Universidad Simón Bolívar

Laboratorio de Ingeniería del Software

Tarea 3

Autores:

Carlos Vazquez 12-

Rafael Blanco 13-10156

Enero, 2017

INTRODUCCIÓN

Con los avances tecnológicos de la nueva era se han creado infinidad de facilidades para la labores diarias del ser humano, de hecho, uno de las principales razones del desarrollo de un proyecto tecnológico es automatizar y facilitar alguna acción de interés para las personas. En esta oportunidad se requirió desarrollar un pequeño programa con estructuras que simulan la función de una tarjeta de débito.

Para cumplir con este objetivo, se emplearon las múltiples funciones y facilidades del lenguaje de programación Python 3.4, complementado con la IDE de *Eclipse/Pydev* y empleando la librería de Python *PyUnit* para realizar los distintos casos de prueba necesarios para verificar el buen funcionamiento del programa.

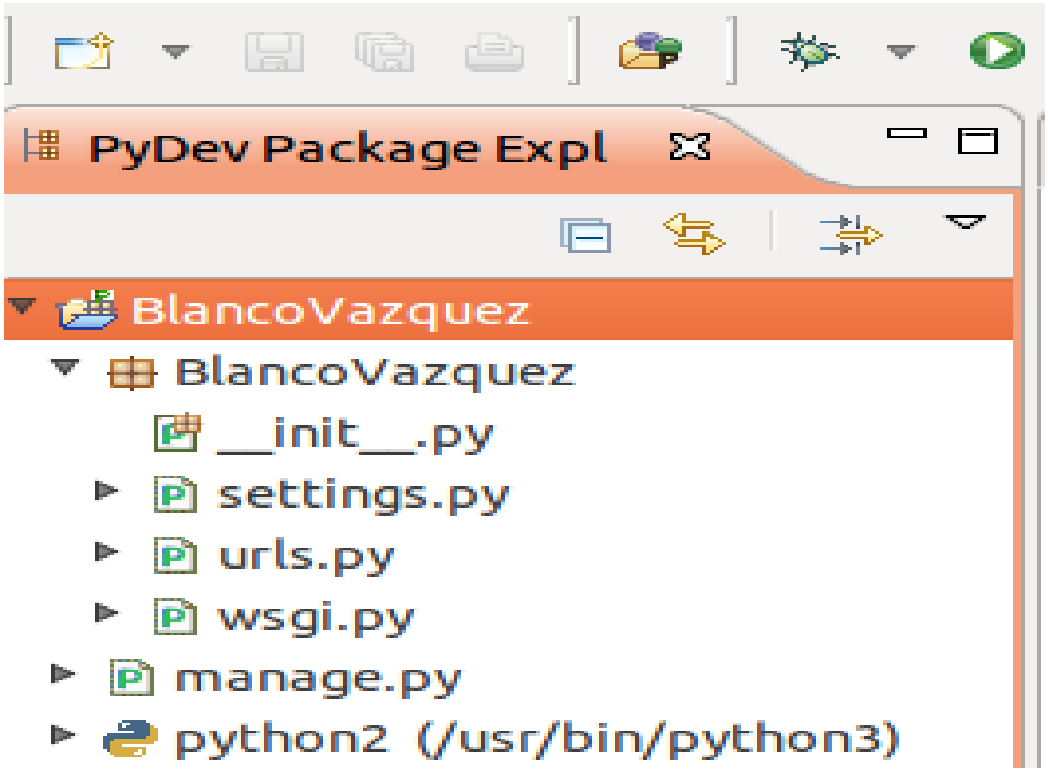
TAREA 3

Se requirió el desarrollo de un mini programa que contiene una estructura llamada *billetera_electronica,* cuya estructura es bastante similar al manejo que hacen los bancos para controlar el uso de tarjetas de débito de sus usuarios y brindar ciertas consultas de interés para los mismos. El desarrollo de la solución fue sencillo, en principio se creó una estructura llamada *LinkedList* que representa una lista enlazada donde se guardará otra estructura llamada *Nodo*, éstos contienen la información de una transacción cualquiera, es decir, una fecha, un monto y la identificación del establecimiento desde donde se está realizando la operación. Luego de esto se creó la estructura *billetera_electronica,* ésta contiene una identificación propia, el nombre, apellido y cédula del dueño, una clave para realizar consumos y dos estructuras *LinkedList* en donde serán almacenados los consumos y recargas asociados a la billetera; de modo que, cada vez que se realiza una recarga, se almacenan los datos en una estructura *Node* y ésta se añade a la *LinkedList* correspondiente, luego se actualiza el saldo. Por otra parte, si se realiza un consumo, en principio se verifica si el pin que introdujo el usuario al realizar el consumo corresponde al de la billetera, luego se verifica que su saldo sea mayor o igual al consumo a realizar, y, en caso de que estas dos condiciones se cumplan, se almacenan los datos en una estructura *Node* y ésta se añade a la *LinkedList* correspondiente, luego se actualiza el saldo.

Esta asignación fue trabajada en parejas, en principio se discutió la posible solución, la manera de implementarlo, los casos bordes a tomar en consideración. Luego los integrantes se reunieron para crear todas las estructuras y funciones requeridas para el funcionamiento de la *billetera_electronica,* asimismo, cada una de ellas fueron probadas independientemente durante el desarrollo, implementando así el método TDD; finalmente el trabajo fue dividido y se trabajó de manera independiente, de modo que, un integrante tenía la responsabilidad de escribir el informe y el otro debía formular las pruebas finales del programa. Cada uno de los integrantes empleó aproximadamente 1 hora de trabajo en solitario y 2 horas de trabajo en pareja.

Las nuevas herramientas de trabajo, como la IDE *Eclipse/Pydev* representan un entorno muy cómodo de trabajo el cual se espera aprovechar al máximo durante todo este período y mientras sea necesario trabajar con lenguajes de programación soportados por este entorno. Por otra parte, la utilización de *Git* nos facilitó la tarea de compartir el código para la modificación individual del mismo.

Como parte de nuestra labor en la tarea era instalar el framework Django para poder trabajarlo con la herramienta Eclipse, por lo cual se procedió a crear un nuevo proyecto de entorno Pydev/Django con interpretador gramatical Python 3.0-3.5, la muestra de que esto fue realizado se encuentra en la siguiente imagen:



CONCLUSIÓN

El TDD es un método bastante eficiente y practicado para el desarrollo de cualquier software a nivel mundial; éste nos permite probar cada una de las estructuras y funciones de un software individualmente, con el propósito de eliminar los errores funcionales que el mismo pueda tener. Un método eficiente y cómodo para realizar estas pruebas es utilizar la librería *PyUnit,* la cual contiene todas las herramientas necesarias para realizar cómodamente los casos de prueba.