**Financial Chat**

Resumen.

Servidor: Se crea un proceso infinito utilizando sockets para recibir peticiones y enrutarlas hacia todos los clientes conectados, estos mensajes se muestran en una interfaz de chat grupal.

Robot: Se crea un micro proceso infinito que escucha la cola(‘bot’) del rabbitMQ para ejecutar la acción descrita en el ejercicio y devolver la respuesta, para lo cual se genera un UUID con cada petición y así el invocador en la cola recibe la respuesta misma de petición que solicito.

Base de Datos: Se genera una base de datos en PostgreSQL para almacenar perfiles de usuario, y guardar los mensajes que los clientes escriben, exceptuando la invocación al Robot. Para mantener la base online se creo una instancia en los servicios de AWS, con dirección (chat.cnyrildivagz.us-east-2.rds.amazonaws.com)

Interfaz Grafica: Se realiza una sencilla interfaz grafica utilizando la librería QT.

Todas las configuraciones se manejan desde un Config, lo cual permite parametrizar el sistema a cualquier ambiente.

**Pre requisitos.**

1 - Tener instalado Python 3.6

2 - Tener instalado y en ejecución RabbitMQ

Ejecución

1 - Descomprimir el archivo “financial\_chat.zip”

2 - Ejecutar "01 - Start Server.cmd" inicia el proceso servidor de chat por sockets.

3 - Ejecutar "02 - Start Bot.cmd" inicia el proceso Robot (Bot) escuchando por la cola "bot" del rabbitMQ para ejecutar la transacción de obtener el stock.

4 - Ejecutar "03 - Execute.cmd" Iniciara en un browser una pantalla de autenticación, al poner los datos correctamente se abrirá inmediatamente una pantalla con el Chatroom.

* 4.1 - Existen dos usuarios predefinidos: user: noel pass:123. user:invitado pass:admin
* 4.2 - Se debe ejecutar "03 - Execute.cmd" para abrir varios clientes de chat.

Muchas gracias por la oportunidad.

Saludos

**Financial Chat**

Summary.

Server: An infinite process is created using sockets to receive requests and route them to all connected clients, these messages are displayed in a group chat interface.

Robot: An infinite micro process is created that listens to the queue ('bot') of the rabbitMQ to execute the action described in the exercise and return the response, for which a UUID is generated with each request and thus the invoker in the queue receives the same request response that I request.

Database: A database is generated in PostgreSQL to store user profiles, and save the messages that the clients write, except the invocation to the Robot. To maintain the base online, an instance was created in the AWS services, with address (chat.cnyrildivagz.us-east-2.rds.amazonaws.com)

Graphic Interface: A simple graphic interface is made using the QT library.

All configurations are managed from a Config, which allows parameterizing the system to any environment.

Pre requirements.

1 - Having Python 3.6 installed

2 - Have RabbitMQ installed and running

Execution

1 - Unzip the file "financial\_chat.zip"

2 - Executing "01 - Start Server.cmd" starts the chat server process by sockets.

3 - Executing "02 - Start Bot.cmd" starts the Robot (Bot) process by listening for the "bot" queue of the rabbitMQ to execute the transaction to obtain the stock.

4 - Execute "03 - Execute.cmd" It will initiate in a browser an authentication screen, when entering the data correctly a screen with the Chatroom will immediately open.

• 4.1 - There are two predefined users: user: noel pass: 123. user: guest pass: admin

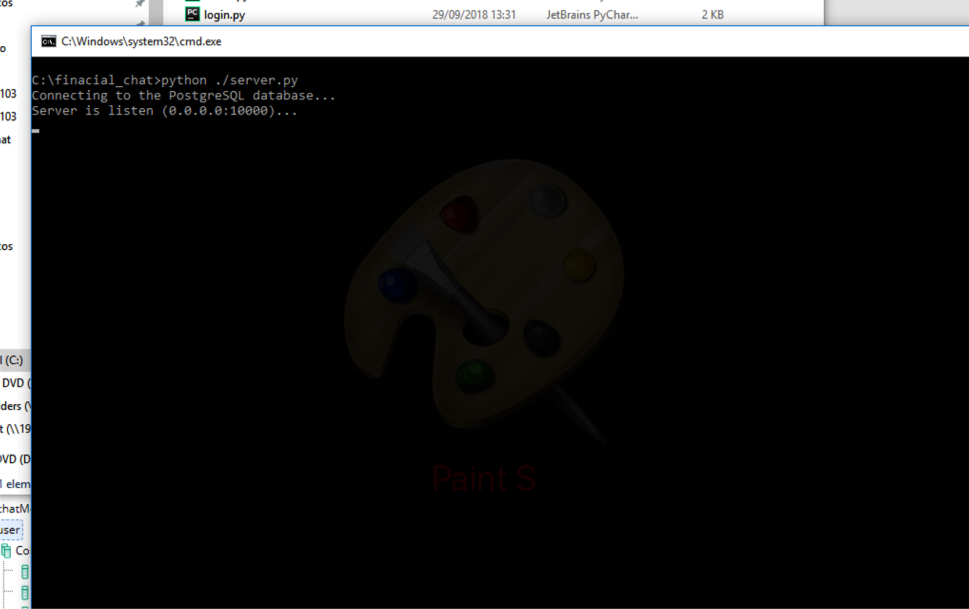
• 4.2 - You must execute "03 - Execute.cmd" to open several chat clients.

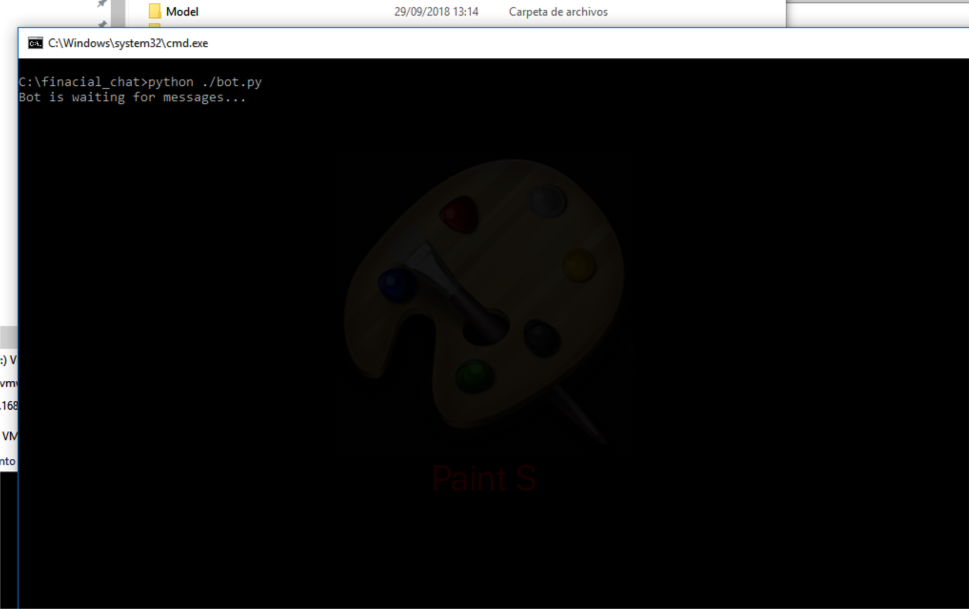
Thank you very much for the opportunity.

regards

Anexos:

01 - Start Server.cmd



02 - Start Bot.cmd 

03 - Execute.cmd (Client)

