INSTITUTO TECNOLÓGICO DE VERACRUZ  Miguel Ángel de Quevedo 2779

Manual técnico servidor

Luis Enrique Sosa Hernández

Julio De Jesús PérEz Castro

Alan Giovanni Cano Caballero

Jonathan Emmanuel Palacios gOMEZ

**Contenido**

[**Introducción** 2](#_Toc452962719)

[**ESPECIFICACIONES TECNICAS** 3](#_Toc452962720)

[**Objetivo y alcances del sistema** 3](#_Toc452962721)

[**Diccionario de datos** 4](#_Toc452962722)

[**Diagrama de la aplicación** 7](#_Toc452962723)

[**Especificación de clases** 8](#_Toc452962724)

# **Introducción**

Este manual fue realizado con el propósito de facilitar el entendimiento del programa “Casino Pokemón” para desarrollares o cualquier persona interesada en mejorar o brindar soporte a la misma.

Este manual pretende dejar clara la estructura de la aplicación así como el funcionamiento de cada una de sus partes.

Se emplearan gráficos para facilitar la compresión del mismo así como se manejara por secciones para facilitar el desplazamiento entre páginas.

# **ESPECIFICACIONES TECNICAS**

**Sistema Operativo:** Windows 7 o superior. (Recomendado), GNU/Linux Ubuntu 12.04 (recomendado).

**Lenguaje de programación:** Java JDK 1.8.

**Servidor de aplicaciones:** Servidor web apache 2.0, servidor.

**IDE:** Netbeans.

# **Objetivo y alcances del sistema**

* Mantener el control de clientes que ingresan al juego
* Llevar un registro de las ganancias obtenidas
* Permitir la conexión vía internet cliente-servidor mediante usuario y contraseña
* Tener un registro de usuarios
* Permitir la recuperación de contraseñas mediante un correo electrónico
* Proveer de mensajes de confirmación al usuario para mejorar la experiencia

El servidor funciona dentro de un área local, por medio de WI-FI o por medio de Ethernet capaz de albergar cientos de usuarios y darle servicio a los mismos gracias a la tecnología de hilos implementada en el lenguaje java.

Capacidad de guardar un sin fin de registros de usuario gracias a la tecnología XML.

# **Diccionario de datos**

|  |  |
| --- | --- |
| Server | |
| Thread servidor | Hilo para implementar tecnología de hilos para la conexión |
| JButton btnPersonalizado | Botón para iniciar el servidor |
| JPanel pnl | Panel principal |
| static JTextField txtBolsa | Etiqueta para visualizar la cantidad de dinero total |
| static JTextField txtGanancia | Etiqueta para visualizar la cantidad de dinero ganada |
| JLabel lblServidor | Etiqueta para visualizar el estado del servidor |
| JLabel lblConexion | Etiqueta para visualizar nuevas conexiones entrantes |
| JLabel img  JLabel img2  JLabel img3  JLabel img4 | Etiquetas para añadir imágenes al frame principal |
| ImageIcon dratini  ImageIcon chango  ImageIcon hitmontop  ImageIcon haunter | Imágenes agregadas al frame principal |
| ImageIcon imgNormal  ImageIcon imgPasar  ImageIcon imgPulsado  ImageIcon imgNormal2  ImageIcon imgPasar2 | Imágenes agregadas al botón principal |
| JMenuItem salir  JMenu mnu  JMenuBar braMenu | Menú despegable añadido al frame principal  Salir funciona para cerrar el programa. |
| JTextField txtFecha | Etiqueta para mostrar la fecha actual |
| Date date = new Date() | Método para retornar la fecha actual |
| ServerSocket ss | Conexión mediante socket |
| public void run() | Hilo de la conexión |
| public void actionPerformed(ActionEvent e) | Método de acción del botón |

|  |  |
| --- | --- |
| ServidorHilo | |
| private Socket socket | Socket de la conexión |
| private DataOutputStream dos  private DataInputStream dis | Lectura y escritura de envió de datos |
| private int idSessio | Identificador de sesión |
| long BolsaAcomulada  long Ganancias | Variables para acumular valores de la bolsa y ganancias |
| String accion  String usuario  String password | Variables para recibir los parámetros enviados por el cliente |
| public synchronized void desconnectar() | Método para desconectar al cliente |

|  |  |
| --- | --- |
| Registro | |
| public void Registro(String nombre,String clave) | Constructor de la clase que recibe parámetros |
| ArrayList<Usuarios> listUsuarios | Lista que almacena los usuarios registrados |
| boolean centi | Bandera booleana |
| agregarUsuario(nombre,clave) | Agregar un nuevo usuario |
| public boolean estaregistrado() | Método booleano para saber si el usuario ya está registrado |
| public boolean UsuarioOcupado() | Método booleano para saber si el nombre de usuario esta en uso |

|  |  |
| --- | --- |
| XMLGenerator | |
| Public static String obtenerNodoValor(String strTag, Element eUser) | Método para obtener nodos |
| public void agregarUsuario(String nombre, String contraseña) | Método para agregar usuarios al XML |
| ArrayList<Usuarios> listaUsuarios = new ArrayList<>() | Lista temporal para almacenar usuarios recuperados |

|  |  |
| --- | --- |
| Usuarios | |
| String nombre | Variable que almacena y retorna el nombre de usuario |
| String contraseña | Variable que almacena y retorna la contraseña de usuario |

|  |  |
| --- | --- |
| Correo | |
| public boolean Esuncorreo(String correo) | Método para validar que el dato ingresado sea un correo |
| public String Mandarcontrasena(String usuario) | Método para enviar contraseña al usuario ingresado |
| boolean centi | Bandera booleana |
| String pass | Variable para almacenar la contraseña |
| ArrayList <Usuarios> listUsuarios | Lista temporal para almacenar usuarios |
| public void mandarCorreo(String correo,String pass) | Método para enviar el correo con la contraseña |

|  |  |
| --- | --- |
| BD | |
| <BD> </BD> | Etiquetas para inicio y fin del documento |
| <user></user> | Etiquetas de usuario |
| <nombre></nombre> | Etiquetas de nombre |
| <contraseña></contraseña> | Etiquetas de contraseña |

# **Diagrama de la aplicación**

Server

# **Especificación de clases**

Server

Su principal función es fungir como clase y frame principal donde se ejecuta el servidor y muestra y actualiza los datos en sus etiquetas.

ServidorHilo

Clase donde se implementa la conexión cliente-servidor y sirve como medio para controlar las conexiones, mandar y recibir datos de inicio de sesión, recuperación de contraseña y las transacciones de efectivo.

Correo

Su principal función consiste en implementar métodos que son capaces de retornar una contraseña a un usuario mediante el correo electrónico.

Registro

Clase cuya función se basa en gestionar si el usuario está ocupado, si ya está registrado, guardar un nuevo registro.

Usuario

Su única función es fungir como nodo para la clase registro.

XMLGenerator

Clase que genera y guarda los datos de registro de usuarios mediante métodos implementados en un archivo XML.

BD

Su principal función es servir como archivo de registro de campos (usuario y contraseña).