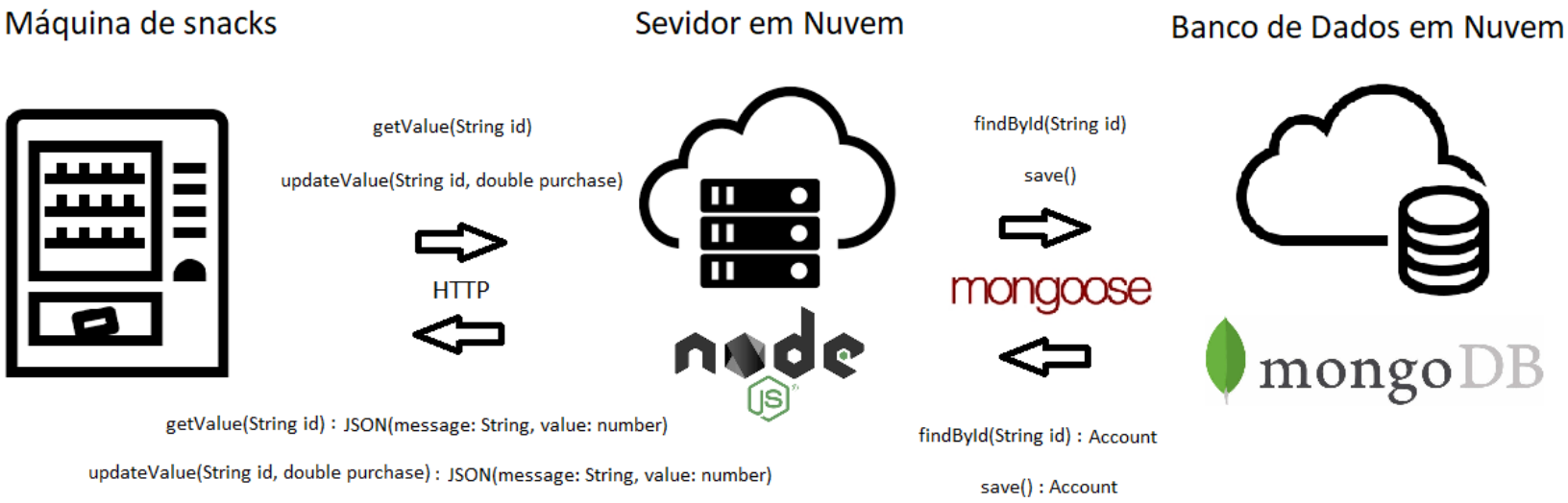
Teste Técnico - Proposta de Solução

André Sampaio Chacham
andre.chacham1@gmail.com
(31)99697-5242

De acordo com o problema definido pelo manuscrito do teste técnico, é necessário projetar um sistema em nuvem que fará a gerência dos saldos de cartões magnéticos distribuídos pela empresa Super Coffee MT LTDA. Para tanto, foi proposta a arquitetura desenhada abaixo e, em seguida, explicações a fundo do funcionamento de cada camada.



1º Camada - Máquina de Snacks:

Esta camada vai servir como o front-end do sistema, a interface para os clientes. Os cartões são inseridos nas máquinas para pegar lanches. A máquina realiza duas operações de requisição utilizando o protocolo HTTP e recebe suas respostas no formato JSON:

1) Leitura do cartão: Quando o cartão magnético é lido pela máquina, ela irá fazer uma requisição para o servidor em nuvem ‘getValue(String id)‘, onde id é o identificador de cada funcionário no banco de dados. Se for a primeira vez no dia que o cartão foi inserido a resposta será o valor inicial do dia definido pela empresa do usuário. Se não for a primeiro uso do dia, será retornado o saldo do usuário armazenado no banco de dados. O valor retornado vai aparecer no display e a próxima operação ficará disponível.

2) Compra de produto: Com o cartão inserido e o saldo à mostra, o cliente agora pode selecionar qualquer produto para fazer a compra. Para cada produto, será feita uma requisição ‘updateValue(String id, double purchase)’. Essa requisição só deve ser realizada se houver saldo disponível para a compra, caso contrário, receberá um erro do servidor. Esta operação pode ser realizada múltiplas vezes desde que houver saldo disponível para fazer uma compra.

2º Camada - Servidor em Nuvem:

Nesta camada é sugerida uma implementação de um servidor no ambiente de runtime Node.js, utilizando o framework Express.js. Este servidor irá ficar à espera das requisições HTTP das múltiplas máquinas de snacks da empresa e responderá de acordo com cada pedido. Para cada requisição que receber, será feita uma comunicação com o banco de dados em nuvem MongoDB, utilizando-se a biblioteca ODM(object data modeling) Mongoose, que provê uma interface simples e flexível com o SGBD em nuvem.

Como especificado pelo manuscrito do teste técnico, foi desenvolvido um trecho do código mais relevante para o funcionamento deste sistema. No caso da camada do servidor, este trecho se encontra nos arquivos ‘controllers/controller.js’ e ‘models/account.js’. O controller é responsável por receber as requisições HTTP das máquinas, aplicar as regras de negócio(como por exemplo, recarregar o saldo do cartão se for seu primeiro uso do dia) e comunicar com o banco de dados por meio do mongoose model ‘Account’ definido pelo schema em account.js.

O código de ambos arquivos se encontram na mesma partição do git que contém este documento.

3º Camada - Banco de Dados em Nuvem:

Como foi descrito anteriormente, esta camada utiliza o SGBD NoSQL MongoDB. Toda comunicação que é feita com o servidor é realizada por meio da biblioteca Mongoose. Esta biblioteca já disponibiliza todas as funções necessárias para o desenvolvimento deste sistema, como findById() e save(). Neste banco só será necessária a definição de um objeto, o Account, e como o MongoDB automaticamente já cria um identificador _id para todo objeto, só foi necessário a definição de mais três campos: ‘value’, ‘lastRechageDate’ e ‘initialValue’.