPI.pay(amount)
    if data:
        return data
    else:
        return False
```

4.1.6 PayPalAPI.py

```
"""Functions for interacting with paypal to receive and make payments"""

import paypalrestsdk
from paypalrestsdk import Payment, Payout, ResourceNotFound
import logging
```

```
import random
import string

paypalrestsdk.configure({
  "mode": "sandbox",
  "client_id": "AVAIfIc67oJ5raw8pFxkp8OGyyOliG_4CQRXJPwU2HMY53gBHRcPwUi-
XqRMpmtvKeSfFDigy6PC2IVB",
  "client_secret":
"EPm1x9KMEVsVK6aymuHFrZgCTG7EGSWQmTTR9ab7RZNXjq5gCFEvfUxOa2Mk_YSAO2NdYzAJtjBGndTz" })

def topUp(card, amount, cvv):
    payment = paypalrestsdk.Payment({
      "intent": "sale",
      "payer": {
        "payment_method": "credit_card",
        "funding_instruments": [{
          "credit_card": {
            "type": card[5],
            "number": card[1],
            "expire_month": card[2],
            "expire_year": card[3],
            "cvv2": cvv,
            "first_name": card[4],
            "last_name": card[6] }}]},
      "transactions": [{
        "item_list": {
          "items": [{
            "name": "Funds Top Up",
            "sku": "Funds Top Up",
            "price": amount,
            "currency": "EUR",
            "quantity": 1 }]},
        "amount": {
          "total": amount,
          "currency": "EUR" },
        "description": "Fund Top Up for ClickNWin" }]})

    if payment.create():
      return True
    else:
      return False


def balanceRedeem(email, amount):
    sender_batch_id = ''.join(
    random.choice(string.ascii_uppercase) for i in range(12))

    payout = Payout({
        "sender_batch_header": {
            "sender_batch_id": sender_batch_id,
            "email_subject": "ClickNWin Balance Redeeemed"
        },
        "items": [
            {
                "recipient_type": "EMAIL",
                "amount": {
                    "value": amount,
                    "currency": "EUR"
                },
                "receiver": email,
                "note": "Thank you.",
                "sender_item_id": "item_1"
```

```
            }
        ]
    })

    if payout.create(sync_mode=True):
        return True
    else:
        return False

def pay(amount):
    payment = Payment({
        "intent": "sale",

        "payer": {
            "payment_method": "paypal"},

        # Url's that users will be redirected to after finishing their payment
        "redirect_urls": {
            "return_url": "https://clicknwin.pythonanywhere.com/paypalStoreReturn",
            "cancel_url": "https://clicknwin.pythonanywhere.com"},

        "transactions": [{

            "item_list": {
                "items": [{
                    "name": "item",
                    "sku": "item",
                    "price": amount,
                    "currency": "EUR",
                    "quantity": 1}]},


            "amount": {
                "total": amount,
                "currency": "EUR"},
            "description": "This is the payment transaction description."}]})

    if payment.create():
        print("Payment[%s] created successfully" % (payment.id))
        for link in payment.links:
            if link.method == "REDIRECT":
                redirect_url = str(link.href)
                data = [redirect_url, payment.id]
                return data
    else:
        return False
```

4.1.7 Encrypt.py

```
from Crypto.Cipher import AES
import base64
"""Contains function for encrypting and decrypting all database information using the
AES algorithm"""


key = "fVJ5YJasSDG3D4Ku"
iv =  "Cw35GddTyBnnuY37"

def encrypt(plain):
```

```
    enc = AES.new(key, AES.MODE_CFB, iv)
    cipher = enc.encrypt(plain)
    cipher = base64.b64encode(cipher)
    cipher = cipher.decode("utf-8")
    return cipher

def decrypt(cipher):
    if cipher is None:
        return ""
    enc2 = AES.new(key, AES.MODE_CFB, iv)
    cipher = base64.b64decode(cipher)
    plain = enc2.decrypt(cipher)
    plain = plain.decode("utf-8")
    return plain
```

4.1.8 __init__.py

The following file was autogenerated by Visual Studio but the yellow highlighted code was modified by the team.

```
"""
The flask application package.
"""

from flask import Flask
from flask_sslify import SSLify

app = Flask(__name__)
sslify = SSLify(app)
app.secret_key = 'ThisIsMySecretKeyForMyProject'


import ClickNWin.views
import ClickNWin.api
import ClickNWin.admin
```

4.2 HTML Code

4.2.1 Layout.html

This file is the layout page that is extended by pages displayed to users who are not logged in. This file was auto generated but modified by the project team to fit the application. Modified code in this file will be yellow highlighted. Code where the only change was a variable name will not be highlighted.

```
<!DOCTYPE html>
<html>
<head>
```

```
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{{ title }}</title>
    <link rel="stylesheet" type="text/css" href="/static/content/bootstrap.min.css" />
    <link rel="stylesheet" type="text/css" href="/static/content/site.css" />
    <script src="../static/scripts/jquery-1.10.2.min.js"></script>
    <script src="../static/scripts/FormScripts.js"></script>
    <script src="../static/scripts/AJAXCalls.js"></script>
    <script src="/static/scripts/bootstrap.js"></script>
    <script src="/static/scripts/respond.js"></script>
</head>

<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <a href="/" class="navbar-brand">ClickNWin</a>
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li><a href="{{ url_for('home') }}">Home</a></li>

                </ul>
                <ul class="nav navbar-nav navbar-right">
                    <li><a href="{{url_for('register')}}">Register</a></li>
                    <li><a href="{{url_for('login')}}">Log In</a></li>
                </ul>
            </div>
        </div>
    </div>



    <div class="container body-content">
        {% block content %}{% endblock %}
        <footer>
            <p>&copy; {{ year }} - IT Carlow</p>
        </footer>
    </div>


    {% block scripts %}{% endblock %}

</body>
</html>
```

4.2.2 Index.html

```
{% extends "layout.html" %}

{% block content %}
```

```
<h1 align="center">Welcome to ClickNWin - The Online Scratch Card Site</h1>
<h3 align="center">Online Gaming at its Finest</h3><br /><br />

<body>
    <div class="container">

        <div class="row row-centered">
            <div class="col-md-4"></div>
            <div class="col-md-4 col-centered mainp">
                <p class="loginphome"><a href="{{url_for('login')}}">Log in</a> or <a
href="{{url_for('register')}}">Register</a> to use the site's features</p>
            </div>
        </div>
    </div>
</body>

{% endblock %}
```

4.2.3 Register.html

```
{% extends "layout.html" %}

{% block content %}

<h1 align="center">Register an account with ClickNWin</h1>
<br><br><br />
<div class="container">
    <form class="form-inline" method="post" action="/registered" onsubmit="return
registerFormValidation()">

        <div class="row">
            <label class="col-md-2 col-form-label form-label">First Name</label>
            <div class="col-md-4">
                <input type="text" required id="firstname" name="firstname"
value="{{request.form['firstname']}}" maxlength="30" />
            </div>

                <label class="col-md-2 col-form-label form-label">Email</label>
            <div class="col-md-4">
                <input type="email" required id="email" name="email"
value="{{request.form['email']}}" maxlength="30" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Surname</label>

            <div class="col-md-4">
                <input type="text" required id="lastname" name="lastname"
value="{{request.form['lastname']}}" maxlength="30" />
            </div>

            <label class="col-md-2 col-form-label form-label">Phone Number</label>
            <div class="col-md-4">
                <input type="tel" id="phone" name="phone"
value="{{request.form['phone']}}" pattern="[0-9]{1,20}" title="Must be between 1-20
characters and contain only numbers" />
            </div>
        </div>
```

```
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Username</label>
            <div class="col-md-4">
                <input type="text" required id="username" name="username"
maxlength="20" value="{{request.form['username']}}" oninput="checkUser(this.value)" />
                <span id="userError"></span>
            </div>
            <label class="col-md-2 col-form-label form-label">Date of Birth</label>
            <div class="col-md-4">
                <input type="date" required id="dob" name="dob"
value="{{request.form['dob']}}" />
                <span id="ageError"></span>
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Password</label>
            <div class="col-md-4">
                <input type="password" required id="password" name="password"
maxlength="20" />
            </div>
            <div class="col-md-2">
                <span id="passMatch1"></span>
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Confirm Password</label>
            <div class="col-md-4">
                <input type="password" required id="cpassword"
name="cpassword"maxlength="20" />
            </div>
            <div class="col-md-2">
                <span id="passMatch2"></span>
            </div>
        </div>
        <br />
        <div class="row">
            <div class="col-md-6 mainp">
                <input type="checkbox" required id="t&c" name="t&c" />I have read and
agreed to the <a href="{{url_for('terms')}}" target="_blank"> Terms and Conditions</a>
of the site.
            </div>
            <div class="col-md-6 mainp">
                <input type="checkbox" required id="over18" name="over18" />I confirm
that I am 18 years of age or older.
            </div>
        </div>
        <br />
        <div class="row row-centered">
            <input class="btn btn-primary btn-lg" type="submit" value="Submit">
        </div>

    </form>
</div>

{% endblock %}
```

4.2.4 Login.html

```
{% extends "layout.html" %}


{% block content %}

<h1 align="center"> Login to your ClickNWin Account</h1>
<br><br><br />

    {% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
    {% for category, message in messages %}
    <span class="{{category}}">{{message}}</span>
    {% endfor %}
    {% endif %}
    {% endwith %}<br />

<div class="container">
    <form class="form-inline" action="/loggedIn" method="post">

        <div class="row">
            <label class="col-md-2 col-form-label">Username</label>
            <div class="col-md-4">
                <input type="text" required id="username" name="username"
value="{{request.form['username']}}" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label">Password</label>
            <div class="col-md-4">
                <input type="password" required id="password" name="password" />
            </div>
        </div>
        <br />
        <div class="row row-centered">
            <input class="btn btn-primary btn-lg" type="submit" value="Submit">
        </div>

    </form>
</div>

    {% endblock %}
```

4.2.5 Registered.html

```
{% extends "layout.html" %}

{% block content %}

<script>
    setTimeout("location.href = '/home';", 5000)
</script>

<h2 align="center">Your account has been registered.  You will now be redirected to
the home page to log in</h2>
```

```
{% endblock %}
```

4.2.6 Terms.html

```
{% extends "layout.html" %}

{% block content %}

<h1>
    Click N Win Terms and Conditions
</h1>

<ul>
    <li>Click N Win promises not to sell your data to bad corporations</li>
    <li>Click N Win will keep any of your personal data we store secure</li>
    <li>Click N Win takes no responsibility for your gambling addiction</li>
</ul>

{% endblock %}
```

4.2.7 LoginLayout.html

This file is extended by pages that are displayed to logged in users. The majority of the code was copied from the auto generated layout file. Changes to the code will be yellow highlighted.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{{ title }}</title>
    <link rel="stylesheet" type="text/css" href="/static/content/bootstrap.min.css" />
    <link rel="stylesheet" type="text/css" href="/static/content/site.css" />
    <link rel="stylesheet" type="text/css"
href="//cdn.datatables.net/1.10.12/css/jquery.dataTables.min.css" />
    <script src="../static/scripts/jquery-1.10.2.min.js"></script>
    <script src="/static/scripts/bootstrap.js"></script>
    <script src="/static/scripts/respond.js"></script>
    <script src="//cdn.datatables.net/1.10.12/js/jquery.dataTables.min.js"></script>
    <script src="../static/scripts/AJAXCalls.js"></script>
    <script src="../static/scripts/FormScripts.js"></script>
    <script src="/static/scripts/modernizr-2.6.2.js"></script>
</head>

<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
```

```
                    <span class="icon-bar"></span>
                </button>
                <a href="{{url_for('loginHome')}}" class="navbar-brand">ClickNWin</a>
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li><a href="{{ url_for('loginHome') }}">Home</a></li>
                    <li><a href="{{ url_for('buyCards') }}">Buy Card</a></li>
                </ul>
                <ul class="nav navbar-nav navbar-right">
                    <li><a>Balance: €<span id="balance">{{balance}}</span></a></li>
                    <li class="dropdown">
                        <a class="dropdown-toggle" data-toggle="dropdown" href="#">
                            My Account
                            <span class="caret"></span>
                        </a>
                        <ul class="dropdown-menu">
                            <li><a href="{{url_for('myCards')}}">My Cards</a></li>
                            <li><a href="{{url_for('redeemBalance')}}">Redeem
Balance</a></li>
                            <li><a href="{{url_for('topUp')}}">Top Up Funds</a></li>
                            <li><a href="{{url_for('addPaymentCard')}}">Update Payment
Methods</a></li>
                        </ul>
                    <li><a href="{{ url_for('logout') }}">Log Out</a></li>
                </ul>
            </div>
        </div>
    </div>
    <div class="overlay">



    <div class="container body-content">
        {% block content %}{% endblock %}
        <footer>
            <p>&copy; {{ year }} - IT Carlow</p>
        </footer>
    </div>
    </div>
    {% block scripts %}{% endblock %}
</body>
</html>
```

4.2.8 LoginHome.html

```
{% extends "loginLayout.html" %}

{% block content %}

<h1 align="center">Welcome to ClickNWin - The Online Scratch Card Site</h1>
<h3 align="center">Online Gaming at its Finest</h3>
{% with messages = get_flashed_messages() %}
{% if messages %}
{% for message in messages %}
<span>{{message}}</span>
{% endfor %}
{% endif %}
```

```
{% endwith %}
<br /><br /><br />

<div class="container">
    <div class="row row-centered">
        <div class="col-md-4 mainp">
            <p class="loginphome">Scratch Cards are available from as little as €2.
Buy some <a href="{{url_for('buyCards')}}">here</a></p>
        </div>
        <div class="col-md-4 mainp">
            <p class="loginphome">Running low on funds to buy more cards?  Top up your
account with your registered payment method in seconds.  Click <a
href="{{url_for('topUp')}}">here</a> to top up.</p>
        </div>
        <div class="col-md-4 mainp">
            <p class="loginphome">Have some unredeemed cards still in your account?
Redeem them today for a chance to win.  Go to <a href="{{url_for('myCards')}}">My
Cards</a> for your chance to win.</p>
        </div>
    </div>
</div>


{% endblock %}
```

4.2.9 AddPaymentCard.html

```
{% extends "loginLayout.html" %}

{% block content %}
<body onload="populateDates()">
    <h1 align="center">Add a New Payment Card to your Account</h1>
    <form class="form-inline" action="/cardAdded" method="post" onsubmit="return
validateCreditCardForm();">
        <div class="row">
            <img class="col-md-2" src="../../static/credit-card-logos.png" height="70"
width="100" />
        </div>
        <div class="row">
            <label class="col-md-1 col-form-label form-label">Card Type</label>
            <div class="col-md-2">
                <select id="cardType" name="cardType">
                    <option value="visa">Visa</option>
                    <option value="mastercard">MasterCard</option>
                </select>
            </div>
        </div><br />
        <div class="row">
            <label class="col-md-1 col-form-label form-label">Card Number</label>
            <div class="col-md-2">
                <input type="text" required id="cardNumber" name="cardNumber"
maxlength="16" pattern="[0-9]{16}" title="Enter a valid card number"/><br />
                <span id="invalidCard"></span>
            </div>
        </div><br />
        <div class="row">
            <label class="col-md-1 col-form-label form-label">Expiry Date</label>
            <div class="col-md-1">
                <select id="expiryMonths" name="expiryMonths" required></select>
```

```
            </div>
            <div class="col-md-1">
                <select id="expiryYears" name="expiryYears" required></select><span
id="invalidDate"></span>
            </div>
        </div><br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Card Holder First
Name</label>

            <div class="col-md-4">
                <input type="text" required id="cardName" name="cardFirstName" />
            </div>
        </div>
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Card Holder
Surname</label>

            <div class="col-md-4">
                <input type="text" required id="cardName" name="cardSurname" />
            </div>
        </div>
            <div class="row">
                <div class="row row-centered">
                    <input class="btn btn-primary btn-lg" type="submit"
value="Submit">
                </div>
            </div>
</form>
</body>

{% endblock %}
```

4.2.10 TopUp.html

```
{% extends "loginLayout.html" %}

{% block content %}

<h1>Top Up Funds</h1><br />
<h3>The selected card will be used to top up your account</h3><br />
{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}
<span class="{{category}}">{{message}}</span>
{% endfor %}
{% endif %}
{% endwith %}

<div class="container">
    <form action="/addFunds" method="post" onsubmit="return confirmTopUp();">
        <div class="row">
            <input type="radio"id="payBy1" name="payBy" value="cardPay" checked
onchange="topUpFormInput();">Pay with Stored Card<br>
            <input type="radio" id="payBy2" name="payBy" value="paypal"
onchange="topUpFormInput();">Pay via PayPal<br>
        </div>
        <div class="row" id="cardRow">
            <label class="col-md-2 col-form-label form-label">Select a Card</label>
```

```
            <div class="col-md-4">
                <select required id="paymentCards" name="card">
                    {% for i in payCards %}
                    <option value="{{i['id']}}">{{i['cardType']}} card ending in
{{i['endNo']}}</option>
                    {% endfor %}
                </select>
            </div><br /><br />
        </div>
        <div class="row" id="cvvRow">
            <label class="col-md-2 col-form-label form-label">Enter your CVV2
Number</label>
            <div class="col-md-4">
                <input type="text" id="cvv" name="cvv" required pattern="[0-9]{3}"
title="Must contain only three digits" />
            </div>
        </div><br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Enter the Top Up
Amount</label>
            <div class="col-md-4">
                <input type="number" required name="amount" id="amount" step="1"
min="1" max="1000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
        </div><br />
            <div class="row">
                <div class="row row-centered">
                    <input class="btn btn-primary btn-lg" type="submit" value="Submit"
/>
                </div>
            </div>
</form>
</div>

{% endblock %}
```

4.2.11 FundsAdded.html

```
{% extends "loginLayout.html" %}

{% block content %}

<h1>Payment Successful</h1>
<br /><br />

<p style="text-align:center">
    <h3>Details</h3>
    User: {{data['user']}}<br />
    Payment Card: Card ending in {{data['cardNo']}}<br />
    Amount: €{{data['amount']}}<br />
    Please save this page for your records<br />
    <a class="btn-primary btn btn-large" href="{{url_for('loginHome')}}">Return
Home</a>
</p>

{% endblock %}
```

4.2.12 PayPalStoreReturn.html

```
{% extends "loginLayout.html" %}

{% block content %}

<h1>Payment Successful</h1>
<br /><br />

<p style="text-align:center">
    <h3>Details</h3>
    User: {{data['user']}}<br />
    Transaction ID: {{data['transactionID']}}<br />
    Amount: €{{data['amount']}}<br />
    Please save this page for your records<br />
    <a class="btn-primary btn btn-large" href="{{url_for('loginHome')}}">Return
Home</a>
</p>

{% endblock %}
```

4.2.13 BuyCard.html

```
{% extends "loginLayout.html" %}

{% block content %}

<h1 align="center">Buy a Scratch Card</h1>
<br><br><br />
<div class="container">
    <form class="form-inline" method="post" action="/cardsBought" onsubmit="return
validateCardPurchase();">

        <div class="row">
            <label class="col-md-2 col-form-label form-label">Game</label>
            <div class="col-md-4">
                <select id="cardTypes" name="types" required onchange="calcPrice()">
                    {% set count = 0 %}
                    {% for i in cards %}
                        <option value="{{ i }}">{{ i }}</option>
                        {% set count = count + 1 %}
                    {% endfor %}
                </select><br /><br />
            </div>
        </div>
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Quantity</label>
            <div class="col-md-4">
                <select id="quantity" name="quantity"  onchange="calcPrice()">
                    {% for i in range(0,10) %}
                        <option value="{{i}}">{{i}}</option>
                    {% endfor %}
                </select><span id="quantityError"></span>
            </div>
        </div><br />
```

```
            <div class="row">
                <label class="col-md-2 col-form-label form-label">Price</label>
                <div class="col-md-4">
                    <input type="text" readonly id="price" name="price" value="€0.00"
/><span></span><br />
                </div>
            </div>
            <div class="row">
                <input type="radio" id="myself" name="user" value="myself" checked
onchange="addInput();">Buy for myself<br>
                <input type="radio" id="gift" name="user" value="gift"
onchange="addInput();">Buy as a gift<br>
            </div>
            <div class="row" id="userSelect" style="display:none;">
                <label class='col-md-2 col-form-label form-label'>User</label>
                <div class='col-md-4'>
                    <input type='text' id='selectedUser' name='selectedUser'
oninput='checkUser(this.value)' />
                    <span id='userError'></span>
                </div>
            </div>
            <div class="row row-centered">
                <input class="btn btn-primary btn-lg" type="submit" value="Submit">
            </div>
</form>
</div>

{% endblock %}
```

4.2.14 MyCards.html

```
{% extends "loginLayout.html" %}

{% block content %}

    <h1 align="center">My Cards</h1><br /><br />
    <form method="post" action="/redeemCard">
    <table id="cardsTable" class="display">
        <thead>
            <tr>
                <td>Card Number</td>
                <td>Bought On</td>
                <td>Type</td>
                <td>Bought By</td>
                <td>Redeem Link</td>
            </tr>
        </thead>
        <tbody>
            {% for i in cards %}
            <tr>
                <td>{{i[0]}}</td>
                <td>{{i[1]}}</td>
                <td>{{i[2]}}</td>
                <td>{{i[3]}}</td>
                <td><input type="submit" class="btn btn-primary btn-lg"
name="{{i[0]}}" value= "Redeem" /></td>
            </tr>

            {% endfor %}
```

```
        </tbody>
    </table>
</form>
<script>//creates data table for user scratch card display
    $(document).ready(function () {
        $('#cardsTable').DataTable({
            "pageLength": 5,
            "searching": false,
            "bLengthChange": false,
            "oLanguage": {
            "sEmptyTable":     "You currently have no unredeemed scratch cards"}
        });
    });
</script>


{% endblock %}
```

4.2.15 RedeemCard.html

```
{% extends "loginLayout.html" %}

{% block content %}

<h1>Click the grey panels to reveal your prize</h1>

<button class="cardPanel" id="panel1" style="top:170px;left:165px;"
onclick="reveal(this.id,{{card}})"></button>
<button class="cardPanel" id="panel2" style="top:270px;left:165px;"
onclick="reveal(this.id,{{card}})"></button>
<button class="cardPanel" id="panel3" style="top:370px;left:165px;"
onclick="reveal(this.id,{{card}})"></button>
<button class="cardPanel" id="panel4" style="top:170px;left:285px;"
onclick="reveal(this.id,{{card}})"></button>
<button class="cardPanel" id="panel5" style="top:270px;left:285px;"
onclick="reveal(this.id,{{card}})"></button>
<button class="cardPanel" id="panel6" style="top:370px;left:285px;"
onclick="reveal(this.id,{{card}})"></button>


<canvas id="card" width="600"
height="300"></canvas><script>drawCard({{card}});</script><br /><br />

<a href="{{url_for('myCards')}}" class="btn btn-lg btn-primary">Return to My Cards</a>

{% endblock %}
```

4.2.16 RedeemBalance

```
{% extends "loginLayout.html" %}

{% block content %}

<h1>Redeem Balance</h1><br />
<h3>Enter the email address of your paypal account<br /> and the selected amount will
be refunded to it.
```

```
    <br />Enter your password to confirm the redemption</h3><br />
{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}
<span class="{{category}}">{{message}}</span>
{% endfor %}
{% endif %}
{% endwith %}<br />

<div class="container">
    <form class="form-inline" action="/balanceRedeemed" method="post" onsubmit="return
checkBalance({{balance}});">
        <div class="row">
            <label class="col-md-1 col-form-label form-label">Email</label>
            <div class="col-md-2">
                <input type="email" required name="email" value="{{email}}" />
            </div>
        </div><br />
        <div class="row">
            <label class="col-md-1 col-form-label form-label">Amount</label>
            <div class="col-md-2">
                <input type="number" required name="amount" id="amount" step="0.01"
pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
        </div><br />
        <div class="row">
            <label class="col-md-1 col-form-label form-label">Password</label>
            <div class="col-md-2">
                <input type="password" required name="password" id="password"
maxlength="20" />
            </div>
        </div>
        <div class="row">
            <div class="row row-centered">
                <input class="btn btn-primary btn-lg" type="submit" value="Submit">
            </div>
        </div>
    </form>
</div>

    {% endblock %}
```

4.2.17 AdminLayout

This file was copied from the auto generated layout file. Code that was modified in this file will be yellow highlighted.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{{ title }}</title>
    <link rel="stylesheet" type="text/css" href="/static/content/bootstrap.min.css" />
    <link rel="stylesheet" type="text/css" href="/static/content/site.css" />
    <link rel="stylesheet" type="text/css"
href="//cdn.datatables.net/1.10.12/css/jquery.dataTables.min.css" />
```

```
    <script src="../static/scripts/jquery-1.10.2.min.js"></script>
    <script src="/static/scripts/bootstrap.js"></script>
    <script src="/static/scripts/respond.js"></script>
    <script src="//cdn.datatables.net/1.10.12/js/jquery.dataTables.min.js"></script>
    <script src="../static/scripts/FormScripts.js"></script>
    <script src="../static/scripts/AJAXCalls.js"></script>
    <script src="/static/scripts/modernizr-2.6.2.js"></script>
</head>

<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <a class="navbar-brand">ClickNWin</a>
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li><a href="{{ url_for('addNewGame') }}">Add Game</a></li>
                    <li><a href="{{ url_for('changeGame') }}">Change Game</a></li>
                    <li><a href="{{ url_for('addAdmin') }}">Add Admin</a></li>
                </ul>
                <ul class="nav navbar-nav navbar-right">
                    <li><a href="{{ url_for('adminLogout') }}">Log Out</a></li>
                </ul>
            </div>
        </div>
    </div>
    <div class="overlay">



        <div class="container body-content">
            {% block content %}{% endblock %}
            <footer>
                <p>&copy; {{ year }} - IT Carlow</p>
            </footer>
        </div>
    </div>
    {% block scripts %}{% endblock %}
</body>
</html>
```

4.2.18 AdminLogin.html

```
<html>
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>{{ title }}</title>
        <link rel="stylesheet" type="text/css"
href="/static/content/bootstrap.min.css" />
        <link rel="stylesheet" type="text/css" href="/static/content/site.css" />
```

```
        <link rel="stylesheet" type="text/css"
href="//cdn.datatables.net/1.10.12/css/jquery.dataTables.min.css" />
        <script src="../static/scripts/jquery-1.10.2.min.js"></script>
        <script src="/static/scripts/bootstrap.js"></script>
        <script src="/static/scripts/respond.js"></script>
        <script
src="//cdn.datatables.net/1.10.12/js/jquery.dataTables.min.js"></script>
        <script src="../static/scripts/AJAXCalls.js"></script>
        <script src="../static/scripts/FormScripts.js"></script>
        <script src="/static/scripts/modernizr-2.6.2.js"></script>
    </head>
<body>
    <h2>ClickNWin Admin Login</h2>
    {% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
    {% for category, message in messages %}
    <span class="{{category}}">{{message}}</span>
    {% endfor %}
    {% endif %}
    {% endwith %}<br />
    <div class="container">
        <form action="/adminLoggedIn" method="post">
            <div class="row">
                <label class="col-md-2 col-form-label form-label">Username</label>
                <div class="col-md-4">
                    <input type="text" id="username" name="username" required  /><br
/>
                </div>
            </div><br />
            <div class="row">
                <label class="col-md-2 col-form-label form-label">password</label>
                <div class="col-md-4">
                    <input type="password" id="password" name="password" required
/><br />
                </div>
            </div><br />
            <div class="row">
                <div class="row row-centered">
                    <input class="btn btn-primary btn-lg" type="submit"
value="Submit">
                </div>
            </div>
        </form>
    </div>

</body>

</html>
```

4.2.19 AdminHome.html

```
{% extends "adminLayout.html" %}

{% block content %}


<h1>
    Click N Win Admin Home
</h1>
```

```
{% with messages = get_flashed_messages() %}
{% if messages %}
{% for message in messages %}
<span>{{message}}</span>
{% endfor %}
{% endif %}
{% endwith %}
<br /><br /><br />

{% endblock %}
```

4.2.20 AddAdmin.html

```
{% extends "adminLayout.html" %}

{% block content %}
<h1>Add a New Admin</h1>
<div class="container">
    <form action="/adminAdded" method="post" onsubmit="return addAdminValidation();">
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Username</label>
            <div class="col-md-4">
                <input type="text" required id="username" name="username"
oninput="checkAdmin();" maxlength="20" /><span id="userError"></span>
            </div>
        </div><br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Password</label>
            <div class="col-md-4">
                <input type="password" required id="password" name="password"
maxlength="20" /><span id="passMatch1"></span><br />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Confirm Password</label>
            <div class="col-md-4">
                <input type="password" required id="cpassword" name="cpassword"
maxlength="20" /><span id="passMatch2"></span><br />
            </div>
                <div class="row">
                    <div class="row row-centered">
                        <input class="btn btn-primary btn-lg" type="submit"
value="Submit">
                    </div>
                </div>
    </form>
</div>

{% endblock %}
```

4.2.21 AddNewGame.html

```
{% extends "adminLayout.html" %}
```

```
{% block content %}

<h1 align="center">Add a New Game</h1>
<br><br><br />
<div class="container">
    <form class="form-inline" method="post" action="/newGameAdded" onsubmit="return
newGameValidation()">
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Game Name</label>
            <div class="col-md-4">
                <input type="text" required id="gameName" name="gameName"
oninput="checkGame()" /><span id="gameError"></span>
            </div>
            <label class="col-md-2 col-form-label form-label">Game Price</label>
            <div class="col-md-4">
                <input type="number" required id="gamePrice" name="gamePrice" min="1"
max="100" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">1st Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize1" name="prize1" step="1"
min="5000" max="100000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">1st Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize1Chance" name="prize1Chance"
oninput="calcNoWinChance();" step="0.01" min="0.01" max="0.06" pattern="[0]{1}+[.]+[0-
9]{2}" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">2nd Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize2" name="prize2" step="1"
min="2000" max="30000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">2nd Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize2Chance" name="prize2Chance"
oninput="calcNoWinChance();" step="0.01" min="0.04" max="0.12" pattern="[0]{1}+[.]+[0-
9]{2}" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">3rd Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize3" name="prize3" step="1"
min="100" max="10000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">3rd Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize3Chance" name="prize3Chance"
oninput="calcNoWinChance();" step="0.01" min="0.08" max="0.17" pattern="[0]{1}+[.]+[0-
9]{2}" />
            </div>
        </div>
        <br />
```

```
        <div class="row">
            <label class="col-md-2 col-form-label form-label">4th Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize4" name="prize4" step="1"
min="10" max="1000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">4th Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize4Chance" name="prize4Chance"
onchange="calcNoWinChance();" step="0.01" min="0.10" max="0.19"
pattern="[0]{1}+[.]+[0-9]{2}" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Non Winner
Chance</label>
            <div class="col-md-4">
                <input type="text" readonly required value="0.00" id="notAWin"
name="notAWin" step="0.01" />
            </div>
        </div>
        <br />
        <div class="row row-centered">
            <input class="btn btn-primary btn-lg" type="submit" value="Submit">
        </div>
    </form>
</div>
{% endblock %}
```

4.2.22 ChangeGame.html

```
{% extends "adminLayout.html" %}

{% block content %}

<h1 align="center">Change a Game</h1><br><br><br />
<div class="container">
    <form class="form-inline" method="post" action="/gameChanged" onsubmit="return
newGameValidation()">
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Select a Game to
Modify</label>
            <div class="col-md-4">
                <select id="games" required onchange="getGame(this.value);">
                    <option disabled selected>Select a game...</option>
                    {% for i in games %}
                    <option value="{{i[0]}}">{{i[0]}}</option>
                    {% endfor %}
                </select>
            </div>
        </div>
        <div class="row">
            <label class="col-md-2 col-form-label form-label">Game Name</label>
            <div class="col-md-4">
                <input type="text" required id="gameName" name="gameName" readonly
/><span id="gameError"></span>
            </div>
            <label class="col-md-2 col-form-label form-label">Game Price</label>
```

```
            <div class="col-md-4">
                <input type="number" required id="gamePrice" name="gamePrice" min="1"
max="100" />
            </div>
        </div> <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">1st Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize1" name="prize1" step="1"
min="5000" max="100000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">1st Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize1Chance" name="prize1Chance"
oninput="calcNoWinChance();" step="0.01" min="0.01" max="0.06" pattern="[0]{1}+[.]+[0-
9]{2}" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">2nd Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize2" name="prize2" step="1"
min="2000" max="30000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">2nd Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize2Chance" name="prize2Chance"
oninput="calcNoWinChance();" step="0.01" min="0.04" max="0.12" pattern="[0]{1}+[.]+[0-
9]{2}" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">3rd Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize3" name="prize3" step="1"
min="100" max="10000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">3rd Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize3Chance" name="prize3Chance"
oninput="calcNoWinChance();" step="0.01" min="0.08" max="0.17" pattern="[0]{1}+[.]+[0-
9]{2}" />
            </div>
        </div>
        <br />
        <div class="row">
            <label class="col-md-2 col-form-label form-label">4th Prize</label>
            <div class="col-md-4">
                <input type="number" required id="prize4" name="prize4" step="1"
min="10" max="1000" pattern="[0-9]{1,}+[.]+[0-9]{2}" />
            </div>
            <label class="col-md-2 col-form-label form-label">4th Prize Chance</label>
            <div class="col-md-4">
                <input type="number" required id="prize4Chance" name="prize4Chance"
onchange="calcNoWinChance();" step="0.01" min="0.10" max="0.19"
pattern="[0]{1}+[.]+[0-9]{2}" />
            </div>
        </div>
        <br />
        <div class="row">
```

```
            <label class="col-md-2 col-form-label form-label">Non Winner
Chance</label>
            <div class="col-md-4">
                <input type="text" readonly required value="0.00" id="notAWin"
name="notAWin" step="0.01" />
            </div>
        </div> <br />
        <div class="row row-centered">
            <input class="btn btn-primary btn-lg" type="submit" value="Submit">
        </div>
    </form>
</div>
{% endblock %}
```

4.3 JavaScript Code

4.3.1 FormScripts.js

This file contains form validation scripts and other utility scripts for use in the application.

```
function registerFormValidation()//preforms validation on the registration form
{
    var pass1 = document.getElementById("password").value;
    var pass2 = document.getElementById("cpassword").value;
    var sPass1 = String(pass1);
    var sPass2 = String(pass2);
    var username = document.getElementById("username").value;
    var dob = document.getElementById("dob").value;
    var today = new Date()
    var birth = new Date(dob);

    var diff = today - birth.getTime();
    ageMs = new Date(diff);
    age = Math.abs(ageMs.getUTCFullYear() - 1970);

    var user = document.getElementById("userError").innerText;

    if(pass1 != pass2)//outputs error messages if passwords do not match and prevents
form submit
    {
        document.getElementById("passMatch1").className = "error"
        document.getElementById("passMatch2").className = "error"
        document.getElementById("passMatch1").innerText = "Passwords do not match";
        document.getElementById("passMatch2").innerText = "Passwords do not match";
        return false;
    }
    else
    {
        document.getElementById("passMatch1").className = ""
        document.getElementById("passMatch2").className = ""
        document.getElementById("passMatch1").innerText = "";
        document.getElementById("passMatch2").innerText = "";
    }

    if (diff < 0)
    {
        document.getElementById("ageError").className = "error";
```

```
        document.getElementById("ageError").innerText = "Incorrect date entered";
        return false
    }
    else
    {
        document.getElementById("ageError").className = "";
        document.getElementById("ageError").innerText = "";
    }

    if(age < 18)//if age is too low, do not submit form and output error message
    {
        document.getElementById("ageError").className = "error";
        document.getElementById("ageError").innerText = "Your age must be 18 or
greater";
        return false;
    }
    else
    {
        document.getElementById("ageError").className = "";
        document.getElementById("ageError").innerText = "";
    }
    if (user != "")//if user error message still displayed, then do not submit form
    {
        return false;
    }


}

function populateDates()//used on addPayment cards to prepare selectable dates
{
    var months = document.getElementById("expiryMonths");
    var years = document.getElementById("expiryYears");

    for(var i = 1; i <= 12; i++)
    {
        var opt = document.createElement("option");
        opt.innerHTML = i;
        opt.value = i;
        months.appendChild(opt);
    }

    var year = new Date().getFullYear();

    for(var i = 0;i < 7;i++)
    {
        var opt = document.createElement("option");
        opt.innerHTML = year;
        opt.value = year;
        years.appendChild(opt);
        year++;
    }
}

function validateCardNo()//uses the Luhn algorithm to validate card number is correct
{
    number = document.getElementById("cardNumber").value;
    numArray = [];
    accepted = ['0','1', '2', '3', '4', '5', '6', '7', '8', '9'];

    for(var i = number.length-1;i >= 0;i--)
    {
```

```
        for (var innerI = 0; innerI < accepted.length; innerI++)
        {
            if (number[i] === accepted[innerI])
            {
                numArray[i] = number.charAt(i);
                numArray[i] = parseInt(numArray[i]);
            }
        }
    }

    for (var i = 0; i < numArray.length; i++)
    {
        if(numArray[i] == null)
        {
            numArray.splice(i, 1);
        }
    }
    for(var i = numArray.length - 2;i >= 0;i -= 2)
    {
        numArray[i] *= 2;

        if (numArray[i] > 9)
        {
            numArray[i] -= 9;
        }
    }
    var total = 0;
    for (var i = 0;i < numArray.length;i++)
    {
        total += numArray[i];
    }

    if (total % 10 != 0)
    {
        document.getElementById("invalidCard").style.backgroundColor = "#EB4141";
        document.getElementById("invalidCard").innerText = "Invalid card number.
Please re-enter"
        return true;
    }
    return false;
}


function validateDate()//ensures a card expiry date is valid
{
    var expiryYear = document.getElementById("expiryYears").value;
    var expiryMonth = document.getElementById("expiryMonths").value;
    var year = new Date().getFullYear();
    var month = new Date().getMonth();
    console.log(expiryMonth)
    console.log(month);

    if(expiryYear == year && expiryMonth <= month)
    {
        document.getElementById("invalidDate").style.backgroundColor = "#EB4141";
        document.getElementById("invalidDate").innerText = "Invalid Expiry Date.
Please re-enter";
        return true;
    }
    return false;
}
```

```
function validateCreditCardForm()//performs validations when adding a new credit to an
account
{

    var cardNumber = validateCardNo();
    var date = validateDate();

    if(date == true || cardNumber == true)
    {
        return false;
    }
}

function addInput()//adds a user field if scratch card is being bought for a friend
{
    var check = document.getElementById("myself").checked;
    if (check)
    {
        document.getElementById("selectedUser").value = ""
        document.getElementById("userSelect").style.display = 'none'
        document.getElementById("userSelect").required = false
    }
    else
    {
        document.getElementById("userSelect").style.display = 'block'
        document.getElementById("userSelect").required = true
    }
}


function validateCardPurchase()//validates scratch card purchases
{
    var quantity = document.getElementById("quantity").value;
    if (quantity == '0')
    {
        document.getElementById("quantityError").innerText = "Please select a quantity
greater than 0";
        document.getElementById("quantityError").className = "error";
        return false
    }

    var balance = document.getElementById("balance").innerText;
    var price = document.getElementById("price").value;
    price = price.substring(1, price.length);
    balance = parseFloat(balance);
    price = parseFloat(price);
    if (price > balance)
    {
        alert("You do not have enough funds to buy these cards.  Please top up and try
again");
        return false
    }

    var user = document.getElementById("userError").innerText;
    if (user != "")
    {
        return false;
    }
}

function checkBalance(balance)//ensures user is able to redeem the requested amount
from their balance
```

```
{
    var amount = document.getElementById("amount").value;
    amount = parseFloat(amount)
    amount = amount.toFixed(2);
    if (amount > balance)
    {
        alert("You do not have enough funds in your balance.  Please try a smaller
amount");
        return false;
    }
    redeem = confirmRedeem();
    if(redeem == false)
    {
        return false;
    }
}

function topUpFormInput()//display elements for card payments or payapl payments
{

    if(document.getElementById("payBy2").checked)
    {
        document.getElementById("cardRow").style.display = "none";
        document.getElementById("cvvRow").style.display = "none";
        document.getElementById("paymentCards").required = false;
        document.getElementById("cvv").required = false;
    }

    else
    {
        document.getElementById("cardRow").style.display = "block";
        document.getElementById("cvvRow").style.display = "block";
        document.getElementById("paymentCards").required = true;
        document.getElementById("cvv").required = true;
    }
}

function confirmTopUp()//confirm the decison to top up balance
{
    var amount = document.getElementById("amount").value;
    var result = confirm("Are you sure you wish to top up your balance by €" + amount
+ ".  Press Ok to continue or Cancel to return.");
    return result;
}

function addAdminValidation()//validates the new admin form
{

    var pass1 = document.getElementById("password").value;
    var pass2 = document.getElementById("cpassword").value;

    if (pass1 != pass2)
    {
        document.getElementById("passMatch1").className = "error"
        document.getElementById("passMatch2").className = "error"
        document.getElementById("passMatch1").innerText = "Passwords do not match";
        document.getElementById("passMatch2").innerText = "Passwords do not match";
        return false;
    }
    else
    {
        document.getElementById("passMatch1").className = ""
```

```
        document.getElementById("passMatch2").className = ""
        document.getElementById("passMatch1").innerText = "";
        document.getElementById("passMatch2").innerText = "";
    }
    var error = document.getElementById("userError").innerText;
    if (error != "")
    {
        return false
    }
}

function newGameValidation()//ensures that newly created games have an overall chance
value of 1
{
    var prize1 = parseFloat(document.getElementById("prize1Chance").value);
    var prize2 = parseFloat(document.getElementById("prize2Chance").value);
    var prize3 = parseFloat(document.getElementById("prize3Chance").value);
    var prize4 = parseFloat(document.getElementById("prize4Chance").value);
    var noWin = parseFloat(document.getElementById("notAWin").value);

    var overall = prize1 + prize2 + prize3 + prize4 + noWin;

    if (overall != 1)
    {
        document.getElementById("chanceError").innerText = "All chances do not add up
to 1.  Please check and re-enter the prize chances";
        document.getElementById("chanceError").className = "error";
        return false;
    }
}

function calcNoWinChance()//when an admin is adding or modifying games, calculates the
chance a card will not be a winner based on prize chances
{
    var prize1 = document.getElementById("prize1Chance").value;
    var prize2 = document.getElementById("prize2Chance").value;
    var prize3 = document.getElementById("prize3Chance").value;
    var prize4 = document.getElementById("prize4Chance").value;
    prizes = [prize1, prize2, prize3, prize4]

    var noWin = 1
    for (var i = 0; i < prizes.length; i++)
    {
        noWin -= prizes[i]
    }

    noWin = noWin.toFixed(2);
    noWin = noWin.toString();
    document.getElementById("notAWin").value = noWin;
}

function confirmRedeem()//confirm the decison to top up balance
{
    var amount = document.getElementById("amount").value;
    var result = confirm("Are you sure you wish to redeem €" + amount + " to your
account.  Press Ok to continue or Cancel to return.");
    return result;
}
```

4.3.2 AJAXCalls.js

This file contains JavaScript AJAX calls for asynchronously getting data for the application.

```
function checkUser(user)//AJAX call to check if a given username exists
{
    var req = new XMLHttpRequest();
    var sPath = window.location.pathname;
    var sPage = sPath.substring(sPath.lastIndexOf('/') + 1);
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200)
         {
            var response = JSON.parse(req.responseText);
            if (!response.exists && sPage == "buyCards")
            {
                document.getElementById("userError").style.backgroundColor =
              "#EB4141";
                document.getElementById("userError").innerText = "This user does not
exist.  Please try again";
                return false;
            }
            else if (response.exists && sPage == "register")
            {
                document.getElementById("userError").style.backgroundColor =
       "#EB4141";
                document.getElementById("userError").innerText = "This username
already exists.  Please try another";
                return false;
            }
            else {
                document.getElementById("userError").style.backgroundColor = "White";
                document.getElementById("userError").innerText = "";
            }
        }
    }

    req.open("POST", "/checkUser");
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('user=' + user);
}


function reveal(panel, id)//Makes scratch card panels disappear.  Once all are gone,
makes an AJAX call to redeem the card in the database and add funds to user balance if
neccessary
{
    document.getElementById(panel).hidden = true;
    var checkHidden = []
    checkHidden.push(document.getElementById("panel1").hidden);
    checkHidden.push(document.getElementById("panel2").hidden);
    checkHidden.push(document.getElementById("panel3").hidden);
    checkHidden.push(document.getElementById("panel4").hidden);
    checkHidden.push(document.getElementById("panel5").hidden);
    checkHidden.push(document.getElementById("panel6").hidden);

    for (var i = 0; i < checkHidden.length; i++) {
        if (checkHidden[i] == false)
        {
            return false;
        }
```

```
    }

    var req = new XMLHttpRequest();
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200) {
            var response = JSON.parse(req.responseText);
            if (response.prize != '') {
                alert("Congratulations, you have won €" + response.prize + ". This
prize will now be credited to your account balance ");
            }
            else {
                alert("Sorry, that card was not a winner.  Please try again");
            }
        }
    }

    req.open("POST", "/cardRedeemed");
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('id=' + id);
}

function getPanels(id)//AJAX call to retrieve panel list from server
{
    panels = []
    var req = new XMLHttpRequest()
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200)
        {
            var response = JSON.parse(req.responseText);
            panels = response.card
        }
    }
    req.open("POST", "/getCard", false);
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('id=' + id);
    return panels
}

function calcPrice()//calculates the price of selected amount of cards
{
    var type = document.getElementById("cardTypes").value;
    var req = new XMLHttpRequest();
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200)
        {
            var response = JSON.parse(req.responseText);
            var price = response.price;
            var quantity = parseFloat(document.getElementById("quantity").value);
            var total = price * quantity;
            total = total.toFixed(2);
            total = total.toString();
            document.getElementById("price").value = '€' + total;

        }
    }
    req.open("POST", "/getCardPrice")
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('type=' + type);
}

function getCardType(id)//AJAX call to get the type of card about to be redeemed
{
```

```
    var game = ""
    var req = new XMLHttpRequest();
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200)
        {
            var response = JSON.parse(req.responseText);
            game = response.cardType;
        }
    }
    req.open("POST", "/getCardType", false);
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('id=' + id);
    return game
}

function drawCard(id)//uses HTML canvas to draw card design by using AJAX call to
retrive card's prize
{
    panelArray = getPanels(id);
    cardType = getCardType(id);
    var x = 70;
    var y = 50;

    if (cardType == "Standard")
    {
        design = "#FF0000"
    }
    else if (cardType == "Premium")
    {
        design = "#003399"
    }
    else
    {
        design = "#00CC00"
    }

    var canvas = document.getElementById("card");
    var ctx = canvas.getContext("2d");
    ctx.fillStyle = design;
    ctx.fillRect(0, 0, 600, 300);
    ctx.font = "15px Engravers MT";
    ctx.fillStyle = "white";
    ctx.textAlign = "left";
    ctx.fillText("ClickNWin", 350, 50);
    ctx.fillText(cardType + " Game", 350, 100);
    ctx.fillText("Great Prizes", 350, 150)

    if (cardType == "")
    {
        var canvas = document.getElementById("card");
        var ctx = canvas.getContext("2d");
        ctx.fillStyle = "#000000";
        ctx.fillRect(0, 0, 600, 300);
        ctx.font = "15px Engravers MT";
        ctx.fillStyle = "white";
        ctx.textAlign = "left";
        ctx.fillText("No card to redeem", 350, 50);
    }


    while (panelArray.length > 0)
    {
```

```
        pick = Math.floor(Math.random() * (panelArray.length)) + 0;
        ctx.fillText("€" + panelArray[pick], x, y);
        panelArray.splice(pick, 1);
        y += 100
        if (panelArray.length == 3)
        {
            x += 120;
            y = 50;
        }
    }
}

function checkAdmin()//AJAX call to check a given admin username is not already taken
{
    var user = document.getElementById("username").value;
    var req = new XMLHttpRequest();
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200) {
            var response = JSON.parse(req.responseText);
            if (response.exists == true)
            {
                document.getElementById("userError").innerText = "Username already
exists.  Try another";
                document.getElementById("userError").className = "error";
            }
            else
            {
                document.getElementById("userError").innerText = "";
                document.getElementById("userError").className = "";
            }

        }
    }

    req.open("POST", "/checkAdmin");
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('user=' + user);
}

function checkGame()//Ajax Call to check that a given game name is not already taken
{
    var game = document.getElementById("gameName").value;
    var req = new XMLHttpRequest();
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200) {
            response = JSON.parse(req.responseText)
            if (response.exists == true)
            {
              document.getElementById("gameError").innerText = "Name already taken.
              Please select another"
                document.getElementById("gameError").className = "error"
            }
            else
            {
                document.getElementById("gameError").innerText = ""
                document.getElementById("gameError").className = ""
            }
        }
    }
    req.open("POST", "/checkGame");
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('game=' + game);
```

```
}

function getGame(game)//AJAX to retrieve game information from the database
{
    var data = []
    var req = new XMLHttpRequest();
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200)
        {
            response = JSON.parse(req.responseText)
            data = response.data;
            document.getElementById("gameName").value = data[0]
            document.getElementById("gamePrice").value = data[1]
            document.getElementById("prize1").value = data[2]
            document.getElementById("prize1Chance").value = data[3]
            document.getElementById("prize2").value = data[4]
            document.getElementById("prize2Chance").value = data[5]
            document.getElementById("prize3").value = data[6]
            document.getElementById("prize3Chance").value = data[7]
            document.getElementById("prize4").value = data[8]
            document.getElementById("prize4Chance").value = data[9]
            calcNoWinChance()
            document.getElementById("sButton").disabled = false;
        }
    }

    req.open("POST", "/getGame");
    req.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    req.send('game=' + game);
}
```

4.4 CSS Code

4.4.1 Site.css

The site.css file was auto generated by Visual Studio but was heavily modified by the project team to add any css code the application needed. Code that was modified by the team will be yellow highlighted.

```
body {
    padding-top: 50px;
    padding-bottom: 20px;
    background-size:50% auto;
    background:url('../../static/money.jpg') no-repeat;
    background-position-y:-340px;
    background-size: cover;
    text-shadow: 40px;
}

.body-content {
    padding-left: 5px;
```

```
    padding-right: 5px;
}


input,
select,
textarea {
    max-width: 280px;
}

.field-validation-error {
    color: #b94a48;
    position:fixed;
    top:0;
    left:0;
}

.field-validation-valid {
    display: none;
}

input.input-validation-error {
    border: 1px solid #b94a48;
}

input[type="checkbox"].input-validation-error {
    border: 0 none;
}

.validation-summary-errors {
    color: #b94a48;
}

.validation-summary-valid {
    display: none;
}

p.loginphome{
    font-size:large;
}

footer{
    position:absolute;
    top: 550px;
}

div.accountTab{
    border-right:solid 1px #000000;
    border-bottom:solid 1px #000000;
    height:300px;
    width:130px;
}

div.mainp{
    font-weight:900;
    text-shadow: 0px 0px 3px white;
    color:darkblue;
}
h1{
    font-weight:900;
    text-shadow: 0px 0px 3px white;
    color:darkblue;
```

```
}
label.form-label{
    font-weight:900;
    text-shadow: 0px 0px 3px white;
    color:darkblue;
}

table.cardsTable{
    font-weight:900;
    text-shadow: 0px 0px 3px white;
    color:darkblue;
}

span.error{
    padding: 2px;
    margin-top: 2px;
    margin-bottom: 0px;
    list-style: none;
    border: 1px solid transparent;
    border-radius: 3px;
    background-color:red;
}

button.cardPanel{
    color: gray;
    height:40px;
    width: 70px;
    z-index:2;
    position:absolute;
}
```

5.0 Mobile Code

The following code was used to build the basic mobile version of ClickNWin using a webview class to display the web applicaton on a mobile device.

```
package com.example.geoff.clicknwinmobile;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebChromeClient;
import android.webkit.WebViewClient;
import android.webkit.WebView;
import android.webkit.WebSettings;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        WebView myWebView = (WebView) findViewById(R.id.webview);
        myWebView.setWebChromeClient(new WebChromeClient());
        WebSettings webSettings = myWebView.getSettings();
        webSettings.setJavaScriptEnabled(true);
        myWebView.loadUrl("https://clicknwin.pythonanywhere.com");
```

```
        myWebView.setWebViewClient(new WebViewClient(){
            public boolean shouldOverrideUrlLoading(WebView view, String url){
                WebView myWebView = (WebView) findViewById(R.id.webview);
                WebSettings webSettings = myWebView.getSettings();
                webSettings.setJavaScriptEnabled(true);
                myWebView.loadUrl("https://clicknwin.pythonanywhere.com");
                view.loadUrl(url);
                return false;
            }
        });
    }
}
```