|  |
| --- |
| Banco Itaú |
| Test Case - Arquitetura de software |
|  |
|  |
| **Bruno Polizelli Castilho** |
| **02/06/2019** |

|  |
| --- |
|  |

Conteúdo

[Descrição da demanda: 3](#_Toc10411075)

[Descrição da solução: 4](#_Toc10411076)

[Divisão de componentes 5](#_Toc10411077)

[Componentes de apresentação. 5](#_Toc10411078)

[App 5](#_Toc10411079)

[Sistema Web para operadores. 5](#_Toc10411080)

[Componentes de negócio. 5](#_Toc10411081)

[ChatRoom 5](#_Toc10411082)

[ChatBot 5](#_Toc10411083)

[Log 5](#_Toc10411084)

[OperatorQueue 6](#_Toc10411085)

[Transactions 6](#_Toc10411086)

[Login 6](#_Toc10411087)

[Notification 7](#_Toc10411088)

[BI Batch 7](#_Toc10411089)

[Fluxos Funcionais 8](#_Toc10411090)

[Execução de transação bancária por ChatBot 8](#_Toc10411091)

[Tentativa de execução de transação indisponível por ChatBot. 11](#_Toc10411092)

[Aplicação Web 12](#_Toc10411093)

[Robô de notificações. 13](#_Toc10411094)

## Descrição da demanda:

Você é um Arquiteto de Soluções e foi designado a atender um grande projeto de Transformação Digital do Banco, onde o objetivo é construir um novo APP para melhorar a experiência dos clientes ofertando todos os produtos disponíveis através deste novo canal, a usabilidade do APP deve ser baseada no Google Allo, e pode ser iniciada pelo usuário ou pelo próprio APP ou via Notificação.

Alguns requisitos importantes foram solicitados, tais como interação conversacional totalmente diferenciada para o Cliente especifico, Camada de segurança, Notificação, Avaliação do comportamento e preferências do Cliente e propor iterações bancárias de acordo com suas preferências.

Com exceção dos serviços de negócio, que já são ofertados via API (Gateway Interno), você deve desenhar todas as features necessárias para atender os requisitos solicitados.

Considere no Escopo apenas os Serviços de Negócio abaixo:

* Consulta Saldo
* Extrato
* Transferência
* Bloqueio temporário de Cartão
* Alteração de Limite de cartão de Crédito

Imagine que você irá apresentar sua proposta em uma banca de projetos e será avaliada por Arquitetura, Engenharia e Desenvolvimento.

## Descrição da solução:

A solução foi pensada para utilizar como base o framework Java Spring, utilizando Spring Boot para desenvolvimento dos componentes de micro serviço, Spring JMS para controle de filas e Spring Batch para a execução de robôs.

Para banco de dados, foram considerados para atender as necessidades a construção de uma estrutura em MongoDB para armazenar as interações entre clientes e atendentes em um formato de BigData escalável para servir de de insumo para a criação de regras de negócio de análise das interações com os clientes.

Também para armazenar informações estruturadas, como a relação entre as palavras chaves digitadas pelos clientes e a transação correspondente, o ideal é utilizar um banco de dados relacional (SqlServer, Oracle, MySQL, etc).

O componente foi pensado para solicitar a senha do cliente somente para as transações bancárias, assim para informações simples como qual e a agência ou caixa eletrônico mais próximo do meu endereço atual, qual e o telefone da central de relacionamentos, etc, o cliente não precisa digitar a sua senha, podendo ser atendido pelo ChatBot ou por um atendente humano.

A solução também prevê a criação de uma aplicação web para ser utilizada por atendentes para quando o ChatBot não consegue encontrar uma resposta para a solicitação.

Essa aplicação deve consumir da fila de mensagens JMS.

A aplicação permite a troca de mensagens entre o cliente e o atendente. Quando o atendente entende a transação que o cliente necessita, ele pode selecionar uma das transações parametrizadas.

O App recebe a transação assim como ocorreria no fluxo realizado pelo ChatBot e o resto do processo ocorre do mesmo modo.

Para o controle de envio de notificações baseado na atividade do cliente, foi especificado um robô em Spring Batch.

O robô seria parametrizado para rodar em determinados horários do dia e ele possui uma interface chamada IBiRule.

O robô executa automaticamente todas as implementações da interface. Os métodos dessa implementação são responsáveis por verificar quais clientes se enquadram nas regras para receber as notificações, gerar conteúdo da notificação e chamar o micro serviço que solicita o envio da notificação por Push para iOS ou Android.

## Divisão de componentes

## Componentes de apresentação.

### App

Aplicativo para celulares com versões em Android e iOS.

Responsável pela interação com o usuario e para o recebimento de notificações.

### Sistema Web para operadores.

Sistema Web criado utilizando Spring MVC.

Sistema responsável por consumir a fila de solicitação de atendimento humano via chat.

Possui integração para enviar e receber mensagens do App.

## Componentes de negócio.

### ChatRoom

Componente de micro serviços a ser construído utilizando Spring Boot.

Funciona como controlador das interações do App com os demais serviços de backend.

### ChatBot

Componente responsável para analisar as mensagens recebidas dos clientes e localizar possíveis respostas automatizadas para as solicitações.

### Log

Componente responsável por armazenar as interações do usuário com o App.

Todas as mensagens trocadas pelo cliente e os atendentes são armazenadas em uma estrutura para big data (MongoDB).

Loga os serviços solicitados pelo cliente e seus status Atendidos ou Não atendidos.

### OperatorQueue

Componente JMS criado utilizando o Spring Boot.

Quando o cliente não consegue ter a sua solicitação atendida pelo chat bot, ele tem a opção de ser atendido por um operador humano.

Ao realizar essa solicitação, o cliente e adicionado a uma fila.

O sistema Web para atendimento consome a informação dessa fila para iniciar o atendimento humano ao cliente.

### Transactions

Componentes que executam as transações bancárias executadas pelo cliente.

Componentes compostos por micro serviços, implementando a inerface IExecuteTransaction.

Os membros dessa interface obrigatoriamente devem implementar os métodos:

* ValidateUser - Método que valida se o cliente tem permissão para executar a transação.
* ValidateParameters - Validação dos parâmetros da transação (e.g. o cliente só pode realizar transferência se possuir saldo suficiente).
* ExecuteTransaction - Chama o componente de negócio que executa a transação.

Para o exemplo proposto, seriam criados os seguinte componentes que implementam a interface acima:

* Transfer - Para realização de transferências.
* Balance - Para obtenção de saldo.
* Extract - Para obtenção do extrato da conta.
* Limit Change - Para solicitar a mudança do limite de cartão.

### Login

Componente que recebe como parâmetro o RFID do celular, o usuário e a sua senha e retorna o cliente logado e um token.

Esse token é utilizado em todas as chamadas transações bancárias executadas pelo aplicativo.

### Notification

Componente de micro serviços que realiza o envio de push notifications para o cliente.

Possui dois serviços, um para Android e um para iOS, devidamente configurados para receber textos e realizar o envio das notificações por push para os celulares dos clientes.

A interface pode ser expandida para novos tipos de notificação (SMS/Whastsapp, por exemplo) para a criação de novas implementações da interface.

### BI Batch

Robô que executa regras para verificar a necessidade de envio de notificações para os clientes

O robô, construido utilizando o framework Spring Batch, executa os métodos definidos para a interface IBiBach para todas as implementações da interface ativas.

* CheckClientsForRule - Busca os clientes que se enquadram nas regras.
* CreateRuleMessage - Cria o texto da notificação que será enviada.
* SendClientNotification - realiza a chamada do NotificationService para cada cliente selecionado com o texto criado.

Com essa estrutura genérica, podem ser criadas implementações dessa estrutura por exemplo, procurar no log de transactions por operações que foram iniciadas mas não completadas, ou então utilizar uma busca no texto de mensagens do atendimento dos clientes procurando por palavras chave como empréstimo, limite, consignado, etc, para o envio de notificações sobre novos produtos ou ofertas.

## Fluxos Funcionais

### Execução de transação bancária por ChatBot

#### Descrição do fluxo:

O usuário cria uma mensagem em texto no App, o App cria um objeto do tipo Message.

O processo e iniciado no microservice ReceiveMessage do componente ChatRoomServices.

O Componente LogServices é chamado para realizar o armazenamento da mensagem que foi enviada.

O componente ChatBotServices é chamado para realizar o parse da mensagem e procurar por palavras chave no texto.

Com base nas palavras chave, é chamado o componente que busca pelas transações associadas às palavras.

O componente retorna uma lista de transações encontradas.

O componente ChatRoomServices monta a mensagem de resposta e a envia ao app.

O cliente seleciona a transação apropriada.

O cliente preenche os parâmetros

O cliente realiza o entra com a sua senha.

A transação é enviada para o componente ChatRoomServices.

O ChatRoomServices realiza a chamada ao componente correto de transação que implementa a interface IExecuteTransaction.

O componente chama o componente de log para que seja logado banco de dados relacional a tentativa de execução de transação.

O componente IExecuteTransaction valida a permissão do usuário para executar a ação.

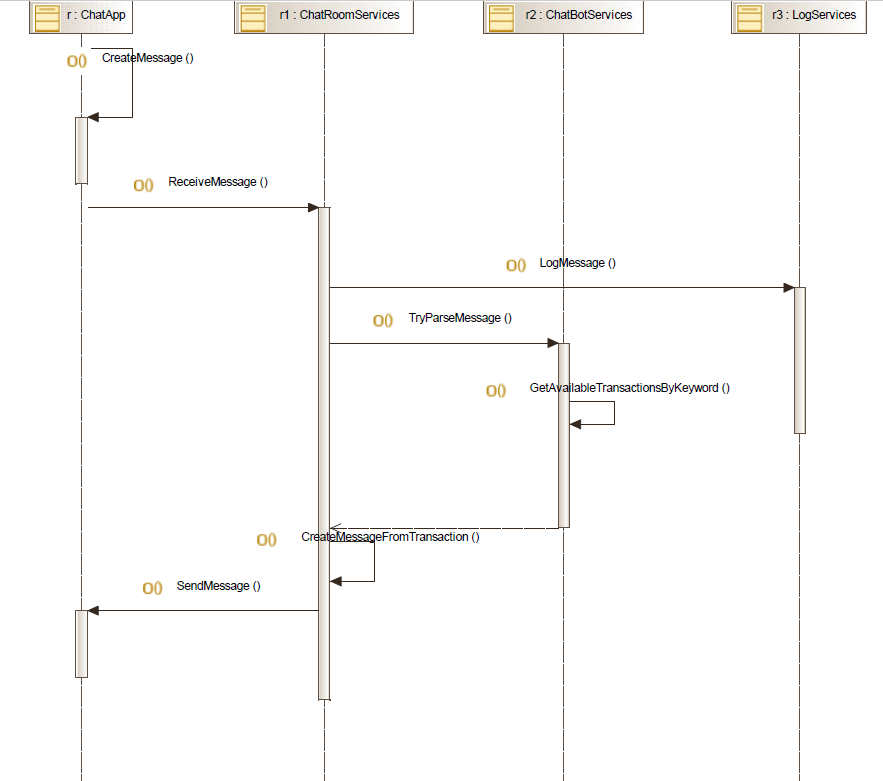
O componente IExecuteTransaction valida se o parâmetro é apropriado para a transação.

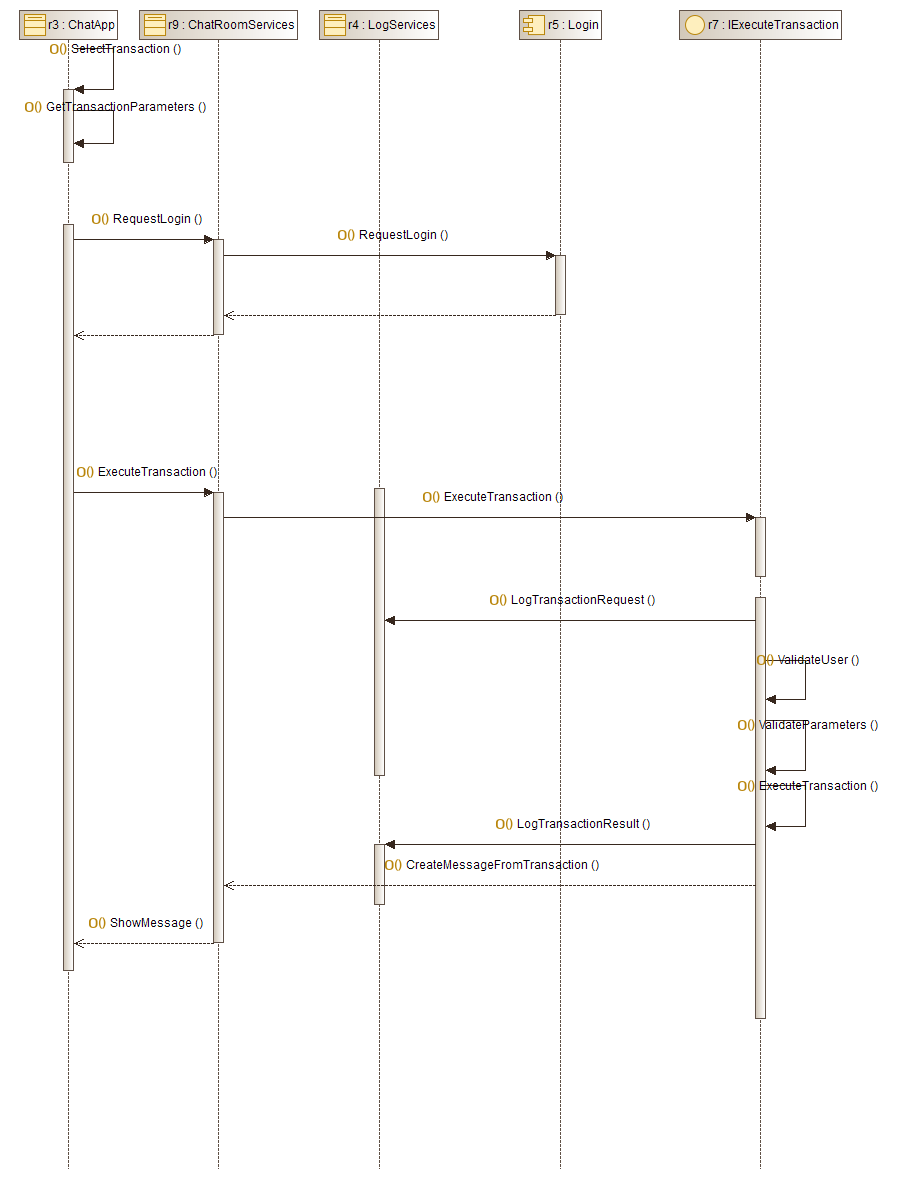
O componente chama o serviço componente de negócios.

O componente chama o componente de log para que seja logado banco de dados relacional o resultado da execução

O resultado é retornado pelas camadas até informar o cliente sobre o resultado.

#### Diagrama de sequência do fluxo:





### Tentativa de execução de transação indisponível por ChatBot.

#### Descrição do Fluxo.

O usuário cria uma mensagem em texto no APP, o App cria um objeto do tipo Message.

O processo e iniciado no microservice ReceiveMessage do componente ChatRoomServices.

O Componente LogServices é chamado para realizar o armazenamento da mensagem que foi enviada.

O componente ChatBotServices é chamado para realizar o parse da mensagem e procurar por palavras chave no texto.

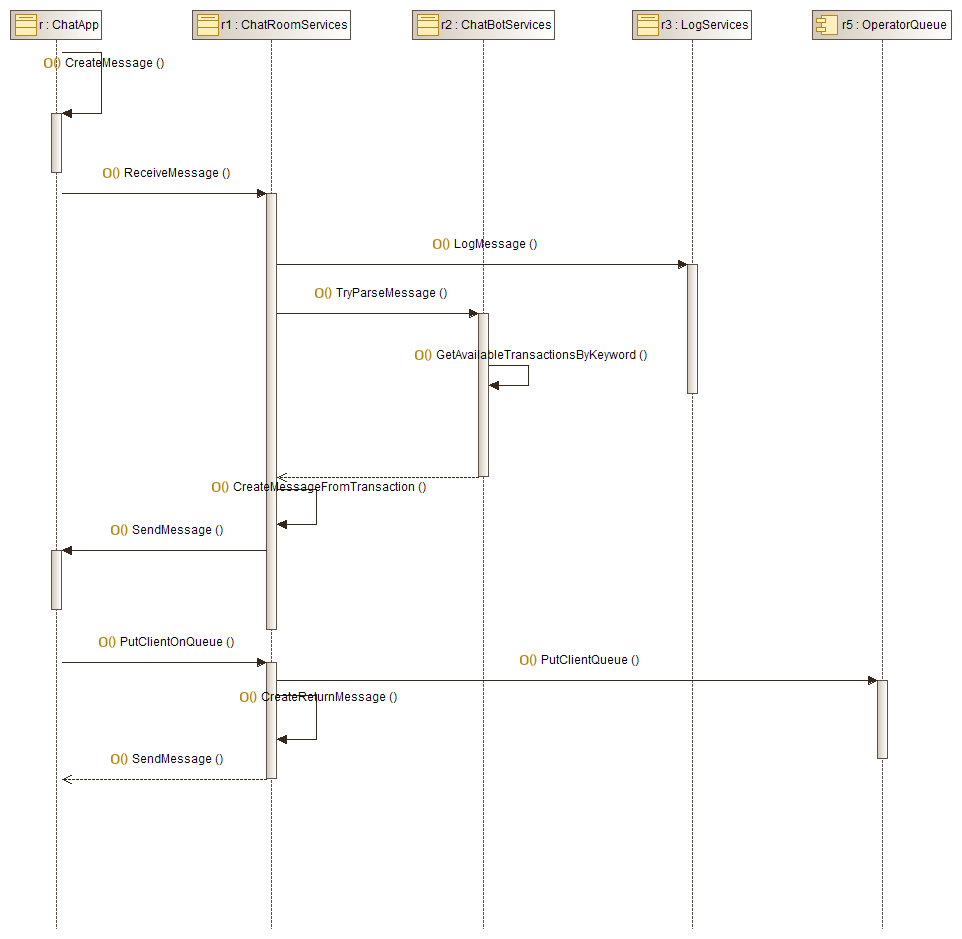
Com base nas palavras chave, é chamado o componente que busca pelas transações associadas às palavras.

O componente retorna uma lista de transações encontradas.

Caso não seja encontrada nenhuma transação para as palavras chave ou o cliente diga que a solicitação não corresponde às transações retornadas, ele tem a opção de falar com um atendente humano.

Essa solicitação é incluida em uma fila pelo componente OperatorQueue.

#### Diagrama de sequência do fluxo:



### Aplicação Web

#### Descrição do fluxo

O usuário operador se loga no sistema.

O operador sinaliza que está pronto para iniciar o atendimento.

O sistema consome a primeira mensagem da fila, que contém os dados do cliente que deseja ser atendido.

O operador avalia a mensagem enviada e inicia a troca de mensagens com o cliente.

Após a interação de troca de mensagens, o operador seleciona a transação apropriada ao cliente.

A transação é enviada para a aplicação e o fluxo continua a sua execução como no fluxo Execução de transação bancária por ChatBot.

### Robô de notificações.

#### Descrição do fluxo.

O robô inicia o seu processamento no horário parametrizado.

O robô inicia threads para cada uma das implementações da interface IBiRule.

Cada thread executa os métodos:

* CheckClientsForRule - Busca os clientes que se enquadram nas regras.
* CreateRuleMessage - Cria o texto da notificação que será enviada.
* SendClientNotification - realiza a chamada do NotificationService para cada cliente selecionado com o texto criado.