Universidade de São Paulo

Instituto de Ciências Matemáticas e Computação

Departamento de Ciências de Computação
Disciplina de
Programação Orientada a Objeto (SSC0103)

Documentação do Trabalho Final de

Programação Orientada a Objeto

Professor: Dr. Márcio Delamaro
Estagiários PAE: Lucas Lagoa Nogueira, Lucas Diniz Dallilo

São Carlos
27 de julho de 2021

1. Introdução

1.1 Sobre o Projeto

O projeto foi criado na disciplina de programação orientada a objetos e tem como seu intuito principal permitir que o usuário possa agendar eventos em grupo com a facilidade de ver quais os horários que estão vagos para suas reuniões.

Inspirado em projetos já consolidados no mercado como when2meet.com, TeamGroups, dentre outros, realizamos a nossa versão que possui o diferencial da conversa do usuário com um chatBot para marcar, organizar e mostrar de forma mais compreensiva e clara os horários preenchidos pelo grupo da atividade.

Usamos para a implementação desse projeto as tecnologias Java 11, Python 3.8.2+ e Telegram Group. Foi testado e compilado no sistema Linux Debian (buster 2018 LTS), Windows 10 (pro ultimate 2021).

1.2 Chat Bot

Para que o desenvolvimento da aplicação não derivasse em problemas de alta complexidade como reconhecimento de linguagem natural e expressões regulares, que sairiam do escopo da matéria e não haveria tempo hábil para depurar, testar e implementar, utilizamos pré-comandos já implementados pela API Telegram Group, que faz as tratativas de conectividade da nossa aplicação com a deles.

Então, utilizando o chatbot referido, o qual reconhece apenas comandos pré-determinados para criação de eventos, agendamentos e cancelamentos, desenvolvemos em Java uma aplicação que é integrada a outras tecnologias para simulação de uma agenda eletrônica compartilhada.

2.Objetivo

O objetivo do trabalho foi a implementação de uma agenda compartilhada, com diferentes recursos como: gerenciar horários, cadastrar reuniões, identificar e sinalizar congruências de agendamentos, evitando problemas de inconsistência de banco.

3. Descrição do Software

A ideia do software é criar uma forma fácil de desenvolver uma solução para agendamento de eventos. Por isso, desenvolvemos uma APIrestfull inspirada no site when2meet.com

3.1 Funcionamento do código JAVA

Inspirados pelo questionamento de qual seria um bom uso para um banco de dados não relacional e vendo a natureza do problema que escolhemos, optamos por usar uma base de dados fundamentada em MongoDB. A API funciona como as partes de Model e Controller do modelo MVC.

Para representar as rotas da API, foram utilizadas as seguintes notações: <Métodos de requisição HTTP> “rota”: descrição da funcionalidade.

- GET “/eventos/”: retorna todos os eventos cadastrados no banco de dados, é uma rota de testes que num possível produto final não seria aberta ao usuário
- POST “/evento”: cria uma entidade evento no banco de dados e recebe no body da requisição um JSON com a entidade
- GET “/evento/{id}”: retorna as informações do evento com o id informado no formato JSON
- PUT “/evento/{id}”: atualiza um evento com o id informado e recebe no body da requisição um evento no formato JSON
- DELETE “/evento/{id}”: deleta um evento no bd. Outra função de teste e que não seria aberta ao usuário
- POST “/evento/{id}/inserePessoa”: insere uma pessoa num evento recebe um objeto JSON

- PUT "/evento/{id}/updatehorarios/{nomePessoa}": atualiza uma pessoa em um evento

3.1.1 Classes Implementadas

- **Evento**

Cria-se ID do evento, nome do evento, dias da semana em que será realizado o evento e horários em que podem ser registrados integrantes.

- **Pessoa**

Insere o nome do participante e seus horários de disponibilidade.

- **DataDisponivel**

Dias da semana em que pode haver reuniões sobre o evento.

- **EventoRepository**

Classe de comunicação entre a API e o banco de dados para receber informações e diretivas do evento.

- **EventoController**

Classe de inserção, atualização, delete, reposição e mapeamento do banco, usando as funções

```
getAllEventos()
saveEvento(@RequestBody Evento evento)
getEventoById(@PathVariable String id)
deleteEventoById(@PathVariable String id)
updateEvento(@RequestBody Evento evento,@PathVariable String id)
addPessoaEvento(@RequestBody Pessoa pessoa,@PathVariable String id)
updatePessoaEvento(@RequestBody Pessoa pessoa,@PathVariable String
id,@PathVariable String nomePessoa)
```

- **DemoApplication**

Roda a classe Man através da Spring Application vinda de Springframework (org.springframework.boot.SpringApplication) para o funcionamento de nossa aplicação.

3.1.2 Classes Java relacionadas

SpringWeb

Classe responsável por lidar com a abstração da comunicação http.

Lombok

Classe que gera códigos “boiler plate” como getters e setters.

4. Telegram-group Bot

Utilizando a API Telegram, nosso modelo Java gera como saída um file .json tratado com os devidos agendamentos e matchs entre as pessoas cadastradas no evento.

Com as funcionalidades python 3.8+ e tecnologia da própria API do Telegram, utilizamos o reconhecimento de diretivas e geramos ao usuário um match entre os dias agendados em forma gráfica como a imagem representada abaixo:

	SEG	QUA	SEX
09:00:00	0	0	0
09:15:00	0	0	0
09:30:00	0	0	0
09:45:00	0	0	0
10:00:00	0	0	0
10:15:00	0	0	0
10:30:00	0	0	0
10:45:00	0	0	0
11:00:00	0	0	0
11:15:00	0	0	0
11:30:00	0	0	0
11:45:00	0	0	0
12:00:00	1	0	0
12:15:00	1	0	0
12:30:00	1	0	0
12:45:00	1	0	0
13:00:00	0	2	0
13:15:00	0	0	0
13:30:00	0	1	0
13:45:00	0	1	0
14:00:00	0	0	0
14:15:00	0	0	0
14:30:00	0	0	0
14:45:00	0	0	0
15:00:00	0	1	0
15:15:00	0	0	0
15:30:00	0	0	0
15:45:00	0	0	0
16:00:00	0	0	0
16:15:00	0	0	0
16:30:00	0	0	0
16:45:00	0	0	0
17:00:00	0	0	0
17:15:00	0	0	0
17:30:00	0	0	0
17:45:00	0	0	0
18:00:00	0	0	0

O usuário receberia uma imagem semelhante a esta em seu celular via aplicação Telegram, a qual daria visibilidade para determinados horários em que a reunião poderia ser realizada com participação do maior número de pessoas.

As cores mostram o grau de importância para determinados horários, quanto mais intensa a tonalidade, maior o número de pessoas disponíveis, sendo assim, mais recomendados para a realização do evento.

5.0 Considerações finais

No geral, o trabalho se desenvolveu como o esperado e acreditamos que conseguimos absorver todos os conceitos ensinados sobre programação orientada a objeto.

Encontramos maior dificuldade em algumas etapas: para implementar as tratativas do Telegram, principalmente, com relação ao efeito gráfico gerado ao usuário; para fazer as consistências de banco via aplicação Java e toda sua conectividade; e ainda, para criar eventos que se comunicassem entre os inputs recebidos pelo Telegram e conectados com nossa nossa API.

Foi desafiador o processo de implementação e arquitetura do projeto, mas conseguimos realizar o corpo e demonstrar os principais pontos de uma criação APIrestfull em Java, utilizando aplicações modernas e de uso comercial para empresas e escolas. Dessa maneira, foi alcançado nosso objetivo inicial de arquitetar uma estrutura sólida, robusta e inteligente a partir da mobilização de conhecimentos adquiridos na matéria integrados a outros conhecimentos que já vínhamos desenvolvendo em outras áreas da computação.

Todos os alunos envolvidos dedicaram-se em igual intensidade, com muito afinco e trabalho em equipe.

GRUPO 3

Vitor Henrique Turqueti dos Santos 10844313 [33%]
Solon Emanuel de Siqueira Marques 9791368 [33%]
Bruno Germano do Nascimento 1089313 [33%]