IFMG *CAMPUS* FORMIGA

CURSO: **CIÊNCIAS DA COMPUTAÇÃO**
DISCIPLINA: **SISTEMAS DISTRIBUÍDOS I**
PROF.º EVERTHON VALADÃO everthon.valadao@ifmg.edu.br

TRABALHO EM GRUPO (v1.0) VALOR: 20 PONTOS

IMPORTANTE, leia atentamente as instruções a seguir:

- Trabalho prático em grupo, a ser realizado em **trios** (três alunos).
  - Exceções devem ser previamente comunicadas e aprovadas pelo professor.
- Para a implementação do trabalho, exige-se a utilização do **middleware** JGroups (www.jgroups.org);
  - Sugere-se a utilização da IDE Netbeans e um sistema operacional GNU/Linux (ex.: Ubuntu).
- Deverá ser submetido no portal ***Google Classroom*** (https://classroom.google.com) da disciplina "*Sistemas Distribuídos (CCOMP-2020.1)*" um arquivo compactado (.zip ou .tar.gz) contendo:
  1. a implementação do trabalho: **códigos-fonte** + programa compilado (.java e .class);
  2. o arquivo **XML** contendo a configuração **personalizada** dos protocolos do JGroups a serem utilizados pelo programa;
  3. arquivo **executável** (ex.: *shellscript.sh* ou *programa.jar*) para facilitar a implantação e execução do trabalho durante a apresentação do grupo.
  4. um arquivo texto **LEIAME.txt** contendo informações de: (i) como compilar e (ii) como executar via terminal CLI o(s) programa(s) em um sistema computacional compatível com o padrão POSIX (ex.s: GNU/linux, MacOS/darwin, Windows/cygwin);
  5. um breve **relatório** (.pdf ou .docx):
     - apresentando a arquitetura do sistema, explicando e justificando as principais decisões de projeto do grupo;
     - justificando a escolha dos protocolos (JGroups) adotados e fornecendo uma análise de desempenho da pilha de protocolos configurada (tempo gasto, mensagens por segundo e vazão do test com o MPerf configurado para 100.000 mensagens);
     - apontando os pontos fortes e fracos da solução desenvolvida;

DATAS DAS ENTREGAS:

Etapa 1: funcionalidades **básicas**
até **18/10** via Google Classroom
Deverá ser agendada pelo grupo a apresentação parcial do trabalho ao professor.

Etapa 2: funcionalidades básicas e **intermediárias**
até **06/11** via Google Classroom
Deverá ser agendada pelo grupo PREVIAMENTE a apresentação do trabalho ao professor.

DESCRIÇÃO DO TRABALHO (v1.0)**:**

Deverá ser projetado um **sistema distribuído** (SD) que proveja um *Serviço de Banco Virtual* (*e-banking*). A infraestrutura do SD deverá ser provida pelo middleware JGroups, com a comunicação entre os componentes — membros do(s) *cluster*(s) — realizada através do conector JChannel (*multicast*, *anycast*, *unicast*) e do componente MessageDispatcher. O serviço de "e-banking" de cada grupo consistirá um banco autônomo, que deverá gerenciar suas contas e prover operações para movimentações financeiras entre elas.

REQUISITOS DA APLICAÇÃO:

O serviço de "e-banking" deverá permitir a abertura de uma conta por cliente (cada conta deverá possuir um identificador único no sistema distribuído e uma senha). Cada nova conta deverá ser iniciada com um **saldo inicial** de $1.000,00 e nunca permitir que seu **saldo atual** fique negativo. O serviço de "e-banking" deverá solicitar que o cliente se **autentique** no sistema antes de realizar qualquer operação em sua conta bancária, verificando as credenciais informadas pelo cliente (identificador e senha) com os dados armazenados no sistema distribuído. Uma vez autenticado, um cliente poderá realizar as seguintes operações bancárias: consulta de **saldo**, **transferência** de valor para outra conta (de um cliente diferente), consulta de **extrato** (histórico de operações da conta).

Para facilitar a interação com o sistema bancário, deverá ser cadastrada em cada conta bancária alguma informação pessoal do seu **cliente** (ex.: Nome e/ou CPF) porém, o serviço de “e-banking” não deverá aceitar o cadastro de duas contas contendo as mesmas informações pessoais (deve evitar duplicatas). E, para facilitar a operação de transferência, o serviço de “e-banking” deverá oferecer ao cliente alguma forma de **pesquisa** pela informação pessoal dos outros clientes para localizar o identificador da conta destino desejada.

Deverá ser garantida a consistência dos dados! Dinheiro virtual será “criado” apenas na abertura de uma nova conta e esse dinheiro não poderá ser “perdido” dentro do sistema bancário distribuído, cujo montante (todo o dinheiro do banco) sempre deverá será igual à quantidade de contas multiplicada pelo saldo inicial padrão. Assim, uma operação especial para consultar o **montante** do banco deverá ser disponibilizada para fins de auditoria.

A interface com o usuário (UI) poderá ser tão simples quanto seja possível, desde que o usuário consiga através dela autenticar-se como cliente do banco e executar as operações financeiras em sua conta. A UI poderá ser uma interface de linha de comando (CLI) ou interface gráfica com o usuário (GUI), da forma como o grupo decidir e conforme os prazos do trabalho.

REQUISITOS DO SISTEMA DISTRIBUÍDO:

O sistema distribuído deverá ter **distribuição** vertical (sugestão: divisão em camadas MVC) e horizontal (replicação de componentes em cada camada). O sistema deverá ser **tolerante a falhas**, permanecendo operacional mesmo que alguns de seus componentes (membros do *cluster*) sofram falha por colapso (processo finalizado via Sistema Operacional). Portanto, sistema distribuído também deverá ter **distribuição** horizontal com cada componente sendo implantado em mais de uma máquina (ou instância).

Quando um novo componente for adicionado ao sistema (ingresso de novo membro no *cluster*) ele deverá receber a **transferência do estado** atualizado do sistema. O estado do sistema bancário consiste em: clientes autenticados (atualmente "*online*"), contas cadastradas com seus respectivos históricos de movimentação e demais informações de estado que se fizerem necessárias, conforme a função de cada componente no sistema distribuído.

O sistema deverá **armazenar** periodicamente seu estado de maneira **persistente** (sugestão: serialização de objetos para arquivo "em disco" — na memória secundária). Quando todos os componentes forem **desligados** e então "reiniciar" o sistema distribuído, deverá ser **carregado** para a memória principal o último estado (o mais atual possível dentre as réplicas) que tenha sido armazenado em memória persistente.

O Sistema Distribuído (SD) deverá possuir as seguintes características do **middleware** (JGroups):

- Serviço de composição do cluster, com rápida descoberta de novos membros e detecção de falhas nos membros de um cluster JGroups;
- Transmissão de mensagens entre os membros do cluster JGroups, através de multicast confiável e com ordenação das mensagens.
- Podem ser utilizadas flags de mensagem (sincronização), opções de requisição (votação), bloqueios / travas ou contadores atômicos, de acordo com o tipo de serviço necessitado pela aplicação;
- A pilha de protocolos XML projetada deverá apresentar um bom desempenho, devendo-se portanto ler a documentação do *middleware* JGroups para conhecer melhor os protocolos por ele providos e evitar incluir protocolos desnecessários ao funcionamento do serviço (cada protocolo adotado deverá ter sua escolha justificada).

ETAPA 1) o SD deverá prover as seguintes **funcionalidades BÁSICAS**:

- Prover alguma interface com o usuário (CLI ou GUI) para acesso rápido às funcionalidades do serviço;
- Definição de um identificador único por conta no sistema, de maneira que um mesmo cliente seja corretamente identificado ao sair e retornar;
- Cadastro de nova conta no sistema bancário sem duplicidade, conforme acordo dos membros do *cluster* (todos ou maioria);
- Operação de transferência de valor entre contas bancárias (atente para a necessidade de acesso mutuamente exclusivo às duas contas envolvidas);
- Operação de consulta ao saldo da conta, conforme acordo dos membros do *cluster* (primeira resposta ou maioria);
- Operação de consulta ao extrato da conta, conforme acordo dos membros do *cluster* (primeira resposta ou maioria);
- O valor do montante de dinheiro do banco (somatório do saldo de todas as contas) deve ser visível e consistente em todos os membros do *cluster* JGroups, para fins de auditoria;

ETAPA 2) o SD deverá prover as seguintes **funcionalidades INTERMEDIÁRIAS**:

- O sistema distribuído deve utilizar mais de um canal (JChannel) ou preferencialmente subcanais (ForkChannel) de comunicação para devidamente implementar a distribuição vertical (ex.: *clusters* "modelo", "controle", "visão"), segmentando as mensagens trocadas conforme a função de cada componente;
- O sistema distribuído deve providenciar armazenamento em disco do estado do sistema (cadastros, movimentações, etc.) em memória secundária (persistente);
- No reingresso de um membro ao *cluster*, deverá ser obtido o estado do sistema, ou seja, atualizações que ele possa não ter recebido enquanto estava desconectado;
- O sistema deverá prover mecanismos de segurança, como criptografia das mensagens trocadas, autenticação dos usuários e autenticidade das solicitações.

EXTRA) OPCIONALMENTE, o SD poderá prover as seguintes funcionalidades AVANÇADAS (de implementação não-obrigatória):

- O serviço de leilão pode ser capaz de comprimir mensagens grandes para agilizar a sua transferência;
- O serviço de leilão pode implementar um limitador de taxa de dados (quantidade de dados enviada por unidade de tempo), evitando o congestionamento do serviço ou ataques de negação de serviço (DoS);
- O serviço de leilão pode ser capaz de realizar controle de fluxo, ajustando a taxa dos transmissores à taxa do receptor mais lento;
- O serviço de leilão poder ser capaz de fragmentar mensagens de aplicação muito grandes em mensagens menores, remontando-as no receptor, agilizando assim sua transferência e evitando fragmentação a nível da camada de rede.
- O serviço de leilão pode ser capaz de permitir que usuários que estejam em LANs diferentes possam se conectar ao mesmo *cluster*, lidando com peculiaridades da infraestrutura de rede;
- O serviço deverá ser capaz de continuar funcionando caso haja particionamento na rede, devendo consolidar os estados das partições quando se reunirem.