UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE COMPUTAÇÃO
CURSO DE ...

NOME COMPLETO DO AUTOR

TÍTULO DO TCC

TRABALHO DE CONCLUSÃO DE CURSO

CORNÉLIO PROCÓPIO
2018

NOME COMPLETO DO AUTOR

TÍTULO DO TCC

Trabalho de Conclusão de Curso apresentado ao Curso de ... da Universidade Tecnológica Federal do Paraná, como requisito parcial para a obtenção do título de ....

Orientador: Nome do orientador
Instituição do orientador

Coorientador: Nome do coorientador
Instituição do coorientador

CORNÉLIO PROCÓPIO
2018

Altere este texto inserindo a dedicatória do seu trabalho.

AGRADECIMENTOS

Edite e coloque aqui os agradecimentos às pessoas e/ou instituições que contribuíram para a realização do trabalho.

É obrigatório o agradecimento às instituições de fomento à pesquisa que financiaram total ou parcialmente o trabalho, inclusive no que diz respeito à concessão de bolsas.

Eu denomino meu campo de Gestão do Conhecimento, mas você não pode gerenciar conhecimento. Ninguém pode. O que pode fazer - o que a empresa pode fazer - é gerenciar o ambiente que otimize o conhecimento. (PRUSAK, Laurence, 1997).

RESUMO

SOBRENOME, Nome. Título do TCC. 2018. 29 f. Trabalho de Conclusão de Curso – Curso de ..., Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2018.

O Resumo é um elemento obrigatório em tese, dissertação, monografia e TCC, constituído de uma seqüência de frases concisas e objetivas, fornecendo uma visão rápida e clara do conteúdo do estudo. O texto deverá conter no máximo 500 palavras e ser antecedido pela referência do estudo. Também, não deve conter citações. O resumo deve ser redigido em parágrafo único, espaçamento simples e seguido das palavras representativas do conteúdo do estudo, isto é, palavras-chave, em número de três a cinco, separadas entre si por ponto e finalizadas também por ponto. Usar o verbo na terceira pessoa do singular, com linguagem impessoal, bem como fazer uso, preferencialmente, da voz ativa. Texto contendo um único parágrafo.

Palavras-chave: Palavra. Segunda Palavra. Outra palavra.

ABSTRACT

SOBRENOME, Nome. Title in English. 2018. 29 f. Trabalho de Conclusão de Curso – Curso de ..., Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2018.

Elemento obrigatório em tese, dissertação, monografia e TCC. É a versão do resumo em português para o idioma de divulgação internacional. Deve ser antecedido pela referência do estudo. Deve aparecer em folha distinta do resumo em língua portuguesa e seguido das palavras representativas do conteúdo do estudo, isto é, das palavras-chave. Sugere-se a elaboração do resumo (Abstract) e das palavras-chave (Keywords) em inglês; para resumos em outras línguas, que não o inglês, consultar o departamento / curso de origem.

Keywords: Word. Second Word. Another word.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
DECOM	Departamento de Computação

LISTA DE SÍMBOLOS

Γ	Letra grega Gama
λ	Comprimento de onda
$\in$	Pertence

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

Edite e coloque aqui o seu texto de introdução.

A Introdução é a parte inicial do texto, na qual devem constar o tema e a delimitação do assunto tratado, objetivos da pesquisa e outros elementos necessários para situar o tema do trabalho, tais como: justificativa, procedimentos metodológicos (classificação inicial), embasamento teórico (principais bases sintetizadas) e estrutura do trabalho, tratados de forma sucinta. Recursos utilizados e cronograma são incluídos quando necessário. Salienta-se que os procedimentos metodológicos e o embasamento teórico são tratados, posteriormente, em capítulos próprios e com a profundidade necessária ao trabalho de pesquisa.

1.1 LEIA ESTA SEÇÃO ANTES DE COMEÇAR

Este documento é um *template* LaTeX que foi concebido, primariamente, para ser utilizado na elaboração de Trabalho de Conclusão de Curs em conformidade com as normas da Universidade Tecnológica Federal do Paraná.

Para a produção deste *template* foi necessário adaptar o arquivo `abntex2.cls`. Assim, foi produzido o arquivo `utfpr-abntex2.cls` que define o `documentclass` específico para a UTFPR.

Antes de começar a escrever o seu trabalho acadêmico utilizando este *template*, é importante saber que há dois arquivos que você precisará editar para que a capa e a folha de rosto de seu trabalho sejam geradas automaticamente. São eles os arquivos `capa.tex` e `folha-rosto.tex`, ambos no diretório `/elementos-pre-textuais`. No arquivo `capa.tex` deverá ser informado nome do autor, título do trabalho, natureza do trabalho, nome do orientador e outras informações necessárias. No arquivo `folha-rosto.tex`, que contém o texto padrão estabelecendo que este documento é um requisito parcial para a obtenção do título pretendido, será necessário apenas comentar as linhas que não se aplicam ao tipo de trabalho acadêmico.

A compilação para gerar um arquivo no formato pdf, incluindo corretamente as referências bibliográficas, deve ser realizada utilizando o comando `makefile`, disponível na mesma pasta onde está o arquivo principal `utfpr-tcc.tex`. Caso seja alterado o nome do arquivo `utfpr-tcc.tex`, deverá ser alterado no arquivo `makefile` também.

1.2 ORGANIZAÇÃO DO TRABALHO

Normalmente ao final da introdução é apresentada, em um ou dois parágrafos curtos, a organização do restante do trabalho acadêmico. Deve-se dizer o quê será apresentado em cada um dos demais capítulos.

2 REVISÃO DE LITERATURA

É uma boa prática iniciar cada novo capítulo com um breve texto introdutório (tipicamente, dois ou três parágrafos) que deve deixar claro o quê será discutido no capítulo, bem como a organização do capítulo. Também servirá ao propósito de "amarrar" o conteúdo deste capítulo com o conteúdo do capítulo imediatamente anterior.

3 DESENVOLVIMENTO DA PESQUISA

3.1 DELINEAMENTO DA PESQUISA

Inserir seu texto aqui...

3.2 REENGENHARIA DO APLICATIVO MIP/MID

A primeira etapa para a reengenharia no aplicativo foi eu ter que desenvolver as funções lambda para as operações da aplicação. A linguagem que eu adotei para utilização do desenvolvimento das funções Lambda foi Java, pelo fato de ser a mesma linguagem da aplicação original e poder reaproveitar o código e não dificultar manutenção futura. Mas isso foi um grande desafio, pois, pelo fato da aplicação MIP/MID ter sido desenvolvida com o *framework Spring* a aplicação realiza várias configurações por de baixo do pano automaticamente, e quando eu tento replicar essas configurações manualmente não é tão simples quanto parece.

A primeira etapa de desenvolvimento foi encontrar um código de exemplo que estivesse integrando com o AWS Lambda e tenha sido desenvolvido com Java e Hibernate. Felizmente, eu encontrei um código aberto no Github (PATEL, 2018). O exemplo mostra uma implementação simples do Hibernate consumindo as credenciais de acesso via variáveis de ambiente, que será fornecida pela AWS Lambda Function, junto com as depedênicas maven necessárias para se integrar com o serviço AWS. Eu clonei esse projeto do repositório Github e comecei as alterações. A primeira alteração que eu achei útil foi adicionar a dependência *com.amazonaws:aws-lambda-java-events* que permite diferente tipos de entrada para os serviços que invocam a função Lambda, isso se mostrou útil pelo fato de que na função original do exemplo se utilizava de *BufferedReader* para realizar a leitura do corpo da requisição através de binários, o que torna toda leitura e interpretação mais complexa. Com a dependência *aws-lambda-java-events* foi possível capturar o JSON do objeto de uma maneira mais direta, utilizando a classe *APIGatewayProxyRequestEvent* como corpo referente ao request da chamada, e da mesma forma, a biblioteca também facilita gerar uma resposta JSON para o evento utilizando a classe *APIGatewayProxyResponseEvent*.

Com a aplicação de exemplo pronta para ser codificada, eu comecei a trazer as entidades de banco mapeadas com Hibernate para o código. As entidades cujo desenvolvimento eu iniciei foram os da camada Base. Copiei todas as classes e colei dentro de um pacote do projeto Lambda, e a primeira função que eu optei para desenvolver foi de encontrar todos os registros daquela entidade no banco de dados, que pode ser chamado de Get All ou Find All.

Figura 1 – Código da função GetFields

```
19 public class GetFieldRequest implements RequestHandler<APIGatewayProxyRequestEvent, APIGatewayProxyResponseEvent> {
20
21     @Override
22     public APIGatewayProxyResponseEvent handleRequest(APIGatewayProxyRequestEvent input, Context context) {
23         var sessionFactory = HibernateUtil.getSessionFactory();
24         var responseEvent = new APIGatewayProxyResponseEvent();
25
26         List<Field> resultList = null;
27         try (Session session = sessionFactory.openSession()) {
28             session.beginTransaction();
29
30             resultList = session.createQuery("from Field", Field.class).getResultList();
31             responseEvent.setHeaders(Collections.singletonMap("timeStamp", String.valueOf(System.currentTimeMillis())));
32             responseEvent.setStatusCode(200);
33
34             responseEvent.setBody(new ObjectMapper().writeValueAsString(resultList));
35
36             session.getTransaction().commit();
37
38         } catch (JsonProcessingException e) {
39             e.printStackTrace();
40         }
41         return responseEvent;
42     }
43
44
45 }
```

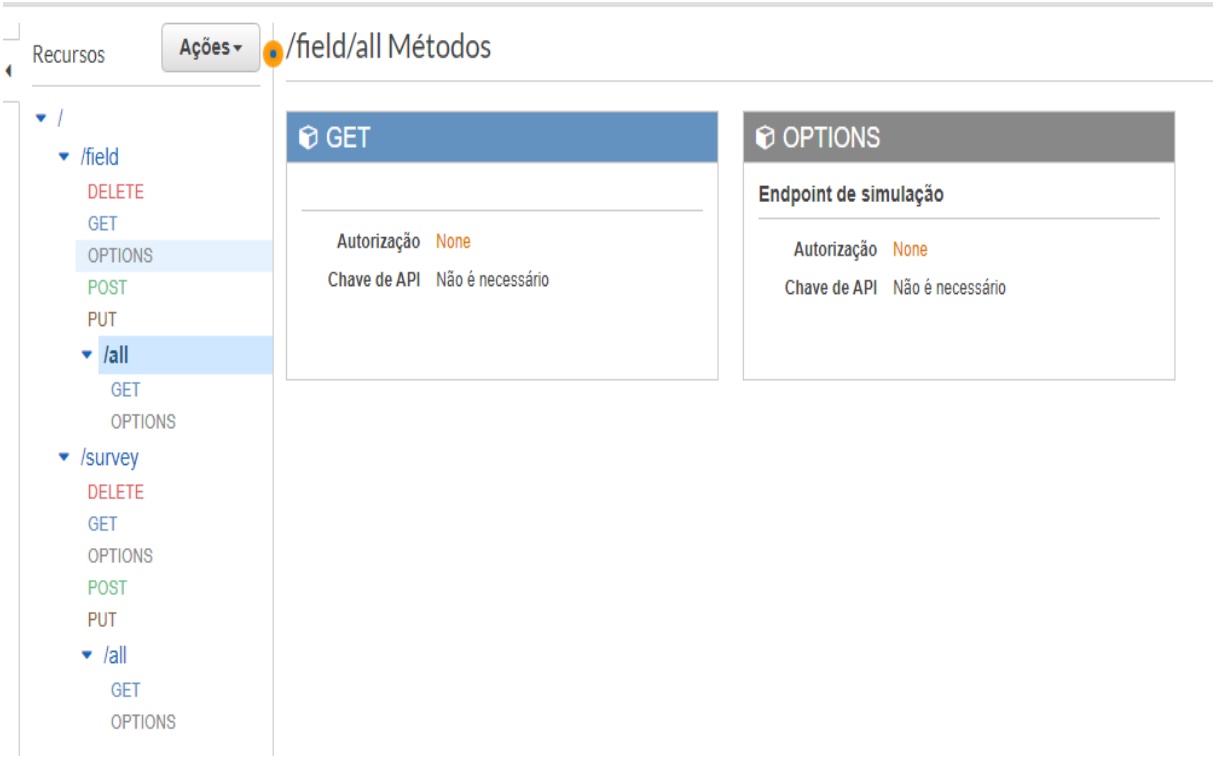
Fonte: Minha autoria

A implementação da função para busca de todos elementos da entidade Field pode ser visualizada na Figura 1. As primeira coisas a se notar no código são as classes de entrada e saída do *RequestHandler* na linha 19, que são as classes da biblioteca que foi mencionada anteriormente para facilitar a transição de dados da função na plataforma da AWS. A interface obriga a implementação do método *handleRequest*: esse método será acionado quando for parametrizado na função. Perceba pela sintaxe da linha 23 e 24 que está utilizando a tipagem *var*, com isso podemos inferir que código roda somente no JDK 10 para cima, no lambda permite-se o código Java 8 e 11 somente, que são as versões LTS atuais, e como o código original foi escrito em Java 11, as funções Lambda também serão escritas da mesma forma. Na linha 23 eu instancio uma sessão hibernate através da classe útil fornecida pelo exemplo e inicio a conexão com o banco na linha 27 com o recurso *try-catch-with-resources*, que fechará a conexão automaticamente após o fechamento do bloco. No começo do bloco dou inicio a uma transação e realizo uma chamada no banco de dados na linha para buscar todos os resultados possíveis que estão registrados na tabela Field. Desta forma, o resultado do banco é armazenado na variável resultList, e começo montando o objeto de resposta setando o cabeçalho e o status code da chamada. Esse response será disponibilizado para o API Gateway poder retornar essa mesma informação. Na linha 34 eu seto o corpo da resposta, serializando o objeto Java em um JSON String para que possa ser disponibilizado como recurso consumível no API Gateway e por fim na filha 36 eu commito a transação e retorno na linha 41.

Com o código da primeira função finalizada, eu acesso a plataforma da AWS e eu

vou primeiramente criar uma rota no serviço API Gateway para servir de gatilho para minha função. Após a API criada, o próximo passo é criar um recurso chamado /field e um sub-recurso chamado /all, dentro desses recursos eu cadastro os métodos HTTPs que serão utilizado dentro deles. Para a busca de todos os elementos e tentar respeitar os níveis de maturidade do modelo de Richardson para API REST, eu crio um metódo GET dentro do sub-recurso all e realizo o deploy da API para disponibilizar em um endpoint público.

Figura 2 – API Gateway



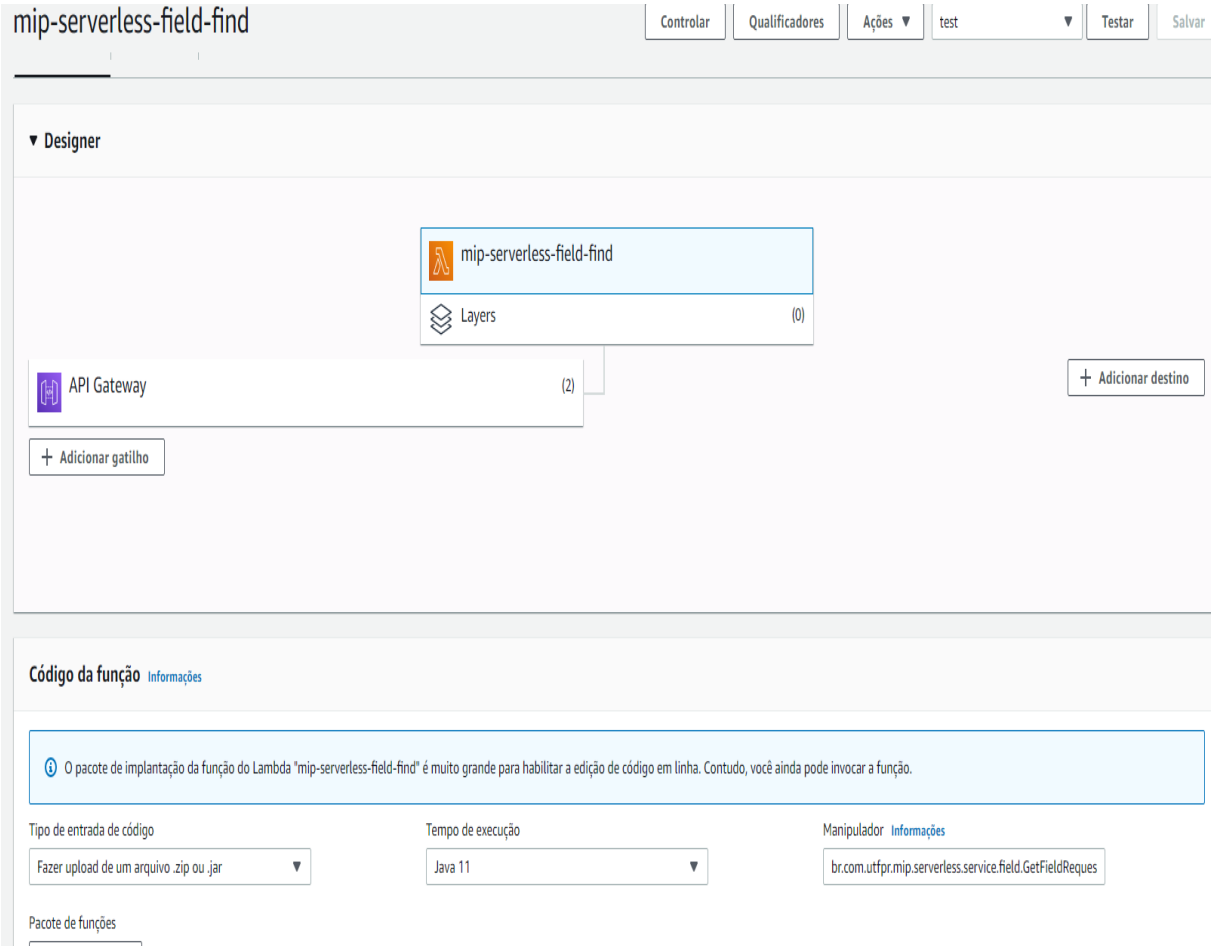
Fonte: Minha autoria

No próximo passo, eu crio uma função Lambda na plataforma da AWS e realizo o upload do JAR executável da aplicação, e aplico o caminho da função dentro do JAR para que a plataforma consiga encontrar o método *handleRequest* que foi implementado.

Na figura 3 eu mostro como fica na plataforma AWS a função já configurada. O serviço precisa ser ativado através de algum caminho, e é possível configurar esse gatilho no campo designer, conforme a figura, sendo que nesse campo eu consigo apontar qual recurso REST configurado no API Gateway, que eu já criei, eu quero que acione minha função lambda. Na figura 4 eu mostro como fica o campo de varáveis de ambiente após ter sido devidamente preenchida para que seja possível a aplicação se conectar com o banco.

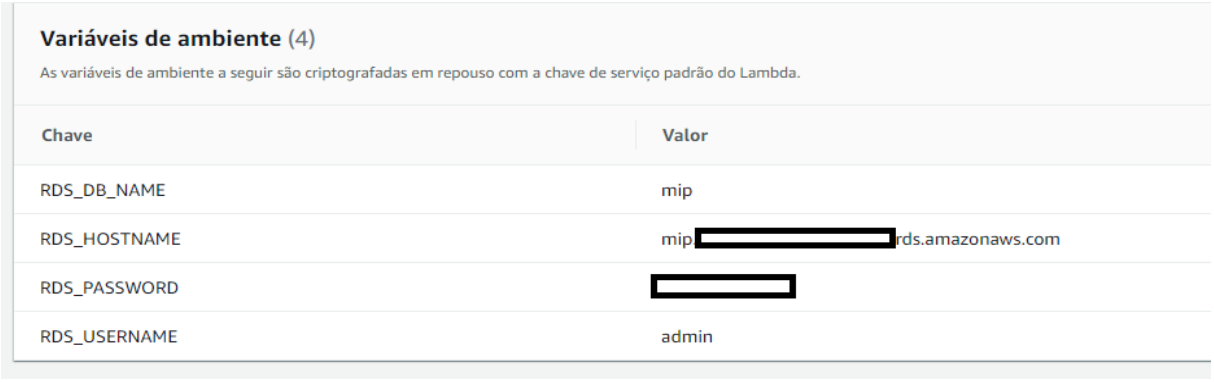
Com todos os parâmetros configurados, eu posso realizar um teste de chamada da função conforme eu mostro na figura 5, eu consigo montar um corpo em JSON para simular uma chamada REST, mas como essa função em especifico não possui parâmetros e retorna todos os elementos do banco basta clicar no botão teste para visualizar o resultado. E conforme

Figura 3 – Função Lambda para buscar todos Fields



Fonte: Minha autoria

Figura 4 – Variáveis de ambiente para se conectar no RDS

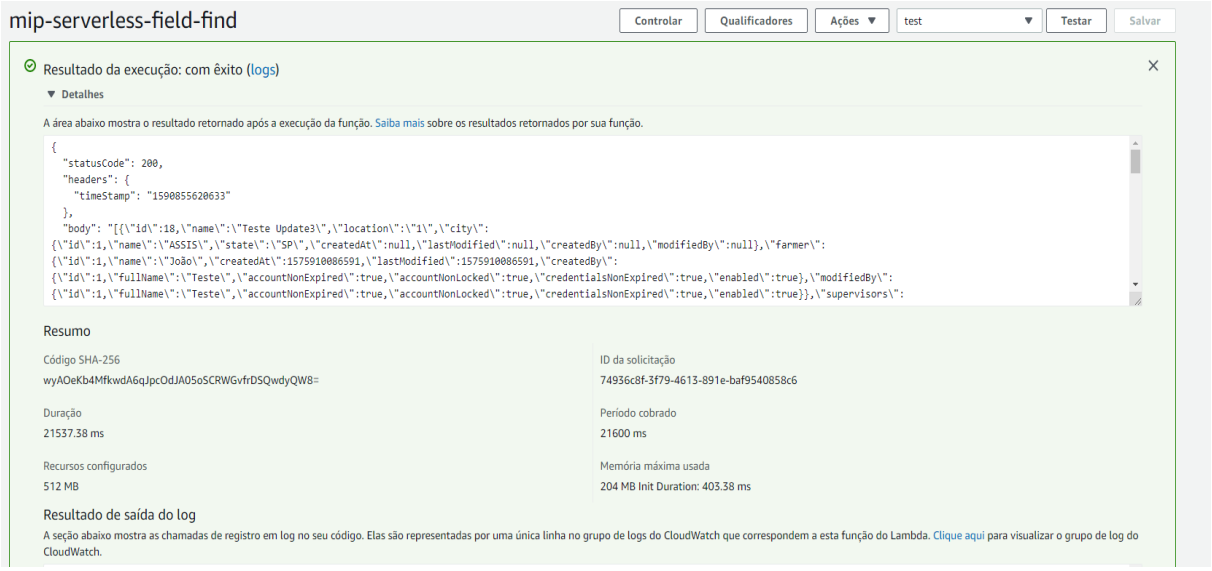


Fonte: Minha autoria

mostra na figura acima é possível visualizar o JSON mostrando informações das entidades cadastradas.

Com o IP público da API REST disponibilizado, eu vou até a aplicação original do MIP/MID e cadastro como propriedade a respectiva URL, no arquivo `application.properties`.

Figura 5 – Teste de chamada da função lambda



Fonte: Minha autoria

Após isso crio um novo pacote nomeado de Lambda no projeto e a classe `FieldLambda` para fazer o papel de wrapper de todas as chamadas Lambda referente a camada Field. Para facilitar as chamadas REST eu adicionei a dependência do Unirest, que é uma biblioteca em java para abstrair e facilitar requisições para APIs. Na imagem 6 eu mostro como fica a implementação para acionar a função lambda na AWS, na linha 29 e 30 eu busco a URL pública e armazeno na variável `ENDPOINT_GATEWAY` e crio o método readAll, utilizando a biblioteca Unirest para realizar a chamada GET na API.

Figura 6 – Serviço para acionar função Lambda

```
25 @Service
26 @RequiredArgsConstructor
27 public class FieldLambda {
28
29     @Value("${mip.gateway.url}")
30     private String ENDPOINT_GATEWAY;
31
32     public List<Field> readAll() {
33
34         var response = Unirest.get(ENDPOINT_GATEWAY+"/field/all")
35                 .asObject((new GenericType<List<Field>>() {})).getBody();
36
37         return response;
38     }
```

Fonte: Minha autoria

Com a estrutura criada para acionar a função Lambda, eu vou até o a camada de serviço e procuro pelo classe responsável por Field e pelo método readAll, que faz consulta direto no banco e simplesmente deleto o método, após isso vejo na aplicação inteira onde o código quebrou por conta da falta do método, e vou um a um substituindo pela nova chamada Lambda. Feito isso, a aplicação agora está se integrando com o Lambda da AWS chamando a função para buscar todos os Fields.

Para os outros cenários se manteve exatamente a mesma estratégia, diferindo somente nas regras de negócios mais especificas, tais como, validações de inserção e atualização de dados, permissão para deleção, mas tudo isso ficou contido dentro da função Lambda.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Cada capítulo deve conter uma pequena introdução (tipicamente, um ou dois parágrafos) que deve deixar claro o objetivo e o que será discutido no capítulo, bem como a organização do capítulo.

5 SOBRE AS ILUSTRAÇÕES

A seguir exemplifica-se como inserir ilustrações no corpo do trabalho. As ilustrações serão indexadas automaticamente em suas respectivas listas. A numeração sequencial de figuras, tabelas e equações também ocorre de modo automático.

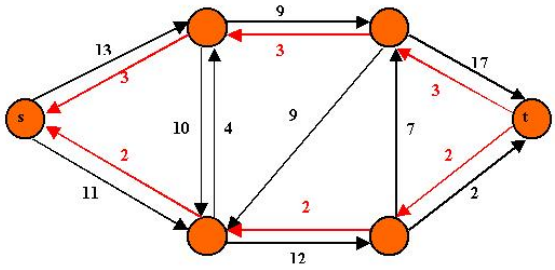
Referências cruzadas são obtidas através dos comandos `\label{}` e `\ref{}`. Sendo assim, não é necessário por exemplo, saber que o número de certo capítulo é 2 para colocar o seu número no texto. Outra forma que pode ser utilizada é esta: Capítulo 2, facilitando a inserção, remoção e manejo de elementos numerados no texto sem a necessidade de renumerar todos esses elementos.

6 FIGURAS

Exemplo de como inserir uma figura. A Figura 7 aparece automaticamente na lista de figuras. Para saber mais sobre o uso de imagens no LaTeX consulte literatura especializada (GOOSSENS et al., 2007).

Os arquivos das figuras devem ser armazenados no diretório de "/dados".

Figura 7 – Exemplo de Figura



Fonte: IRL (2014)

7 QUADROS E TABELAS

Exemplo de como inserir o Quadro 1 e a Tabela 1. Ambos aparecem automaticamente nas suas respectivas listas. Para saber mais informações sobre a construção de tabelas no LaTeX consulte literatura especializada (MITTELBACH et al., 2004).

Ambos os elementos (Quadros e Tabelas) devem ser criados em arquivos separados para facilitar manutenção e armazenados no diretório de "/dados".

Quadro 1 – Exemplo de Quadro.

BD Relacionais	**BD Orientados a Objetos**
Os dados são passivos, ou seja, certas operações limitadas podem ser automaticamente acionadas quando os dados são usados. Os dados são ativos, ou seja, as solicitações fazem com que os objetos executem seus métodos.	Os processos que usam dados mudam constantemente.

Fonte: Barbosa et al. (2004)

A diferença entre quadro e tabela está no fato que um quadro é formado por linhas horizontais e verticais. Deve ser utilizado quando o conteúdo é majoritariamente não-numérico. O número do quadro e o título vem acima do quadro, e a fonte, deve vir abaixo. E Uma tabela é formada apenas por linhas verticais. Deve ser utilizada quando o conteúdo é majoritariamente numérico. O número da tabela e o título vem acima da tabela, e a fonte, deve vir abaixo, tal como no quadro.

Tabela 1 – Resultado dos testes.

	Valores 1	Valores 2	Valores 3	Valores 4
Caso 1	0,86	0,77	0,81	163
Caso 2	0,19	0,74	0,25	180
Caso 3	1,00	1,00	1,00	170

Fonte: Barbosa et al. (2004)

8 EQUAÇÕES

Exemplo de como inserir a Equação (1) e a Eq. 2 no corpo do texto [1]. Observe que foram utilizadas duas formas distintas para referenciar as equações.

$$X(s) = \int_{t=-\infty}^{\infty} x(t)\, \mathrm{e}^{-st}\, dt \tag{1}$$

$$F(u,v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \exp\left[-j2\pi \left(\frac{um}{M} + \frac{vn}{N}\right)\right] \tag{2}$$

[1] Deve-se atentar ao fato de a formatação das equações ficar muito boa esteticamente.

9 ALGORITMOS

Exemplo de como inserir um algoritmo. Para inserção de algoritmos utiliza-se o pacote `algorithm2e` que já está devidamente configurado dentro do template.

Os algoritmos devem ser criados em arquivos separados para facilitar manutenção e armazenados no diretório de "/dados".

Algoritmo 1: Exemplo de Algoritmo

Input: o número n de vértices a remover, grafo original $G(V, E)$
Output: grafo reduzido $G'(V,E)$
$removidos \leftarrow 0$
while *removidos* $<$ *n* **do**
  $v \leftarrow$ Random$(1, ..., k) \in V$
  for $u \in adjacentes(v)$ **do**
    remove aresta (u, v)
    $removidos \leftarrow removidos + 1$
  end
  if *há componentes desconectados* **then**
    remove os componentes desconectados
  end
end

10 SOBRE AS LISTAS

Para construir listas de "*bullets*" ou listas enumeradas, inclusive listas aninhadas, é utilizado o pacote `paralist`.

Exemplo de duas listas não numeradas aninhadas, utilizando o comando `\itemize`. Observe a indentação, bem como a mudança automática do tipo de "*bullet*" nas listas aninhadas.

- item não numerado 1
- item não numerado 2
  - subitem não numerado 1
  - subitem não numerado 2
  - subitem não numerado 3
- item não numerado 3

Exemplo de duas listas numeradas aninhadas, utilizando o comando `\enumerate`. Observe a numeração progressiva e indentação das listas aninhadas.

1. item numerado 1
2. item numerado 2
   a) subitem numerado 1
   b) subitem numerado 2
   c) subitem numerado 3
3. item numerado 3

11 SOBRE AS CITAÇÕES E CHAMADAS DE REFERÊNCAS

Citações são trechos de texto ou informações obtidas de materiais consultadss quando da elaboração do trabalho. São utilizadas no texto com o propósito de esclarecer, completar e embasar as ideias do autor. Todas as publicações consultadas e utilizadas (por meio de citações) devem ser listadas, obrigatoriamente, nas referências bibliográficas, para preservar os direitos autorais. São classificadas em citações indiretas e diretas.

12 CITAÇÕES INDIRETAS

É a transcrição, com suas próprias palavras, das idéias de um autor, mantendo-se o sentido original. A citação indireta é a maneira que o pesquisador tem de ler, compreender e gerar conhecimento a partir do conhecimento de outros autores. Quanto à chamada da referência, ela pode ser feita de duas maneiras distintas, conforme o nome do(s) autor(es) façam parte do seu texto ou não. Exemplo de chamada fazendo parte do texto:

Enquanto Maturana e Varela (2003) defendem uma epistemologia baseada na biologia. Para os autores, é necessário rever ....

A chamada de referência foi feita com o comando `\citeonline{chave}`, que produzirá a formatação correta.

A segunda forma de fazer uma chamada de referência deve ser utilizada quando se quer evitar uma interrupção na sequência do texto, o que poderia, eventualmente, prejudicar a leitura. Assim, a citação é feita e imediatamente após a obra referenciada deve ser colocada entre parênteses. Porém, neste caso específico, o nome do autor deve vir em caixa alta, seguido do ano da publicação. Exemplo de chamada não fazendo parte do texto:

Há defensores da epistemologia baseada na biologia que argumentam em favor da necessidade de ... (MATURANA; VARELA, 2003).

Nesse caso a chamada de referência deve ser feita com o comando `\cite{chave}`, que produzirá a formatação correta.

13 CITAÇÕES DIRETAS

É a transcrição ou cópia de um parágrafo, de uma frase, de parte dela ou de uma expressão, usando exatamente as mesmas palavras adotadas pelo autor do trabalho consultado.

Quanto à chamada da referência, ela pode ser feita de qualquer das duas maneiras já mencionadas nas citações indiretas, conforme o nome do(s) autor(es) façam parte do texto ou não. Há duas maneiras distintas de se fazer uma citação direta, conforme o trecho citado seja longo ou curto.

Quando o trecho citado é longo (4 ou mais linhas) deve-se usar um parágrafo específico para a citação, na forma de um texto recuado (4 cm da margem esquerda), com tamanho de letra menor e espaçamento entrelinhas simples. Exemplo de citação longa:

> Desse modo, opera-se uma ruptura decisiva entre a reflexividade filosófica, isto é a possibilidade do sujeito de pensar e de refletir, e a objetividade científica. Encontramo-nos num ponto em que o conhecimento científico está sem consciência. Sem consciência moral, sem consciência reflexiva e também subjetiva. Cada vez mais o desenvolvimento extraordinário do conhecimento científico vai tornar menos praticável a própria possibilidade de reflexão do sujeito sobre a sua pesquisa (SILVA; SOUZA, 2000, p. 28).

Para fazer a citação longa deve-se utilizar os seguintes comandos:

```
\begin{citacao}
<texto da citacao>
\end{citacao}
```

No exemplo acima, para a chamada da referência o comando `\cite[p.~28]{Silva2000}` foi utilizado, visto que os nomes dos autores não são parte do trecho citado. É necessário também indicar o número da página da obra citada que contém o trecho citado.

Quando o trecho citado é curto (3 ou menos linhas) ele deve inserido diretamente no texto entre aspas. Exemplos de citação curta:

A epistemologia baseada na biologia parte do princípio de que "assumo que não posso fazer referência a entidades independentes de mim para construir meu explicar"(MATURANA; VARELA, 2003, p. 35).

A epistemologia baseada na biologia de Maturana e Varela (2003, p. 35) parte do princípio de que "assumo que não posso fazer referência a entidades independentes de mim para construir meu explicar".

14 DETALHES SOBRE AS CHAMADAS DE REFERÊNCIAS

Outros exemplos de comandos para as chamadas de referências e o resultado produzido por estes:

Maturana e Varela (2003) `\citeonline{Maturana2003}`
Barbosa et al. (2004) `\citeonline{Barbosa2004}`
(SILVA; SOUZA, 2000, p. 28) `\cite[p.~28]{Silva2000}`
Silva e Souza (2000, p. 33) `\citeonline[p.~33]{v}`
(MATURANA; VARELA, 2003, p. 35) `\cite[p.~35]{Maturana2003}`
Maturana e Varela (2003, p. 35) `\citeonline[p.~35]{Maturana2003}`
(BARBOSA et al., 2004; MATURANA; VARELA, 2003) `\cite{Barbosa2004,Maturana2003}`

15 SOBRE AS REFERÊNCIAS BIBLIOGRÁFICAS

A bibliografia é feita no padrão BibTEX. As referências são colocadas em um arquivo separado. Neste template as referências são armazenadas no arquivo "base-referencias.bib".

Existem diversas categorias documentos e materiais componentes da bibliografia. A classe abnTEX define as seguintes categorias (entradas):

```
@book
@inbook
@article
@phdthesis
@mastersthesis
@monography
@techreport
@manual
@proceedings
@inproceedings
@journalpart
@booklet
@patent
@unpublished
@misc
```

Cada categoria (entrada) é formatada pelo pacote abnTeX2 e Araujo (2014b) de uma forma específica. Algumas entradas foram introduzidas especificamente para atender à norma ABNT (2002), são elas: `@monography`, `@journalpart`,`@patent`. As demais entradas são padrão BibTEX. Para maiores detalhes, refira-se a abnTeX2 e Araujo (2014b), abnTeX2 e Araujo (2014a), Araujo e abnTeX2 (2014).

16 NOTAS DE RODAPÉ

As notas de rodapé pode ser classificadas em duas categorias: notas explicativas[1] e notas de referências. A notas de referências, como o próprio nome ja indica, são utilizadas para colocar referências e/ou chamadas de referências sob certas condições.

[1] é o tipo mais comum de notas que destacam, explicam e/ou complementam o que foi dito no corpo do texto, como esta nota de rodapé, por exemplo.

17 CONCLUSÃO

Parte final do texto, na qual se apresentam as conclusões do trabalho acadêmico. É importante fazer uma análise crítica do trabalho, destacando os principais resultados e as contribuições do trabalho para a área de pesquisa.

17.1 TRABALHOS FUTUROS

Também deve indicar, se possível e/ou conveniente, como o trabalho pode ser estendido ou aprimorado.

17.2 CONSIDERAÇÕES FINAIS

Encerramento do trabalho acadêmico.

Referências

ABNTEX2; ARAUJO, L. C. **A classe abntex2**: Documentos técnicos e científicos brasileiros compatíveis com as normas abnt. [S.l.], 2014. 46 p. Disponível em: <http://abntex2.googlecode.com/>. Acesso em: 12 de setembro de 2014. Citado na página 20.

ABNTEX2; ARAUJO, L. C. **O pacote abntex2cite**: Estilos bibliográficos compatíveis com a abnt nbr 6023. [S.l.], 2014. 91 p. Disponível em: <http://abntex2.googlecode.com/>. Acesso em: 12 de setembro de 2014. Citado na página 20.

ARAUJO, L. C.; ABNTEX2. **O pacote abntex2cite**: Tópicos específicos da abnt nbr 10520:2002 e o estilo bibliográfico alfabético (sistema autor-data). [S.l.], 2014. 23 p. Disponível em: <http://abntex2.googlecode.com/>. Acesso em: 12 de setembro de 2014. Citado na página 20.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR 6023**: Informação e documentação — referências — elaboração. Rio de Janeiro, 2002. 24 p. Citado na página 20.

BARBOSA, C. et al. **Testando a utilização de "et al."**. 2. ed. Cidade: Editora, 2004. Citado 2 vezes nas páginas 12 e 19.

GOOSSENS, M. et al. **The LaTeX graphics companion**. 2. ed. Boston: Addison-Wesley, 2007. Citado na página 11.

IRL. **Internet Research Laboratory**. 2014. Disponível em: <http://irl.cs.ucla.edu/topology>. Acesso em: 8 de março de 2014. Citado na página 11.

MATURANA, H. R.; VARELA, F. J. **A Árvore do Conhecimento**: as bases biológicas da compreensão humana. 3. ed. São Paulo: Editora Palas Athena, 2003. Citado 3 vezes nas páginas 17, 18 e 19.

MITTELBACH, F. et al. **The LaTeX companion**. 2. ed. Boston: Addison-Wesley, 2004. Citado na página 12.

PATEL, A. **AWS Lambda Java with MySQL Integration using Hibernate**. 2018. Disponível em: <https://github.com/abhiptl/aws-lambda-java-mysql>. Acesso em: 10 de fevereiro de 2020. Citado na página 3.

SILVA, J.; SOUZA, J. a. L. **A Inteligência da Complexidade**. São Paulo: Editora Petrópolis, 2000. Citado 2 vezes nas páginas 18 e 19.

Apêndices

APÊNDICE A – Nome do apêndice

Lembre-se que a diferença entre apêndice e anexo diz respeito à autoria do texto e/ou material ali colocado.

Caso o material ou texto suplementar ou complementar seja de sua autoria, então ele deverá ser colocado como um apêndice. Porém, caso a autoria seja de terceiros, então o material ou texto deverá ser colocado como anexo.

Caso seja conveniente, podem ser criados outros apêndices para o seu trabalho acadêmico. Basta recortar e colar este trecho neste mesmo documento. Lembre-se de alterar o "label" do apêndice.

Não é aconselhável colocar tudo que é complementar em um único apêndice. Organize os apêndices de modo que, em cada um deles, haja um único tipo de conteúdo. Isso facilita a leitura e compreensão para o leitor do trabalho.

APÊNDICE B – Nome do outro apêndice

conteúdo do novo apêndice

Anexos

ANEXO A – Nome do anexo

Lembre-se que a diferença entre apêndice e anexo diz respeito à autoria do texto e/ou material ali colocado.

Caso o material ou texto suplementar ou complementar seja de sua autoria, então ele deverá ser colocado como um apêndice. Porém, caso a autoria seja de terceiros, então o material ou texto deverá ser colocado como anexo.

Caso seja conveniente, podem ser criados outros anexos para o seu trabalho acadêmico. Basta recortar e colar este trecho neste mesmo documento. Lembre-se de alterar o "label" do anexo.

Organize seus anexos de modo a que, em cada um deles, haja um único tipo de conteúdo. Isso facilita a leitura e compreensão para o leitor do trabalho. É para ele que você escreve.

ANEXO B – Nome do outro anexo

conteúdo do outro anexo