|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | | | |  | |
|  | escola  superior  de tecnologia  e gestão  POLITÉCNICO  DO PORTO | | | | |  | |
|  | | | CTeSP |  |  | | |
| Curso Técnico Superior Profissional  DESENVOLVIMENTO PARA A WEB E DISPOSITIVOS MÓVEIS | | |  |
|  | | | Grupo 3  Maria Dias  Tiago Costa  2021/2022 | | | |  |
|
|
|  | | | | | | | |

# 1. Introdução

## 1.1 Contextualização do problema

Está a decorrer um concurso que tem como intuito incentivar a melhoria das condições envolventes aos principais edifícios de cada localidade portuguesa. Desta forma, para cada edifício candidato podem ser apresentadas propostas de melhoria do espaço envolvente. Como o número de candidaturas é muito elevado, definiu-se que, por cada localidade, existe um financiamento máximo de 5000€ disponível. Se o edifício candidato for um dos selecionados, será registado.

## 1.2 Objetivos

O vocabulário **XSD** e **XML** deverá contemplar: informação acerca do edifício candidato. Esta informação inclui **nome**, **tipologia**, **localidade**, **facilidade** (inclui a designação e a capacidade) e **propostas** (inclui a descrição e o custo).

A **API** deverá suportar: a **submissão** de **documentos** correspondestes a um registo, **validar** o **documento** **XML** **submetido** e **registar** o **registo**. Durante o processo é **apresentado** o **resultado** (sucesso, caso o registo tenha sido efetuado, ou erro) e um **ID** **autoincrementado**. Tendo em conta os edifícios registados, **retorna** o **financiamento** **disponível** para **determinada** **localidade**. Posteriormente, a equipa de informática atualiza periodicamente a Base de Dados indicando um **nível** de **satisfação** da proposta implementada.

Deveremos extrair os dados do BaseX. Em que se converte a Base de Dados XML em **JSON**.

No **MongoDB**, carrega-se os dados em JSON. Deverá associar a localidade de cada edifício às respetivas coordenadas geográficas. Deverá estruturar os dados e ainda preparar algumas consultas.

No **MongoCharts**, deverá desenvolver diversos gráficos, recorrendo á Base de Dados desenvolvida anteriormente.

## 1.3 Identificação dos pressupostos

Na parte do **XSD** e **XML**, considerámos que a **tipologia** deverá ser aceites apenas os seguintes valores: “monumento”, “loja”, “restaurante”, “hotel”, “outro”. As **propostas**, só poderão ser submetidas no mínimo 1 e no máximo 3, e indicadas por ordem preferencial uma vez que só uma poderá ser aceite.

Na parte da **API**, só poderemos registar um registo caso seja possível registar uma das propostas sem exceder o financiamento disponível para a localidade.

# Implementação

## 2.1 Descrição das etapas

Primeiramente, desenvolvemos o documento XSD e 20 ficheiros acerca dos edifícios em documento XML.

De seguida, desenvolvemos a API no BaseX.

Convertemos a Base de Dados resultante em JSON.

Após a conversão concluída, importou-se para o MongoDB e estruturou-se os dados (p.ex. converter string para inteiro). De seguida, associou-se a localidade dos edifícios às respetivas coordenadas. Por fim, realizou-se algumas consultas.

Por fim, no MongoAtlas disponibilizamos a nossa Base de Dados desenvolvida. Criámos uma data source e desenvolvemos uma dashboard que apresenta os gráficos realizados recorrendo ao MongoCharts.

## 2.2 Abordagem de modelação seguida para armazenar dados

Para validar a proposta do edifício, desenvolvemos a path “/addProposta”.

A função “addProposta”, retorna o resultado e/ou submete a proposta na Base de Dados.

Na função “addProposta”: a “$fd” irá retornar o financiamento disponível de uma determinada localidade, a “$proposta” guarda a primeira proposta do edifício que seja menor ou igual ao financiamento disponível, a “$p” guarda todas as propostas do edifício, a “$custo” guarda os custos de todas as propostas, a “$count” adiciona mais uma unidade ao número de edifícios registados na Base de Dados, a “$newBody” guarda a estrutura de dados do edifício registado na Base de Dados e a “$some” retorna true se na “$newBody” a proposta não é vazia (ou seja, pode ser aceite) ou false se a proposta é vazia.

Uma imagem com texto

Descrição gerada automaticamente

Figura 1: path addProposta

Uma imagem com texto

Descrição gerada automaticamente

Figura 2: função financiamento

Para avaliar o nível de satisfação da proposta implementada no edifício, desenvolvemos a path “/avaliar/{$id}”.

A função “avaliar”, insere o nível de satisfação no respetivo edifício. Caso num certo edifício já exista uma avaliação, este será substituído pela atual avaliação.

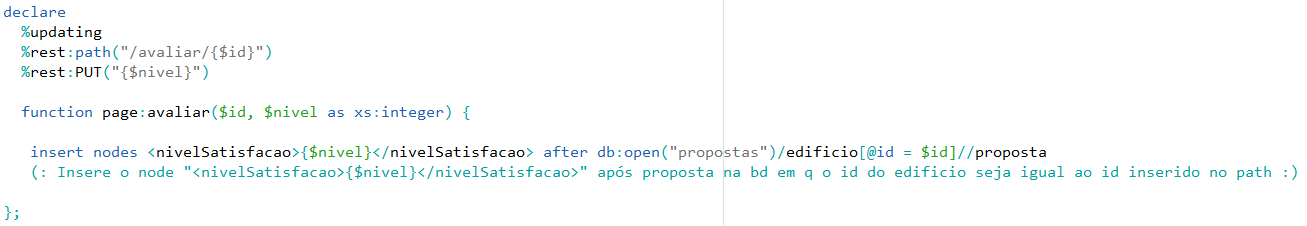


Figura 3: path "/avaliar"

## 2.3 Descrição do processo de transformação de dados

Após importar para o MongoDB o JSON e associar a localidade dos edifícios às respetivas coordenadas, tivemos que estruturar os dados.

Uma imagem com texto

Descrição gerada automaticamente

Figura 4: estrutura - antes

Primeiro, converteu-se os campos “id” e “nivelSatisfação” de string para int na coleção “edifícios”.

Segundo, converteu-se o objeto “proposta” em array na coleção “edifícios”. Para posteriormente conseguirmos converter alguns campos.

Terceiro, converteu-se o campo “capacidade” de string para int dentro de um array na coleção “edifícios”.

Quarto, converteu-se o campo “custo” de string para int dentro de um array na coleção “edifícios”.

Quinto, converteu-se o array “proposta” em object na coleção “edifícios”. Uma vez que, como pressuposto, concluímos que um edifício registado sempre terá apenas uma proposta aceite.

Sexto, renomeou-se os campos “admin\_name” e “population” na coleção “coordenadas”. Uma vez que, queremos que todos os campos sejam em português.

Sétimo, converteu-se o campo “populacao” para “0” caso seja vazio na coleção “coordenadas”.

Oitavo, converteu-se o campo “populacao” para int na coleção “coordenadas”.

Por fim, embutimos a coleção “coordenadas” na coleção “edifícios”.

Uma imagem com texto

Descrição gerada automaticamente

Figura 5: estrutura - depois

## 2.4 Consultas

Após o processo de transformações de dados, realizámos as consultas:

* “Apresentar total edifícios registados, por tipologia”

Uma imagem com texto

Descrição gerada automaticamente

Figura 6:Consulta: Apresentar total edifícios registados, por tipologia

* “Apresentar, por distrito e por tipologia, a média do custo das propostas aceites”

Uma imagem com texto

Descrição gerada automaticamente

Figura 7: Consulta: Apresentar, por distrito, a média do custo das propostas aceites

Uma imagem com texto

Descrição gerada automaticamente

Figura 8: Consulta: Apresentar, por tipologia, a média do custo das propostas aceites

* “Apresentar a percentagem das propostas aceites que obtiveram um nível de satisfação superior a 90”



Figura 9: Consulta: Apresentar a percentagem das propostas aceites que obtiveram um nível de satisfação superior a 90

* “Apresentar o total de edifícios por localidade”

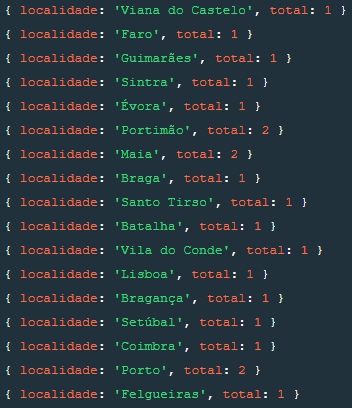


Figura 10: Consulta: Apresentar o total de edifícios por localidade

* “Apresentar o Financiamento Disponível por localidade”

Uma imagem com texto

Descrição gerada automaticamente

Figura 11: Consulta: Apresentar o Financiamento Disponível por localidade

* “Apresentar a percentagem dos edifícios por tipo de proposta”

Uma imagem com texto

Descrição gerada automaticamente

Figura 12: Consulta: Apresentar a percentagem dos edifícios por tipo de proposta

# Indicação do link para a dashboard

<https://charts.mongodb.com/charts-project-0-jfgxo/public/dashboards/040a53ad-6ad0-4112-b7d6-86196c3d9b1b>

# Conclusão

Com este trabalho, aprendemos mais funcionalidades do XQuery para fazer a API, realizar transformação de dados e consultas complexas no MongoDB, e criar gráficos no MongoCharts relacionados com o nosso tema.

Índice

[1. Introdução 2](#_Toc94432306)

[1.1 Contextualização do problema 2](#_Toc94432307)

[1.2 Objetivos 2](#_Toc94432308)

[1.3 Identificação dos pressupostos 3](#_Toc94432309)

[2. Implementação 3](#_Toc94432310)

[2.1 Descrição das etapas 3](#_Toc94432311)

[2.2 Abordagem de modelação seguida para armazenar dados 3](#_Toc94432312)

[2.3 Descrição do processo de transformação de dados 5](#_Toc94432313)

[2.4 Consultas 6](#_Toc94432314)

[3. Indicação do link para a dashboard 8](#_Toc94432315)

[4. Conclusão 8](#_Toc94432316)

[Índice de imagens 10](#_Toc94432317)

# Índice de imagens

[Figura 1: path addProposta 4](#_Toc94432318)

[Figura 2: função financiamento 4](#_Toc94432319)

[Figura 3: path "/avaliar" 5](#_Toc94432320)

[Figura 4: estrutura - antes 5](#_Toc94432321)

[Figura 5: estrutura - depois 6](#_Toc94432322)

[Figura 6:Consulta: Apresentar total edifícios registados, por tipologia 6](#_Toc94432323)

[Figura 7: Consulta: Apresentar, por distrito, a média do custo das propostas aceites 7](#_Toc94432324)

[Figura 8: Consulta: Apresentar, por tipologia, a média do custo das propostas aceites 7](#_Toc94432325)

[Figura 9: Consulta: Apresentar a percentagem das propostas aceites que obtiveram um nível de satisfação superior a 90 7](#_Toc94432326)

[Figura 10: Consulta: Apresentar o total de edifícios por localidade 7](#_Toc94432327)

[Figura 11: Consulta: Apresentar o Financiamento Disponível por localidade 8](#_Toc94432328)

[Figura 12: Consulta: Apresentar a percentagem dos edifícios por tipo de proposta 8](#_Toc94432329)