Staff Performance Evaluation Cycle 360

José Miguel Peixoto da Silva - a39964

Relatório de projeto apresentado à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Informática.

Trabalho orientado por:

Prof. Rui Pedro Lopes

Bragança

Novembro 2023

Staff Performance Evaluation Cycle 360

José Miguel Peixoto da Silva - a39964

Relatório de projeto apresentado à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Informática.

Trabalho orientado por:

Prof. Rui Pedro Lopes

Bragança

Novembro 2023

Resumo

A avaliação de funcionários é uma prática fundamental para o crescimento e aprimoramento contínuo de uma organização conjuntamente ao facto de a capacidade de avaliar de forma justa e abrangente o desempenho dos colaboradores ser crucial para garantir um ambiente de trabalho saudável e produtivo. Com o avanço da tecnologia, a criação de uma "Application Programming Interface" (API) dedicada à gestão de ciclos de avaliação poderá simplificar e aprimorar significativamente esse processo. A criação de uma RestAPI para a administração de ciclos de avaliação oferece uma solução eficaz e eficiente para os gestores de recursos humanos sendo que esta pode ser projetada de forma a garantir a facilidade de utilização e a segurança dos dados, permitindo aos responsáveis pela avaliação o acesso e gestão as informações dos funcionários de maneira integrada e centralizada. Assim, este projeto tem como objetivo desenvolver uma API capaz de criar Ciclos de Avaliação e responder aos mesmos que posteriormente irão resultar numa melhoria no ambiente de trabalho e na compreensão de problemas que possam existir e fazer a sua gestão de forma a solucionar o problema existente na GNG relativamente há avaliação dos seus funcionários.

Palavras-chave: RestApi e Ciclos de Avaliação

Abstract

Employee evaluation is a fundamental practice for the growth and continuous improvement of an organization, and the ability to assess employees' performance fairly and comprehensively is crucial to ensuring a healthy and productive work environment. With the advancement of technology, the creation of an API dedicated to managing evaluation cycles can significantly simplify and enhance this process.

Creating a RestAPI for the administration of evaluation cycles offers an effective and efficient solution for human resource managers, and it can be designed to ensure ease of use and data security, allowing evaluation managers to access and manage employee information in an integrated and centralized manner.

Therefore, the aim of this project is to develop an API capable of creating Evaluation Cycles and responding to them, which will subsequently lead to an improvement in the work environment and understanding of any existing issues, enabling their management to solve the existing problem at GNG regarding the evaluation of its employees.

Keywords: RestApi e Ciclos de Avaliação

Conteúdo

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Nos dias de hoje no mundo empresarial, a avaliação de funcionários tornou-se um processo essencial para o desenvolvimento empresarial, bem como manter a qualidade dos serviços prestados aos seus compradores. Assim sendo, uma empresa que deseje obter melhores resultados e que ambicione otimizar a sua performance em relação aos seus concorrentes necessita de ferramentas para analisar os resultados dos seus funcionários forma a detetar possíveis falhas. Existe uma necessidade de obter técnicas ou ferramentas que lhes permita ter um melhor controlo dos funcionários que contrata. Neste contexto, a construção de uma "Representational State Transfer Application Programming Interface" (RESTAPI) desempenha um papel fundamental, ao simplificar os processos de recolha, análise e interpretação de dados relacionados com os seus funcionários.

No âmbito deste cenário, a empresa alvo desta dissertação, a GNG uma empresa que consiste na gestão de lojas de roupa, busca otimizar o processo de avaliação de funcionários através da implementação de uma RESTAPI. A GNG, como organização em ascensão no setor de serviços, deparou-se com a necessidade de uma plataforma capaz de selecionar o problema existente, com a realização de ciclos de avaliação dos seus funcionários. Ao aproveitar a tecnologia avançada desenvolvida pela Omnium-Retail, a GNG visa otimizar a colheita, armazenamento e análise de dados de desempenho dos seus colaboradores, permitindo uma compreensão mais profunda do progresso individual e coletivo.

A abordagem personalizada que a Omnium-Retail utiliza nos seus sistemas vai ao

Figura 1.1: legenda

encontro diretamente com necessidade existente da GNG. Através desta colaboração, a GNG pode beneficiar da experiência da Omnium-Retail para desenvolver uma RESTAPI que se integre perfeitamente com as suas operações, atendendo aos requisitos específicos de avaliação de funcionários da GNG.

1.1 Enquadramento

O presente projeto foi desenvolvido para a Omnium-retail e consiste no desenvolvimento de uma RESTAPI que permita fazer uma avaliação 360, que consiste na criação de formulários ou seja, uma API que permita a criação de diferentes ciclos de avaliação, criação de questionários, responder aos ciclos de avaliação e por ultimo divulgar a análise das respostas obtidas. Este projeto foi iniciado em resposta a um desafio enfrentado pela GNG, que atualmente ainda depende de arquivos Excel para conduzir avaliações de desempenho

aos seus funcionários. A implementação dessa API personalizada representa uma solução fundamental para superar a dependência de métodos manuais e desatualizados e possibilitará à GNG a transição para um processo mais eficiente, digitalizado e abrangente de avaliação de desempenho.

1.2 Objetivos

O presente projeto visa o desenvolvimento de uma RESTAPI que seja capaz de criar ciclos de avaliação e responder aos mesmos. Pretende-se que a API desenvolvida responda a todos os requisitos propostos pela empresa GNG para além disso a API será desenvolvida da forma mais universal possível, para que no futuro a mesma possa ser utilizada para outras empresas.

1.3 Estrutura do Documento

Este documento está subdividido em capítulos, para que este relatório possa, de uma forma organizada, demonstrar todo o trabalho realizado.

- **Capítulo 1 - Introdução**
  Este capítulo faz uma abordagem superficial do tema e uma descrição do projeto.

- **Capítulo 2 - Contexto e Tecnologias/Ferramentas**
  Este capítulo faz uma abordagem ás tecnologias e ferramentas utilizadas no desenvolvimento da plataforma a ser desenvolvida.

- **Capítulo 3 - Estado de arte**
  Este capítulo faz uma abordagem ao estado de arte.

- **Capítulo 4 - Identificação de Requisitos, Modelação e Desenho**
  Este capítulo faz uma abordagem aos requisitos, Modelação e Desenho.

- **Capítulo 5 - Desenvolvimento do projeto e Testes de qualidade**

  Este capítulo faz uma abordagem aos desenvolvimentos necessários para a realização do projeto.

- **Capítulo 6 - Conclusão e trabalhos futuros**

  Este capítulo faz uma abordagem á conclusão do projeto e desenvolvimentos futuros.

Capítulo 2

Contexto e Problema

Neste capítulo será apresentado o problema existente na GNG que este projeto tem como objetivo responder, bem como as ferramentas e tecnologias necessárias para o desenvolvimento do mesmo, tais como Apache Airflow e Rabbit.

2.1 Gestão e Avaliação de Desempenho

O projeto que foi desenvolvido durante esta dissertação, tem como objetivo selecionar um problema existente na empresa GNG que consiste na dificuldade de avaliar os seus funcionários e de armazenar os seus dados. A GNG neste momento realiza os seus ciclos de avaliação via folhas excel, o que faz com que o trabalho necessite que diversas pessoas para realizar distribuição e recolha dos ciclos de avaliação.

Assim sendo a GNG fez a proposta para a realização deste projeto á Omnium-Retail, que consiste no desenvolvimento de uma RESTAPI que seja capaz de criar ciclos de avaliação, distribuí-los pelos funcionários, enviar notificações, realizar respostas, recolher as respostas e por último conseguir mostrar os resultados obtidos, aos funcionários.

Com esta RESTAPI a GNG deixará de necessitar de disponibilizar diversos recursos humanos na realização desta tarefa e terá um local em que será mais fácil realizar qualquer tipo de pesquisa ou visualização dos dados como um todo.

2.1.1 Omnium-Retail

A Omnium-Retail (Figura 2.1) é uma empresa de desenvolvimento de software com a Gestão de Retail Omni-Channel como o seu núcleo central. Possui bastante experiência no setor do retalho para uma Plataforma SaaS Omni-Channel totalmente integrada, capaz de atualizar o seu negócio para a era orientada por dados.

O produto principal da Omnium-Retail é uma plataforma moderna de gestão de integração que permite não ser apenas uma solução típica "tamanho único serve para todos", mas sim adotar uma abordagem personalizada para as necessidades de integração do cliente, capacitando o seu negócio com uma solução de Gestão Omnichannel.

A Omnium-Retail torna fácil aproveitar os seus dados de acordo com as suas necessidades [1].

Figura 2.1: Omnium-retail

2.1.2 GNG

A GNG (Figura 2.2) é uma empresa de gestão de lojas de vestuário, oferecendo ao consumidor um alto nível de serviço e satisfação, baseado em ética, respeito e valores humanos. Atualmente, a GNG é um dos maiores clientes da Levi® Strauss na Europa e um investimento firme e futuro da Dockers® e Adidas®, impulsionado por um crescimento constante através de uma política de investimento específica, baseada na capacidade de trabalho dos seus profissionais e na busca por novos mercados. Os principais pilares da

empresa são: seleção correta de roupas e gestão rigorosa de *stocks*; otimização do atendimento ao cliente; monitorização de todos os aspetos relacionados aos negócios, "em tempo real", de vários indicadores de desempenho e condições gerais do negócio. A GNG orgulha se de ter pontos fortes chave em toda a empresa: relações interpessoais fantásticas, valores humanos, competição saudável e capacidade de trabalho dos seus funcionários [2].

Figura 2.2: GNG

2.2 Tecnologias e Ferramentas

Esta secção faz a exploração detalhada das diversas ferramentas e tecnologias necessárias para a criação deste projeto, estabelecendo uma sólida relação com os objetivos e metas previamente estabelecidos.

Será dada atenção especial à análise dos tipos de bases de dados mais adequados para sustentar as funcionalidades da RESTAPI, que correspondam aos requisitos funcionais propostos pela GNG.

Além disso, serão exploradas outras ferramentas complementares que viabilizarão a monitorização contínua e eficaz da API, como também irão permitir que a própria API se torne assíncrona. Ao considerar cuidadosamente cada uma destas ferramentas, será possível criar uma RESTAPI sólida, eficiente e altamente funcional. Essa API estará plenamente capacitada de atender ás demandas específicas da GNG.

2.2.1 SQL vs. NoSQL

As tecnologias Not Only SQL (NoSQL) já existem desde os anos 1960, com vários nomes, mas estão a beneficiar de uma crescente popularidade à medida que o panorama dos dados muda e os programadores precisam de se adaptar para processarem o grande volume e o vasto leque de dados gerados na *cloud*, em dispositivos móveis, redes sociais e macrodados. As bases de dados NoSQL conseguem processar volumes enormes de dados não estruturados e em rápida mutação de formas diferentes de uma base de dados relacional ("Standard Query Language" (SQL)) com linhas e tabelas. Desde *tweets* virais de celebridades a informações que salvam vidas em registos de saúde eletrónicos, estão a ser gerados novos dados e tipos de dados a um ritmo vertiginoso. As bases de dados NoSQL evoluíram para ajudar os programadores a criarem rapidamente sistemas de bases de dados para armazenar as novas informações e torná-las prontamente disponíveis para pesquisa, consolidação e análise [3].

2.2.2 Modelos mais utilizados

- Chave de valor.
- Documentos.
- Colunas.
- Graph.

As principais diferenças entre as bases de dados SQL e NoSQL são as seguintes: As bases de dados SQL são relacionais, enquanto as bases de dados NoSQL são não-relacionais. As bases de dados SQL usam *querys* e têm um esquema predefinido; por outro lado, as bases de dados NoSQL não têm um esquema fixo, o que permite a utilização de dados não estruturados. As bases de dados SQL escalam bem, no sentido vertical, enquanto que as bases de dados NoSQL têm boa escalabilidade em ambos os sentidos. As bases de dados SQL são baseadas em tabelas, enquanto as bases de dados NoSQL podem ser

construídas através de documentos, armazenamento de valores-chave, gráficos ou colunas amplas. As bases de dados SQL são mais indicadas para transações de várias linhas, enquanto as bases de dados NoSQL, são mais adequadas para dados não estruturados, como documentos ou "JavaScript Object Notation" (JSON).

2.2.3 Bases de dados NoSQL

As bases de dados NoSQL são um conjunto diversificado de sistemas de gestão de bases de dados que oferecem soluções alternativas aos modelos tradicionais. Alguns dos exemplos desses sistemas incluem: Table Api, Cassandra, Mongodb, Gremelim.

Esta solução de base de dados foi desenvolvida pela primeira vez pelo Facebook. O Cassandra (Figura 2.3) está entre os mais escaláveis, pois, pode lidar com *petabytes* de informações e milhares de solicitações simultâneas. É a melhor opção para casos de uso que requerem mais operações de gravação do que de leitura [4].

Figura 2.3: Cassandra

2.2.4 Vantagens

- Hybrid
- Fault Tolerant
- Focus on Quality
- Performant
- You’re In Control

- Security and Observability
- Distributed
- Scalable
- Elastic

MongoDB [5] é um software de base de dados orientado a documentos livres, de código aberto e multiplataforma (Figura 2.4). Classificado como um programa de base de dados NoSQL, o MongoDB usa documentos semelhantes a JSON. Esta base de dados possui uma linguagem de consulta rica e expressiva que permite filtrar e classificar por qualquer campo, não importa o quão familiarizado este possa estar num documento, também suporta agregações e outros casos de uso modernos, como pesquisa baseada em localização geográfica, pesquisa de gráfico e pesquisa de texto. As próprias consultas são JSON e portanto facilmente combináveis.

Figura 2.4: MongoDB

2.2.5 Integrated Development Environment

O Visual Studio é um ambiente de desenvolvimento integrado (IDE) criado pela Microsoft, projetado para facilitar a criação de aplicações para diversas plataformas. É uma das ferramentas mais populares e amplamente utilizada por programadores de software em todo o mundo. Através da sua interface amigável e poderosa, o Visual Studio oferece aos seus utilizadores uma grande variedade de linguagens de programação, incluindo C#, C++, Python, JavaScript e muitas outras [6].

- **Mecanismo de fazer debug ao código**

  Esta tecnologia permite que os programadores sigam o processo passo a passo, examinar variáveis e encontrar erros em tempo real. Isso ajuda a identificar problemas e a corrigi-los rapidamente, economizando tempo e melhorando a qualidade do código.

- **Suporte integrado para o sistema Git**

  Este suporte permite uma maior facilidade no trabalho em equipa, permitindo que os programadores colaborem e partilhem o código desenvolvido de forma mais eficiente.

- **Grande quantidade de livrarias**

  As Livrarias permitem que os programadores personalizem o ambiente de desenvolvimento de acordo com suas necessidades e adicionem recursos adicionais, como suporte a novas linguagens, ferramentas de produtividade e integração com serviços de nuvem.

- **Suporte para desenvolvimento em diversas plataformas**

  Com a capacidade de desenvolver aplicações multiplataforma, os programadores podem atingir uma ampla audiência e oferecer uma experiência consistente em diferentes dispositivos.

De forma geral, o Visual Studio é uma ferramenta poderosa e versátil para programadores de software. Com a sua ampla gama de recursos, suporte a várias linguagens e facilidade de utilização.

2.2.6 Apache Airflow

O Apache Airflow (Figura 2.5) é uma plataforma de código aberto desenvolvida para automatizar, agendar e monitorizar fluxos de trabalho complexos de processamento de dados. Uma das principais características do Airflow é a sua abordagem baseada em código, que permite que os programadores definam os seus fluxos de trabalho com a utilização da linguagem Python. Isso oferece flexibilidade e controlo total sobre a lógica e sequência de tarefas do fluxo de trabalho. Os fluxos de trabalho no *Airflow* são definidos

como "Directed Acyclic Graphs" (DAG), onde as tarefas são representadas como nós e as dependências entre elas são representadas como arestas. Além disso as DAGs permitem aos programadores agendar execuções e programar execuções cíclicas. O painel de controle do Airflow oferece uma interface gráfica para monitorizar e visualizar o status dos fluxos de trabalho. Os utilizadores podem verificar a execução das tarefas, visualizar o histórico de execução, identificar erros e monitorizar o desempenho geral das *pipelines* de dados. Além disso, o Airflow permite que os utilizadores criem fluxos de trabalho com dependências temporais, agendem execuções, visualizem falhas e reagendem tarefas com facilidade. Isso torna o processo de desenvolvimento e manutenção de pipelines de dados, mais eficiente e organizado **Airflow**.

Figura 2.5: Airflow

Algumas das vantagens na utilização do Airflow: *Open-Source Software*, Os dados são encriptados, interface gráfica de fácil compreensão, fácil monitorização, permite o agendamento de processos e disponibiliza funções capazes de trabalhar com fluxos de trabalho. Assim sendo, o Apache Airflow na realização deste projeto, terá como intuito a criação de preços que irão ser executados diariamente e irão satisfazer alguns dos requisitos funcionais propostos pela empresa GNG.

2.2.7 Postman

O Postman (Figura 2.6) é uma ferramenta utilizada para realizar testes nas API. É um cliente HTTP que permite aos programadores enviar pedidos para as API, testar suas respostas e visualizar os resultados de maneira fácil e interativa.

Com o Postman, os programadores podem criar e organizar coleções de pedidos, que

podem ser agrupados em diferentes pastas para uma melhor organização. Cada solicitação dentro da coleção pode ser configurada com parâmetros, cabeçalhos, autenticação e outros detalhes relevantes para a API em questão.

Figura 2.6: Postman

Além disso, o Postman oferece suporte a diferentes tipos de solicitação, como GET, POST, PUT, DELETE, etc., permitindo testar diferentes métodos de acesso à API. Os programadores podem enviar solicitações para *endpoints* específicos, enviar dados no corpo da solicitação e receber as respostas correspondentes dos servidores.

Além disso, o Postman também possui recursos de teste automatizado, permitindo que os programadores criem *scripts* de testes automatizados para as suas APIs. Isso é particularmente útil para testes de regressão e garantia da qualidade, pois os testes podem ser executados repetidamente para verificar se as alterações no código não afetam negativamente a API.

O Postman está disponível como um aplicativo *desktop* para Windows, macOS e Linux, bem como uma extensão do Chrome que permite que os programadores consigam aceder á ferramenta diretamente do navegador [7].

2.2.8 Rabbit

RabbitMQ (Figura 2.7) é um software de mensagens assíncronas, de código aberto, projetado para facilitar a comunicação e o envio de mensagens entre diferentes aplicativos ou serviços em um ambiente distribuído. RabbitMQ é baseado no protocolo "Advanced

Message Queuing Protocol" (AMQP) e é amplamente utilizado para a troca de mensagens entre sistemas em tempo real.

O conceito principal por trás do RabbitMQ é o de uma fila de mensagens, onde as mensagens são publicadas num produtor e, em seguida, entregues a um ou mais consumidores. Isso permite que os aplicativos comuniquem de forma assíncrona, o que significa que podem continuar as suas operações sem esperar por uma resposta imediata. Essa abordagem é especialmente útil para cenários em que a escalabilidade, a resiliência e o desacoplamento entre os componentes são necessários.

Algumas características e conceitos importantes do RabbitMQ incluem:

- **Filas:**

  As mensagens são armazenadas em filas antes de serem entregues aos consumidores. As filas garantem que as mensagens sejam processadas de maneira ordenada e controlam o fluxo de tráfego entre os produtores e consumidores.

- **Produtores:**

  São as aplicações ou serviços que enviam as mensagens para o RabbitMQ. Estes publicam as mensagens nas filas do RabbitMQ.

- **Consumidores:**

  São os aplicativos ou serviços que recebem e processam as mensagens. Estes "inscrevem-se"nas filas e consomem as mensagens conforme elas chegam.

- **Exchanges:**

  São responsáveis por distribuir as mensagens para as filas corretas com base em um conjunto de regras definidas. O RabbitMQ suporta diferentes tipos de trocas, como trocas diretas, tópicas e de fanout, que permitem a configuração flexível das rotas de mensagens.

- **Virtual Hosts:**

  O RabbitMQ suporta a criação de vários *virtual hosts*, que são espaços de trabalho

isolados para grupos específicos de aplicativos ou serviços. Isso ajuda a organizar e separar as filas e trocas entre diferentes projetos ou equipes.

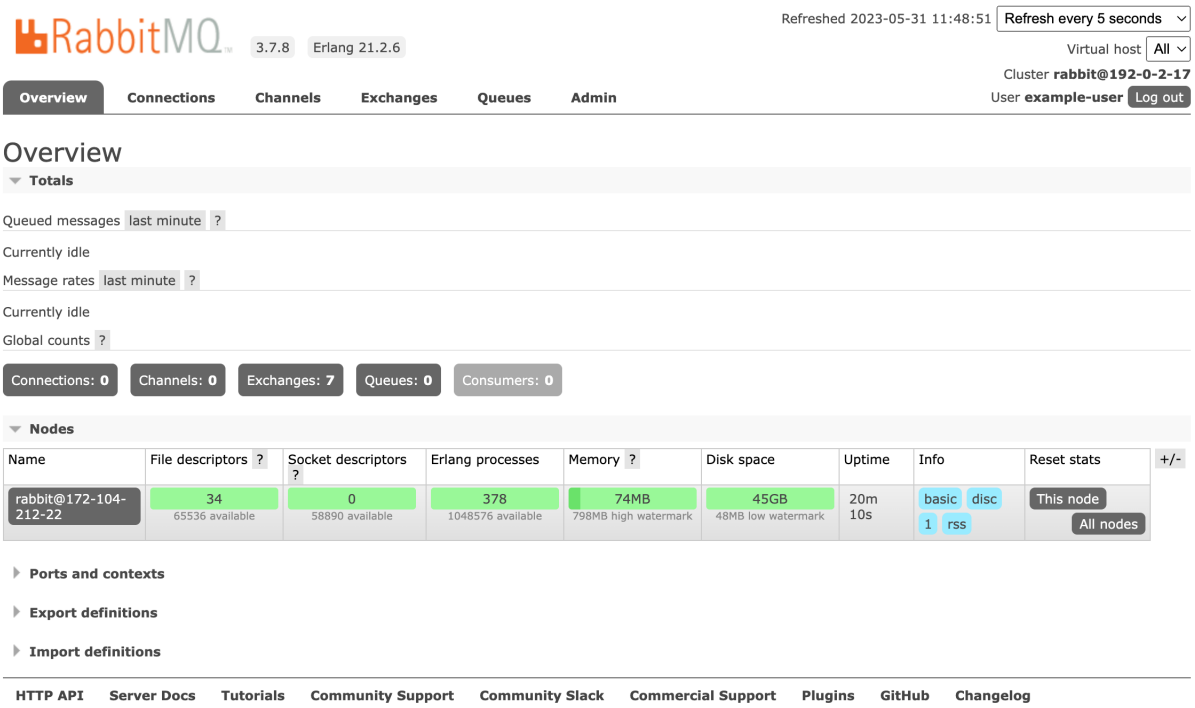

Figura 2.7: RabbitMQ Dashboard

O RabbitMQ é amplamente utilizado em ambientes de desenvolvimento e produção para implementar soluções de mensagens assíncronas, gestão de tarefas em segundo plano, processamento de eventos e muitos outros casos de uso onde a comunicação assíncrona é fundamental para a arquitetura do sistema. A sua flexibilidade, confiabilidade e suporte a diferentes linguagens de programação, tornam-no uma escolha popular entre os programadores para implementar sistemas distribuídos e altamente escaláveis [8].

2.2.9 SendinBlue

O SendinBlue (Figura 2.8) é uma plataforma de marketing digital que oferece uma variedade de ferramentas e serviços para ajudar empresas e profissionais de marketing a alcançar os seus objetivos de comunicação, automação de marketing e relacionamento

com clientes. Alguns dos principais recursos e serviços oferecidos pelo SendinBlue são usados por empresas de diversos tamanhos e setores para aumentar o envolvimento do cliente, gerar *leads*, promover produtos e serviços e automatizar processos de marketing. O SendinBlue oferece planos de preços que variam de acordo com o número de contactos e a frequência de uso, tornando-o acessível para empresas de diferentes portes. No contexto do projeto em desenvolvimento o SendinBlue será utilizado para o envio das notificações aos utilizadores **fig:sending**.

Figura 2.8: SendingBlue

2.3 Sumário

Neste capítulo foram abordadas todas as tecnologias e ferramentas que serão necessárias para o desenvolvimento deste projeto, para além disso foi também abordado o problema que se tenciona resolver, mas antes de qualquer desenvolvimento foi necessário realizar uma pesquisa para inferir que outras RESTAPI que possam existir.

Capítulo 3

Estado de arte

Neste capítulo, serão discutidas as diferentes APIs já existentes para a criação de questionários para os seus funcionários, tais como a SurveyMonkey e a Google Forms API. A SurveyMonkey API oferece recursos abrangentes para criar e distribuir questionários online, permitindo a recolha e análise eficiente de dados, que é utilizada principalmente em ambientes empresariais, mais precisamente nas áreas de pesquisa de mercado, feedback dos clientes, entre outros. Por outro lado, a Google Forms API oferece uma plataforma intuitiva e integrada para a criação de formulários personalizados, facilitando a recolha de informações de forma organizada e acessível, muitas vezes utilizada nos meios educacionais como, por exemplo na criação de testes.

3.1 SurveyMonkey API

A API do SurveyMonkey (Figura 3.1) é uma interface de programação, uma plataforma de criação e distribuição de pesquisas e questionários online. A API permite que os programadores acessem e interajam com os recursos e dados da plataforma de pesquisa, para criar integrações personalizadas, automatizar processos e extrair informações relevantes [9].

Com a API do SurveyMonkey, os programadores podem realizar várias operações, tais como:

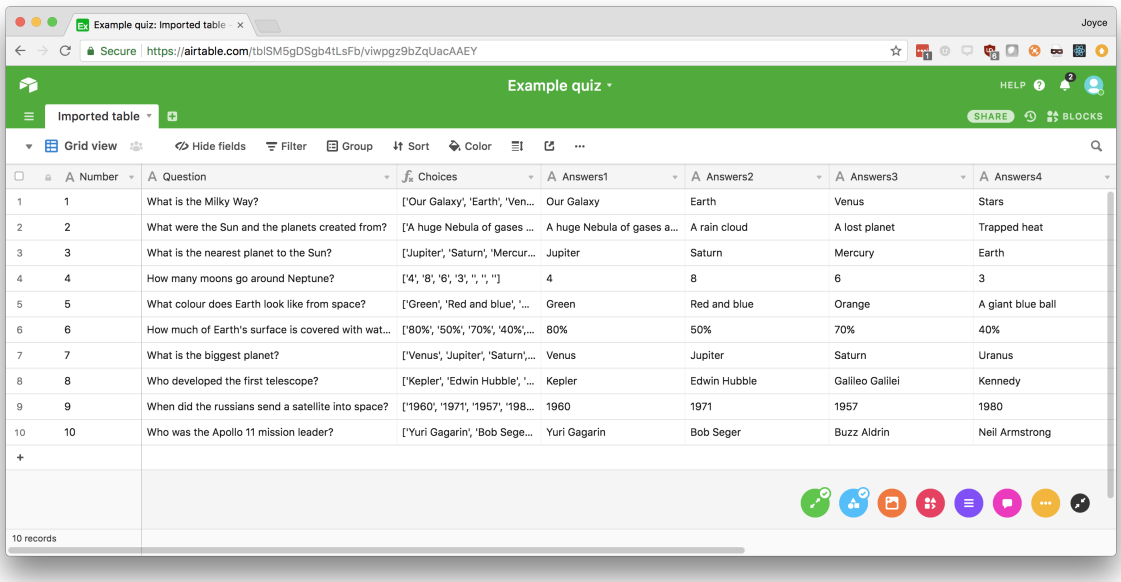

Figura 3.1: SurveyMonkey API

- Criar e gerir pesquisas: Os programadores podem criar novas pesquisas, configurar opções de perguntas, adicionar ou remover perguntas e ajustar as configurações da pesquisa.

- Recolha de respostas: É possível obter respostas dos participantes das pesquisas e analisar os dados coletados.

- Gestão de listas de destinatários: Os programadores podem importar, criar e gerir as listas de destinatários para distribuir pesquisas.

- Obter informações sobre pesquisas e relatórios: Os programadores podem acessar informações detalhadas sobre pesquisas, resultados e relatórios.

- Integração com outras ferramentas e sistemas: A API permite que os programadores integrem a plataforma SurveyMonkey com outras ferramentas e sistemas para automatizar os fluxos de trabalho e sincronizar dados.

3.2 Typeform API

A Typeform (Figura 3.3) API é uma plataforma popular para a criação de formulários e pesquisas online. A API permite que os programadores tenham acesso e interajam com os recursos e dados pela plataforma Typeform para criar integrações personalizadas, automatizar processos e extrair informações relevantes dos formulários [10].

Com a Typeform API, os programadores podem realizar diversas operações, tais como:

- Criar e gerir formulários: Os programadores podem criar novos formulários, definir perguntas e opções de resposta, personalizar a aparência e configurar as configurações do formulário.

- Partilha de formulários: Uma vez criado, o formulário pode ser partilhado com outras pessoas de forma fácil e rápida. Através do envio de um simples link diretamente, incorporar o formulário num site ou compartilhá-lo por e-mail.

- Recolher respostas: Os programadores podem aceder a informações detalhadas sobre os formulários criados e as respostas fornecidas pelos participantes.

- Integrar com outras ferramentas e sistemas: A API permite que os programadores integrem a plataforma Typeform com outras ferramentas e sistemas, para automatizar fluxos de trabalho e sincronizar dados.

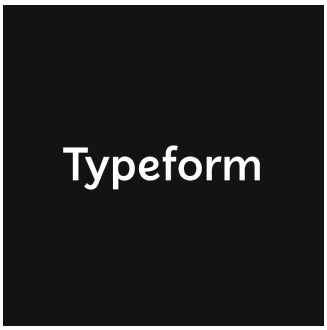

Figura 3.2: Typeform API

3.3 Google Forms API

O Google Forms é uma ferramenta para criação de formulários online desenvolvida pela Google. Esta permite que os seus utilizadores criem facilmente questionários, pesquisas e formulários personalizados para coletar informações, feedback ou respostas de outras pessoas pela internet.

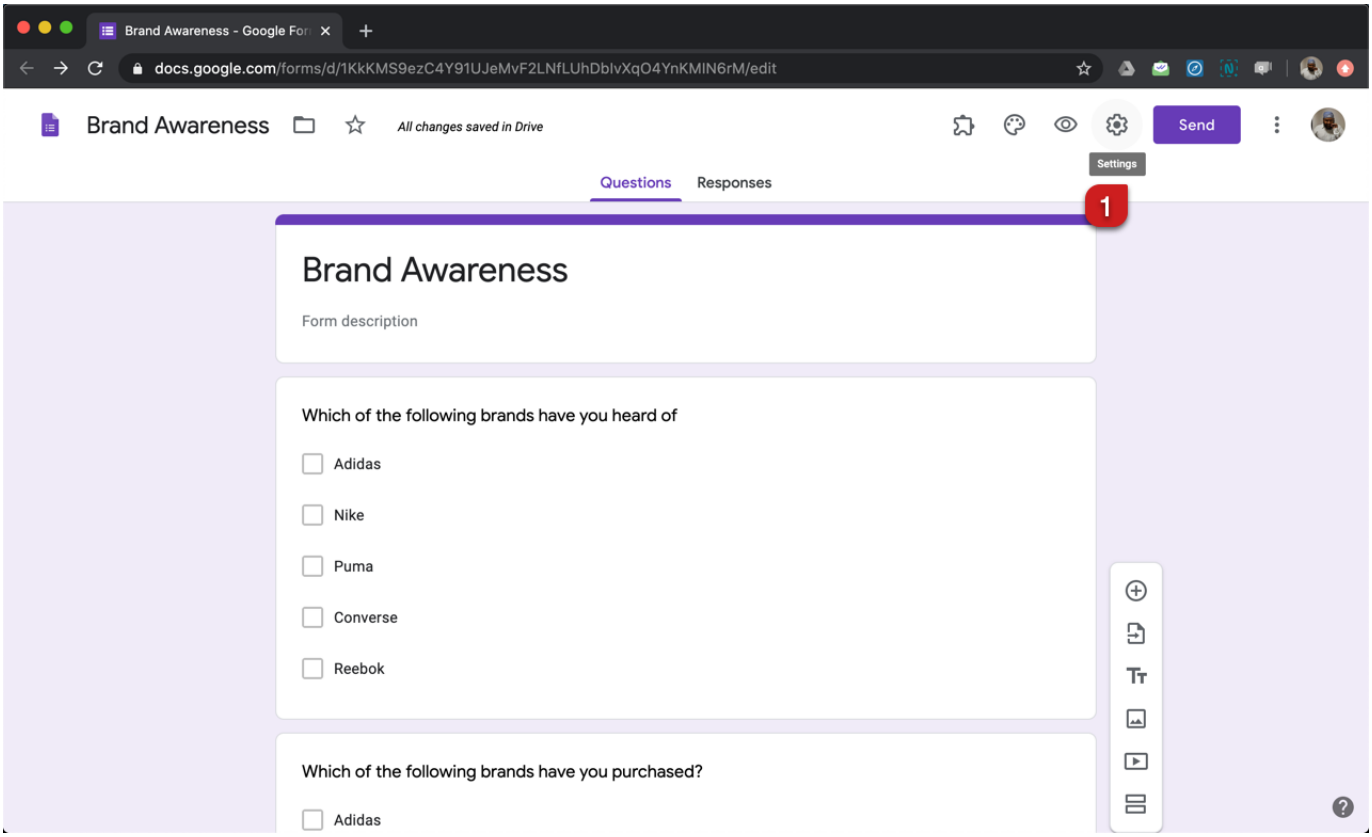

Figura 3.3: Google Forms

3.3.1 Características do Google Forms

- Criação de Formulários: O Google Forms, permite a criação de formulários personalizados que têm uma grande variedade de perguntas, incluindo caixas de texto, caixas de seleção, botões de opção, escalas de classificação e muito mais.
- Partilha de formulários: Uma vez criado, o formulário pode ser partilhado com

outras pessoas de maneira fácil e rápida com o envio de um simples link diretamente, incorporar o formulário num site ou compartilha-lo por e-mail.

- Recolha de respostas: À medida que o público alvo preenche o formulário, todas as respostas são automaticamente recolhidas e escritas no Google Sheets, facilitando a análise e a gestão de dados.

- Personalização: O formulário pode ter a sua aparência personalizada.

- Integração com outras ferramentas: O Google Forms pode ser integrado com outras ferramentas da Google, como Google Drive, Google Classroom e Google Analytics.

Assim sendo, o Google Forms é uma ferramenta versátil que pode ser usada em uma variedade de cenários, desde a criação de pesquisas simples até a realização de avaliações complexas ou formulários de inscrição. O Google Forms é acessível através de uma conta do Google e é uma opção popular e fácil de usar para coletar informações online de maneira eficaz [11].

3.4 Qualtrics API

A Qualtrics é uma API líder no setor de pesquisas online e experiência do cliente. A API permite que os programadores acessem e interajam com os recursos da plataforma Qualtrics, incluindo a criação e gestão de pesquisas, o envio de convites para pesquisas, a coleta de respostas e muito mais.

A API do Qualtrics é baseada em "Representational State Transfer" (REST) (Representational State Transfer) e usa solicitações "Hypertext Transfer Protocol" (HTTP) para realizar operações como criar pesquisas, enviar respostas, gerir utilizadores e obter dados. Os dados normalmente são retornados em formato JSON, tornando-os fáceis de serem processados e consumidos por aplicações e serviços.

Para utilizar a API do Qualtrics, os programadores precisam de obter uma chave de API e autenticar as suas solicitações. Também precisam de seguir a documentação oficial

fornecida pelo Qualtrics, de forma a entenderem, como realizar solicitações e interagir com os diversos recursos disponíveis.

Com a API do Qualtrics, é possível integrar pesquisas diretamente em aplicações e sistemas, automatizar processos de recolha e análise de dados, criar painéis personalizados de visualização e relatórios, além de incorporar pesquisas personalizadas em sites e aplicações [12].

3.5 Microsoft Forms

O Microsoft Forms é uma ferramenta da Microsoft que permite criar facilmente formulários e pesquisas online. Esta ferramenta faz parte do ecossistema Microsoft 365 (anteriormente conhecido como Office 365) e oferece uma maneira simples e intuitiva de coletar informações, opiniões e dados dos utilizadores de forma eficaz [13]. Aqui estão alguns detalhes sobre o Microsoft Forms:

- Criação de Formulários e Pesquisas: Com o Microsoft Forms, é possível criar formulários personalizados e pesquisas online em uma questão de minutos. Permite escolher entre diversos tipos de perguntas, incluindo escolha múltipla, respostas curtas, escalas de classificação e muito mais.

- Personalização: É possível personalizar a aparência dos formulários, escolhendo temas, cores e fontes que se alinhem à identidade visual da sua organização.

- Lógica de Ramificação: O Microsoft Forms permite a criação de lógica condicional nas perguntas, direcionando os utilizadores para diferentes perguntas com base nas respostas anteriores. Isso ajuda a tornar as pesquisas mais relevantes e direcionadas.

- Integração com Microsoft 365: Como parte do ecossistema Microsoft 365, o Microsoft Forms integra-se perfeitamente com outros aplicativos e serviços da Microsoft, como o SharePoint e o Excel.

- Análise de Dados: Depois da angariação de respostas, o Microsoft Forms oferece opções para análise de dados. É possível visualizar os resultados em gráficos e tabelas e até mesmo exportar os dados para o Excel para uma análise mais avançada.
- Segurança e Controle: O Microsoft Forms possui recursos de segurança para controlar quem pode aceder e responder aos formulários. É possível restringir o acesso a pessoas específicas ou grupos dentro da sua organização.
- Uso Educacional: Além do uso empresarial, o Microsoft Forms é frequentemente utilizado em ambientes educacionais para criar questionários, testes e avaliações online.

3.6 Sumário

Neste capítulo foram abordadas algumas ferramentas e tecnologias, bem como RESTAPI que possuem funcionalidades que se assemelham às necessidades e objetivos do projeto a desenvolver, assim obtendo uma perspectiva mais concreta dos requisitos que tornam uma solução de questionários empresariais viável e funcional. Assim sendo será necessário realizar uma modelação concreta do projeto e levantamento de requisitos funcionais para que seja possível passar para a fazer de implementação já com objetivos bem definidos.

Capítulo 4

Requisitos, Modelação e Desenho

Neste capítulo será abordada a metodologia para a realização do projeto, serão realizados os levantamentos dos requisitos funcionais consoante o FrontEnd partilhado pela GNG e por último a estrutura da base de dados.

4.1 Metodologia

Para o sucesso de todo e qualquer projeto é estritamente necessário definir uma metodologia bem estruturada de forma a encontrar uma solução para o problema proposto. No caso deste projeto iremos iniciar pelas reuniões com o cliente para que possam ser discutidos os desenvolvimentos que irão ser necessários. Após as reuniões o cliente partilhou um conjunto de Mocks de uma futura aplicação que será alimentada com a API que a desenvolver, o que possibilitou a realização de um levantamento de requisitos para o bom funcionamento do projeto. Foi também definido após reuniões com os clientes, que a melhor abordagem para a criação da base de dados seria através do modelo NoSql, o que deu origem à criação das coleções baseadas nos mockups disponibilizados pela GNG. Assim definidos os requisitos e a estrutura da base de dados será iniciado o desenvolvimento que se dividirá em três grandes partes, sendo elas: ciclos de avaliação, questionários e respostas.

4.2 Características do sistema

Após reuniões com o cliente foi possível recolher os requisitos funcionais para as páginas que os mesmo pretendiam desenvolver.

4.2.1 Página de criar ciclos de avaliação

Descrição sucinta

Os utilizadores que sejam do tipo Administrador poderão realizar criação de questionários onde poderão também selecionar os utilizadores que irão fazer parte da avaliação, poderão também escolher o tipo de ciclo que será criado e o questionário que será utilizado.

Requisitos funcionais associados

Os requisitos funcionais afetos à caracterıstica do sistema em epıgrafe encontram-se documentados na tabela 4.1.

ID	**Cabeçalho Descrição**	**Prioridade**
RF00	O sistema deve permitir ao administrador definir o intervalo de tempo em que o ciclo de avaliação deve estar ativo	Média
RF01	O sistema deve permitir ao administrador definir o tipo de ciclo de avaliação	Média
RF02	O sistema deve permitir ao administrador definir se o ciclo de avaliação é 360	Média
RF03	O sistema deve permitir ao administrador visualizar todos os utilizadores existentes para serem posteriormente selecionados para o ciclo de avaliação	Média
RF04	O sistema deve permitir ao administrador filtrar os utilizadores	Média
RF05	O sistema deve permitir ao administrador selecionar as datas para o envio das notificações	Média
RF06	O sistema deve permitir ao administrador ver os questionários existentes e selecionar 1	Média
RF07	O sistema deve permitir ao administrador dar início ao ciclo de avaliação	Média
RF08	O sistema deve permitir ao administrador guardar os ciclos de avaliação ciclo de avaliação	Média

Tabela 4.1: Ciclos de avaliação .

Caso de uso

Os casos de uso serão uma representação do que foi descrito anteriormente, como pode ser visto na Figura 4.1.

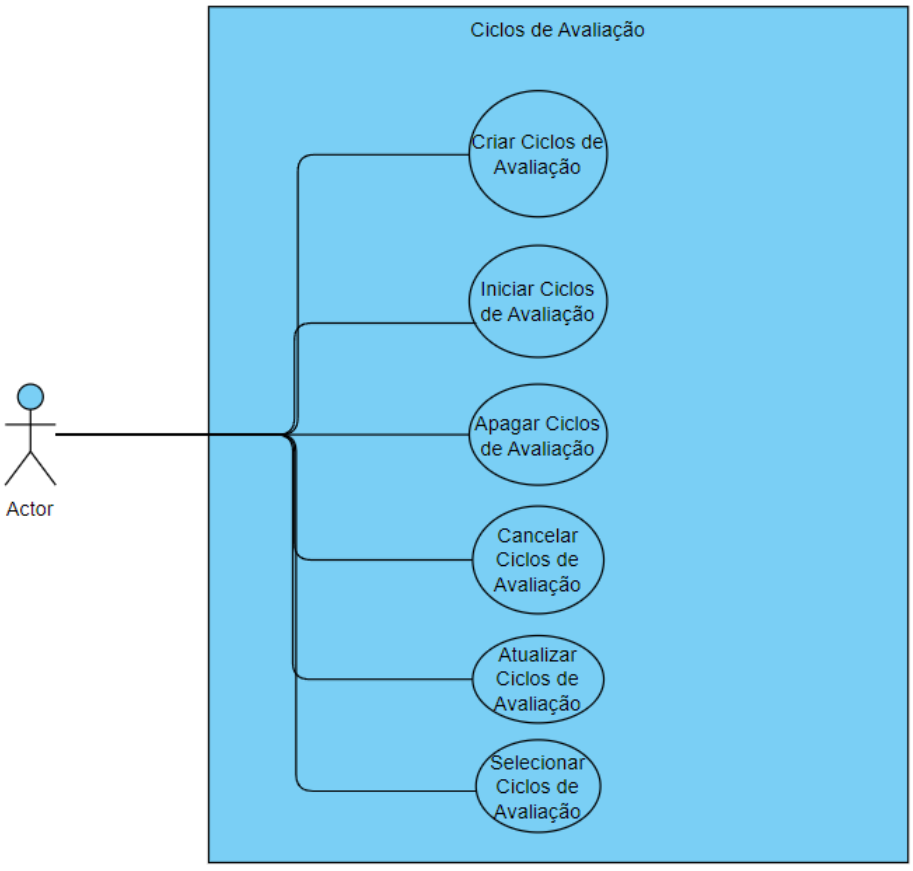

Figura 4.1: Criação de ciclos de Avaliação

4.2.2 Criação de Questionários

Descrição sucinta

Os utilizadores que tenham permissão para realizar a criação de funcionários conseguirão aceder á página para o fazer, nessa página poderão definir um nome, escolher o tipo de ciclos de avaliação alvo, sendo eles do tipo permanentes ou experimentais.

Em seguida criar-se-ão as categorias que irão ter uma ponderação e poderão definir o tipo de perguntas que nela irão existir.

Por último irão criar as perguntas que desejarem onde irão escrever a sua ponderação e uma descrição.

Requisitos funcionais associados

Os requisitos funcionais afetos à característica do sistema em epıgrafe encontram-se documentados na Tabela 4.2.

ID	**Cabeçalho Descrição**	**Prioridade**
RF00	O sistema deve permitir ao administrador definir o nome do questionário	Alta
RF01	O sistema deve permitir ao administrador definir o tipo de questionário	Média
RF02	O sistema deve permitir ao administrador definir um numero infinito de categorias a não ser que a soma da sua ponderação ultrapasse os 100	Média
RF03	O sistema deve permitir ao administrador definir o tipo de pergunta que a categoria irá englobar	Alta
RF04	O sistema deve permitir ao administrador criar subcategorias e ao mesmo tempo adicionar subsubcategorias	Baixa
RF05	O sistema deve permitir ao administrador guardar o questionário	Média
RF06	O sistema deve permitir ao administrador alterar o estado do questionário para depois poder ser utilizado na criação de ciclos de avaliação	Média

Tabela 4.2: Criação de Questionários.

Caso de uso

Os casos de uso serão uma representação do que foi descrito anteriormente (Figura 4.2).

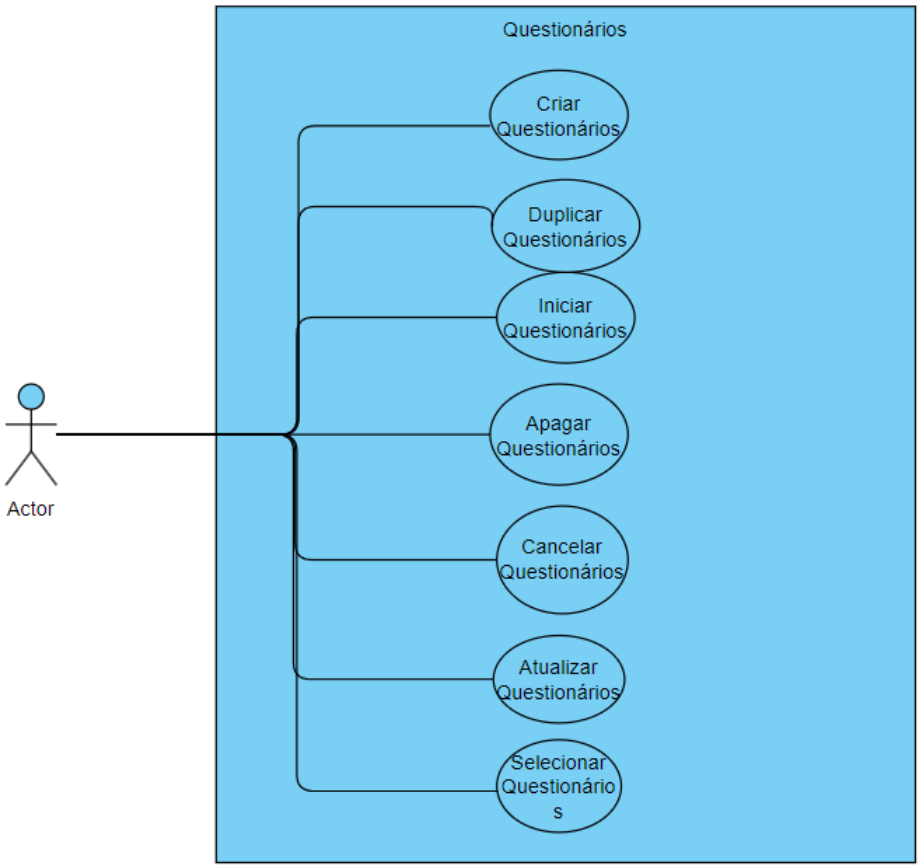

Figura 4.2: Criação de Questionários

4.2.3 Visualização de Ciclos de Avaliação

Descrição sucinta

Os utilizadores poderão ver a listagem dos ciclos de avaliação, para além da visualização, poderão fazer pesquisas e até realizar pesquisas mais avançadas através de querys e por último poderão alterar o estado do ciclo de avaliação para Finalizado, Cancelado, Apagado

e Editar.

Requisitos funcionais associados

Os requisitos funcionais afetos à caracterıstica do sistema em epıgrafe encontram-se documentados na Tabela 4.3.

ID	Cabeçalho Descrição	Prioridade
RF00	O sistema deve permitir ao administrador visualizar todos os ciclos de avaliação	Média
RF01	O sistema deve permitir ao administrador realizar pesquisas por termos ou por querys	Média
RF02	O sistema deve permitir ao administrador Finalizar um ciclo de avaliação	Média
RF03	O sistema deve permitir ao administrador Cancelar um ciclo de avaliação	Média
RF04	O sistema deve permitir ao administrador Apagar um ciclo de avaliação	Média
RF05	O sistema deve permitir ao administrador Editar um ciclo de avaliação	Média

Tabela 4.3: Visualização de Ciclos de Avaliação.

Caso de uso

Os casos de uso serão uma representação do que foi descrito anteriormente (Figura 4.3).

4.2.4 Visualização de Questionários

Descrição sucinta

Os utilizadores poderão ver a listagem de questionários para além da visualização, poderão fazer pesquisas e até realizar pesquisas mais avançadas através de querys. Por último poderão alterar o estado do questionário para Finalizado, Cancelado, Apagado e Editar.

Requisitos funcionais associados

Os requisitos funcionais afetos à caracterıstica do sistema em epıgrafe encontram-se documentados na Tabela 4.4.

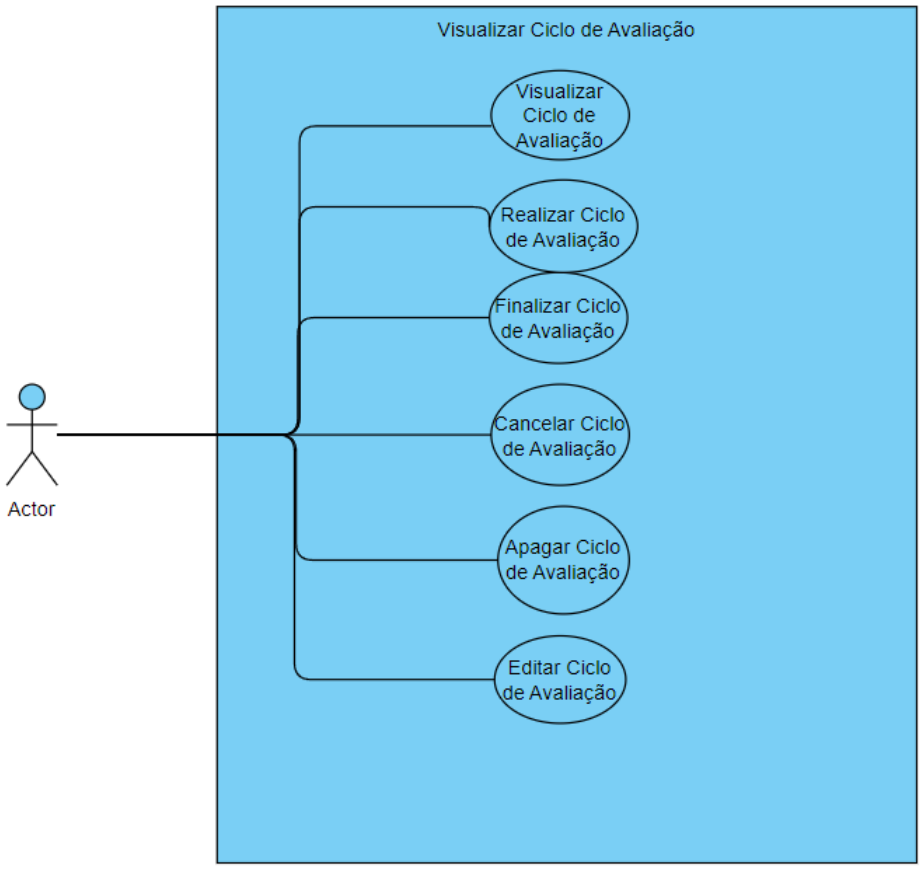

Figura 4.3: Visualização de Ciclos de Avaliação

ID	**Cabeçalho Descrição**	**Prioridade**
RF00	O sistema deve permitir ao administrador visualizar todos os questionários	Média
RF01	O sistema deve permitir ao administrador realizar pesquisas por termos ou por querys	Alta
RF02	O sistema deve permitir ao administrador Finalizar um questionário	Média
RF03	O sistema deve permitir ao administrador Cancelar um questionário	Média
RF04	O sistema deve permitir ao administrador Apagar um questionário	Média
RF05	O sistema deve permitir ao administrador Editar um questionário	Média

Tabela 4.4: Visualização de Questionários.

Caso de uso

Os casos de uso serão uma representação do que foi descrito anteriormente (Figura 4.4).

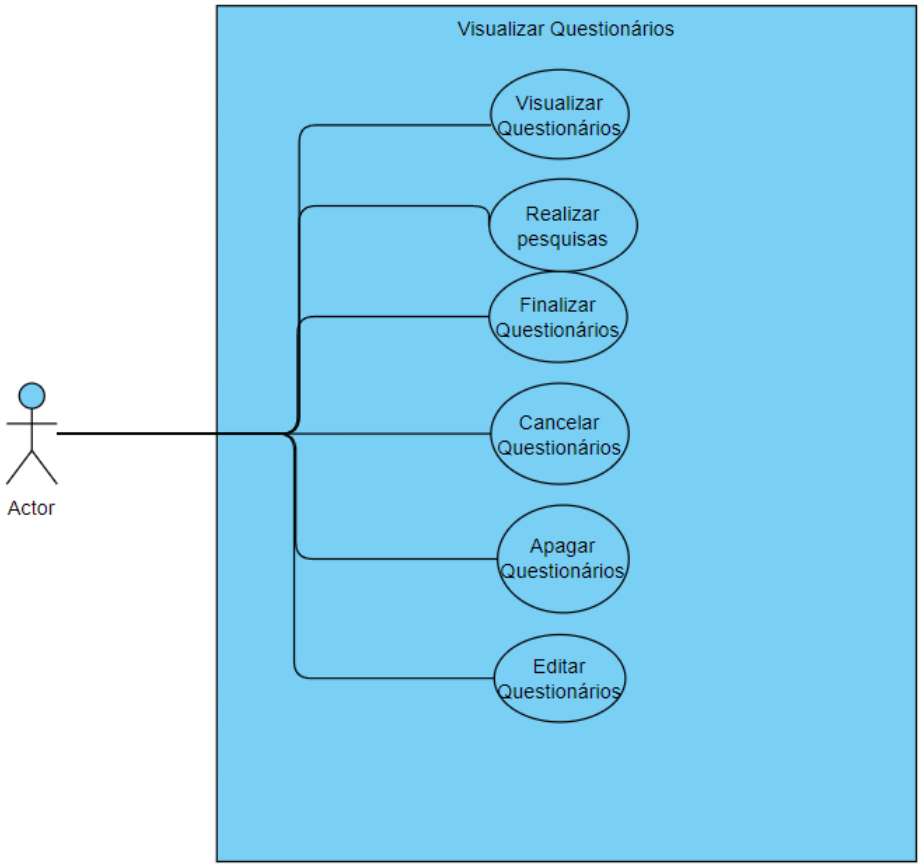

Figura 4.4: Visualização de Questionários

4.2.5 Analytics

Descrição sucinta

Os utilizadores poderão ver a listagem de todos os ciclos de avaliação que estão a decorrer no momento e o respetivo utilizador que está a responder ao mesmo. Para além da visualização poderão fazer pesquisas e até realizar pesquisas mais avançadas através de

querys e por último poderão entrar nas páginas dos utilizadores, para visualizar uma listagem precisa.

Requisitos funcionais associados

Os requisitos funcionais afetos à caracterıstica do sistema em epıgrafe encontram-se documentados na Tabela 4.5.

ID	Cabeçalho Descrição	Prioridade
RF00	O sistema deve permitir ao administrador visualizar todos os utilizadores que participam nos ciclos de avaliação	Baixa
RF01	O sistema deve permitir ao administrador visualizar os resultados do utilizador e do seu supervisor	Média
RF02	O sistema deve permitir ao administrador escolher o ciclo de avaliação para o qual quer visualizar os resultados dos utilizadores	Média
RF03	O sistema deve permitir o agrupamento dos ciclos de avaliação por ano ou por tipo	Média

Tabela 4.5: Analytics.

Caso de uso

Os casos de uso serão uma representação do que foi descrito anteriormente (Figura 4.5).

4.2.6 Histórico do Utilizador

Descrição sucinta

O utilizador terá acesso á sua página perfil onde poderá visualizar todos os ciclos de avaliação para o quais terá de responder, nessa listagem o utilizador também verá os ciclos de avaliação para os quais terá de responder como supervisor.

Requisitos funcionais associados

Os requisitos funcionais afetos à caracterıstica do sistema encontram-se documentados na Tabela 4.6.

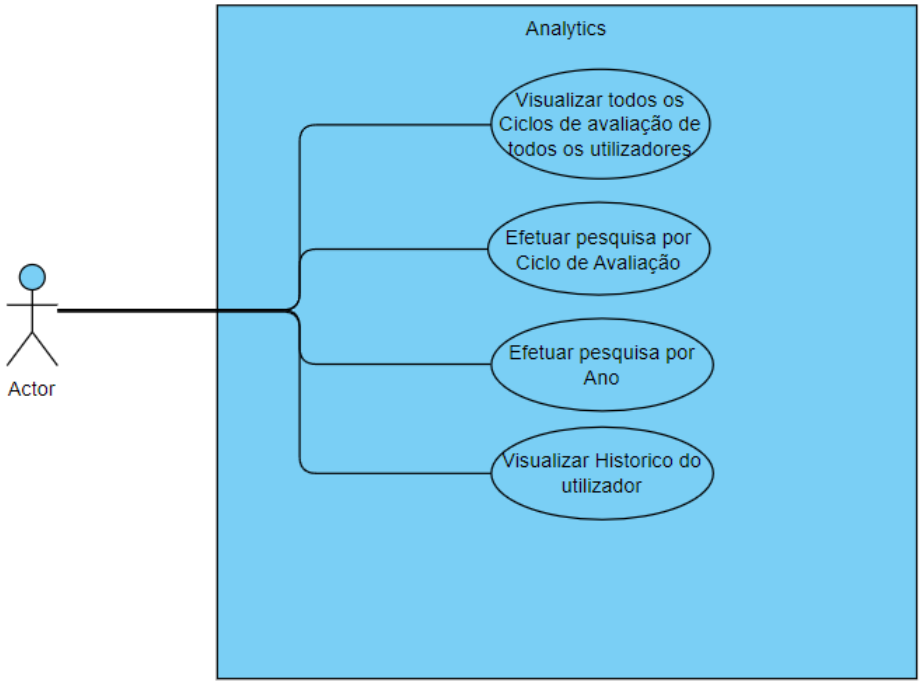

Figura 4.5: Analytics

ID	**Cabeçalho Descrição**	**Prioridade**
RF00	O sistema deve permitir a visualização de todos os ciclos para o qual o utilizador deve responder, inclusive os em que ele é supervisor e deve avaliar o funcionário.	Média
RF01	O sistema deve permitir a visualização dos resultados do utilizador e supervisor e quando acontecer poderá ser possível visualizar a comparação das respostas de ambos.	Alta
RF02	O sistema deve permitir realizar a escolha do ciclo de avaliação para o qual quer visualizar.	Média
RF03	O sistema deve permitir o agrupamento dos ciclos de avaliação por ano ou por tipo.	Média

Tabela 4.6: Histórico do Utilizado.

Caso de uso

Os casos de uso serão uma representação do que foi descrito anteriormente (Figura 4.6).

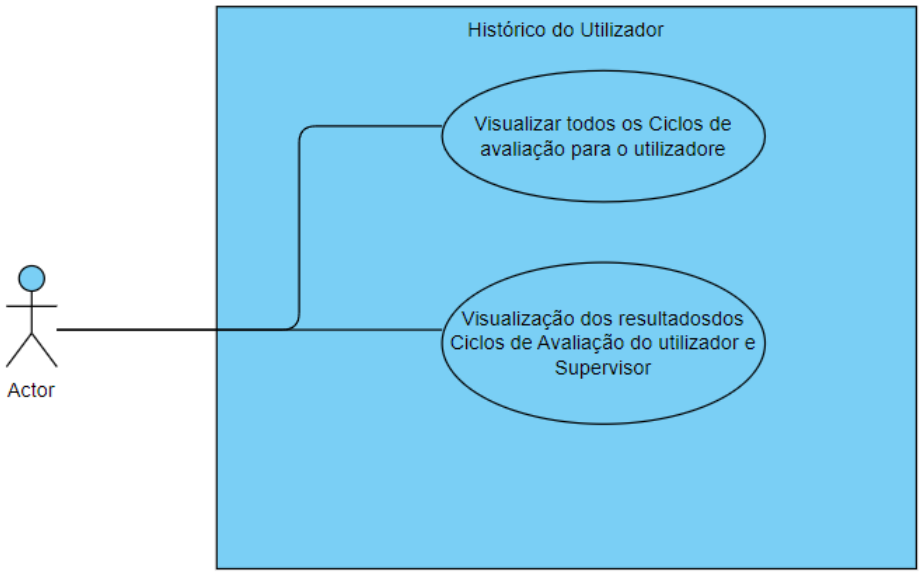

Figura 4.6: Histórico do Utilizador

4.2.7 Responder ao Ciclo de avaliação

Descrição sucinta

O utilizador poderá realizar a resposta aos ciclos de avaliação, o sistema deve permitir que a qualquer momento o utilizador para de responder e que continue mais tarde.

Requisitos funcionais associados

Os requisitos funcionais afetos à caracterıstica do sistema em epıgrafe encontram-se documentados na Tabela 4.7.

Caso de uso

Os casos de uso serão uma representação do que foi descrito anteriormente (Figura 4.7).

ID	Cabeçalho Descrição	Prioridade
RF00	O sistema deve permitir a realização das respostas ao ciclo de avaliação pela ordem que utilizador desejar	Média
RF01	O sistema deve permitir que a resposta seja salva, sem estar completa	Média
RF02	o sistema deve permitir a finalização de uma resposta	Média

Tabela 4.7: Responder ao Ciclo de avaliação.

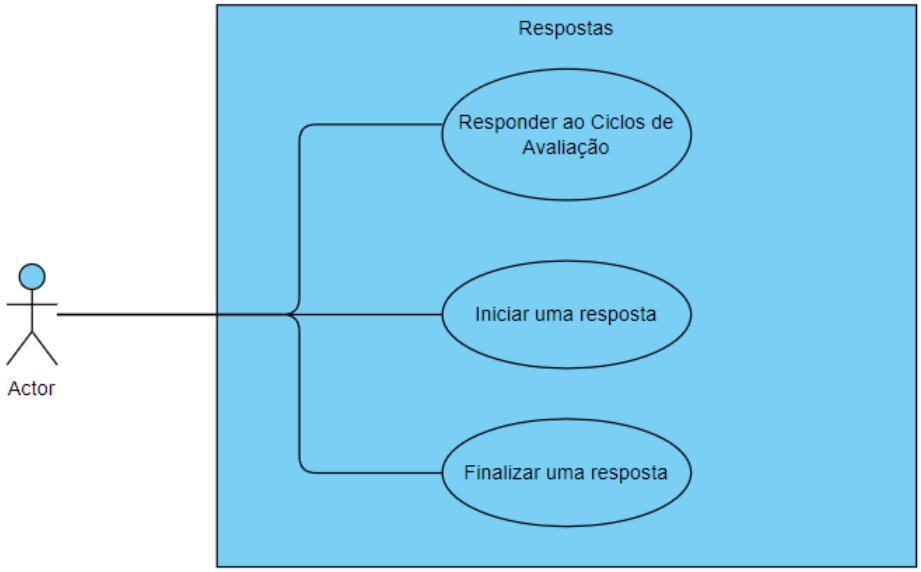

Figura 4.7: Responder ao Ciclo de avaliação

4.3 Base de Dados

Após realizada a análise dos mocks disponibilizados pela GNG, foi possível realizar a extração das seguintes entidades necessárias para a construção da base de dados NoSQL em Mongodb (Figura 4.8).

4.3.1 Ciclos de Avaliação

Os ciclos de avaliação são constituídos por Id, Type que irá definir o tipo de ciclo de avaliação que para a GNG são permanentes e experimentais. Name, Year corresponde ao ano de realização e é apenas utilizado para quando o ciclo de avaliação é do tipo permanente. StartDate data de início e EndDate são campos que só são utilizados em ciclos de avaliação do tipo permanente. Users corresponde a listagem de objetos do tipo utilizador que ficaram associados ao ciclo de avaliação. QuestionnaireId corresponde ao id do questionário, state indica o estado do ciclo de avaliação como por exemplo started ou finished e Notification este campo irá guardar as datas em que as notificações serão enviadas aos utilizadores.

4.3.2 Questionário

Os Questionários são compostos por Id, Name, Type que irá definir o tipo de ciclo de avaliação que para a GNG são permanentes e experimentais, State e lista de Categories.

4.3.3 Categorias

As Categorias serão compostas por Id, Name, IsOpenAnswer que será um campo boleano que irá ajudar o sistema a perceber se a resposta é por extenso, se for true, caso contrário a resposta é numérica, Weighting que corresponderá à ponderação da categoria, Description, Level que irá permitir saber o nível da categoria, por exemplo, Level 0 é uma categoria Level 1 é uma subcategoria, lista de Questões e lista de subcategorias.

4.3.4 Questões

A coleção Questões ser constituídas por Question, Number, Description e Weighting.

4.3.5 Utilizadores

A coleção Utilizador será constituída por Id e por SupervisorId esta tabela tem poucos dados, devido á proteção de dados, ou seja, toda a informação do utilizador está guardada na plataforma do Auth0.

4.3.6 Questionário Resolvido

A coleção Questionários resolvidos será constituída por UserId e AssociatedId, corresponde ao id utilizador que o user está a supervisionar, caso não exista, este campo é vazio, EvaluationCycleId, AnswerList, State e FinalGrade.

4.3.7 Respostas

A Coleção Respostas é constituída por Answer, QuestionnaireId, QuestionId, EvaluationCycleId e por último Grade.

4.4 Sumário

Neste capítulo, foram explorados os requisitos funcionais propostos pela GNG para o desenvolvimento deste projeto. Além disso, foi também estabelecido o modelo para a base de dados. Por fim, descreveu-se a metodologia proposta para o desenvolvimento do projeto, fornecendo uma visão clara do planeamento e das etapas de execução.

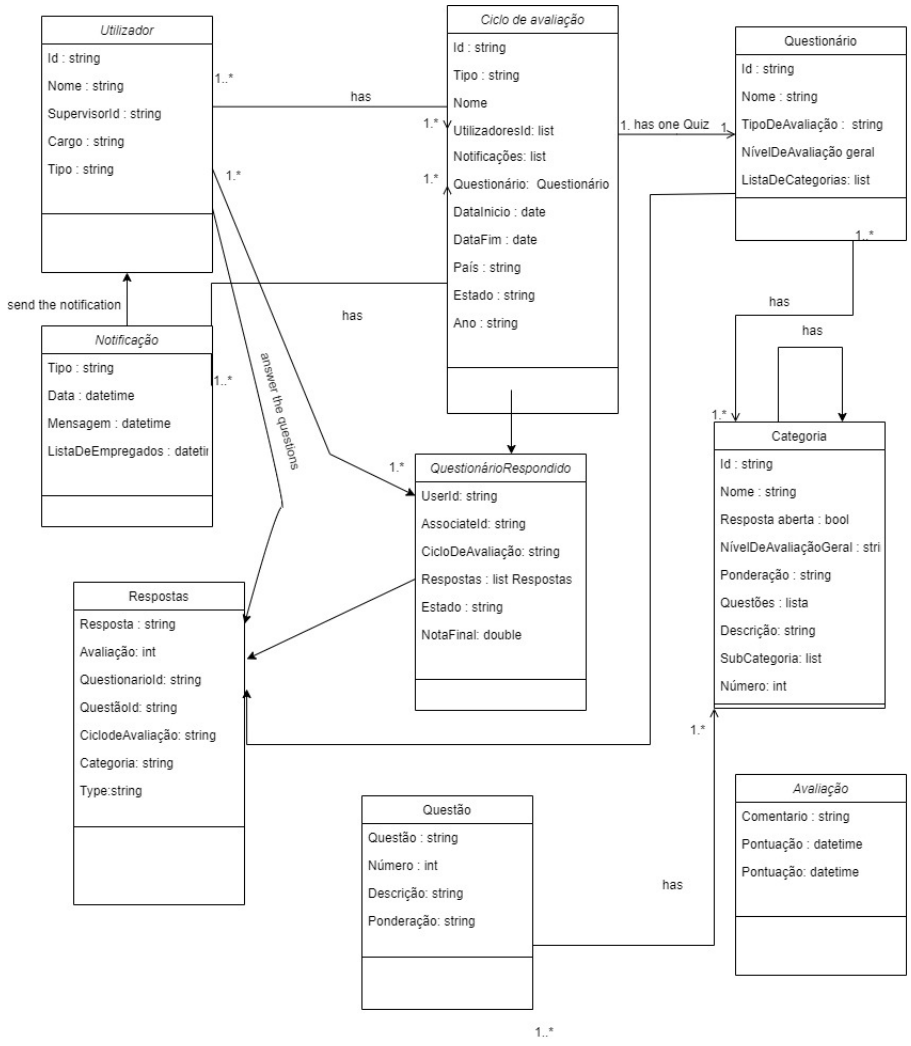

Figura 4.8: Diagrama da base de dados.

Capítulo 5

Desenvolvimento do projeto

Neste capítulo será apresentada a estruturação necessária para que seja possível a realização de todos os desenvolvimentos necessários para o bom funcionamento deste projeto, bem como a discussão dos testes realizados.

5.1 Discussão dos desenvolvimentos

A primeira etapa para o desenvolvimento do projeto consistiu numa série de reuniões com a empresa alvo, para que fosse realizado o levantamento de requisitos, discutindo o problema que a API irá resolver e a ordem pelo qual os desenvolvimentos iria seguir.

5.1.1 Problema encontrado

A API desenvolvida tem como objetivo responder ás necessidades existentes na GNG para entrega e realização de avaliações aos seus funcionários, visto que a mesma possui inúmeras lojas por todo pais e até ao momento todos os funcionários estão a responder a estes questionários numa folha de excel, o que dificulta a obtenção de uma visão geral do funcionamento de uma loja e do comportamento dos seus funcionários.

5.1.2 Desenvolvimento

No que diz respeito ao desenvolvimento da RESTAPI, foi tomada a decisão de dividir os desenvolvimentos por 3 partes principais Questionários, Ciclos de Avaliação e por último respostas ao Ciclos de Avaliação.

Criação de questionários

O primeiro *endpoint* a ser desenvolvido consistiu na criação dos questionários. Este foi desenvolvido de forma a poder ter dois tipos de questionários para funcionários permanentes e para funcionários que se encontram em período experimental. Para além do tipo, os questionários possuem a capacidade de abordarem um elevado número de categorias e subcategorias. Dentro das categorias/subcategorias serão inseridas as questões para o questionário e poderão ser inseridas as propriedades, que irão afetar as questões, tais como, o tipo de resposta, a ponderação da categoria ou se a própria categoria será do tipo aberta ou não, ou seja, a categoria pode não contar para o resultado final do questionário. Condições para o bom funcionamento de um questionário (Figura 5.1).

```
/// <summary>
/// Add Question to the database
/// </summary>
/// <param name="pQuestionnaire">Question to be added.</param>
/// <returns code="200">Returns the created attendance.</returns>
/// <returns code="400">Returns an error explaining the invalid part of the request.</returns>
/// <returns code="500">Returns an error explaining why the server did not process the request.</returns>
[SwaggerOperation(Description = "This endpoint Creates one Questionnaire.", Tags = new string[] { "Questionnaire" }
[HttpPost]
[ProducesResponseType(typeof(QuestionnaireResponseDto), StatusCodes.Status200OK)]
[ProducesResponseType(typeof(Error), StatusCodes.Status400BadRequest)]
[ProducesResponseType(typeof(Error), StatusCodes.Status500InternalServerError)]
5 references
public ActionResult<QuestionnaireResponseDto> PostQuestionnaire(QuestionnaireRequestDto pQuestionnaire)
{
    if (pQuestionnaire.NrLevel > 4)
    {
        throw new TheNumberOfLevelsIsGreaterThanFourCustomException(pQuestionnaire.Name);
    }
    var questionnaireType = myDatabaseContext.EvaluationCycleTypeCollection.GetById(pQuestionnaire.Type);
    var adapter = new QuestionnaireAdapter();
    var questionnaire = adapter.Adapt(pQuestionnaire);
    var questionnaireResponse = myService.AddQuestionnaire(questionnaire);
    return Ok(questionnaireResponse);
}
```

Figura 5.1: legenda

- Quando um questionário está a ser criado não precisa de ter todos os campos preenchidos, para poder ser gravado na base de dados.

- Um questionário quando é gravado na base de dados está sempre num estado de *Draft*, ou seja, ainda não se encontra apto para ser associado a um ciclo de avaliação.

- Quando as categorias estão a ser adicionadas, se as suas respostas forem necessárias para o resultado final, é necessário que a soma da sua ponderação com as restantes seja 100% no final.

Iniciação de um Questionário.

O *endpoint* para a iniciação de um questionário foi desenvolvido para validar se o questionário está preparado para ser utilizado pelos ciclos de avaliação, assim sendo, um número de regras terão de ser validadas para que o estado do questionário passe de *draft* para *Ready to Use*.

- O nome do Questionário deve ser único.

- Deve possuir pelo menos uma categoria.

- A soma das ponderações das categorias deverá ser 100%.

- A soma da ponderação das questões dentro de uma categoria deverá ser 100 se as mesmas não forem do tipo resposta aberta.

- O estado do questionário deverá ser do tipo *Draft*.

Apagar Questionários.

O *Endpoint* apagar questionário irá servir para apagar questionários que estejam no estado *Draft* ou *Ready to Use* (Figura (5.2).

```
2 references
public ActionResult<QuestionnaireResponseDto> DeleteQuestionnaire(string pId)
{
    if (pId == null)
    {
        throw new CantDeleteQuestionnaire();
    }
    var questionnaireToDelete = myDatabaseContext.QuestionnaireCollection.GetById(pId);
    if (questionnaireToDelete == null)
    {
        throw new EntityXDoesNotExistsCustomException("Questionnaire");
    }

    if (questionnaireToDelete.State == EQuestionnaireState.Deleted.ToString())
    {
        throw new EntityXIsAlreadyDeletedCustomException("Questionnaire");
    }

    questionnaireToDelete.DeleteQuestionnaire(questionnaireToDelete.Name);
    myDatabaseContext.QuestionnaireCollection.Update(questionnaireToDelete);
    var questionnaireType = myDatabaseContext.EvaluationCycleTypeCollection.GetById(questionnaireToDelete.Type);
    var adapter = new QuestionnaireAdapter();
    var questionnaireResponseDto = adapter.Adapt(questionnaireToDelete, questionnaireType.Value);
    return Ok(questionnaireResponseDto);
}
```

Figura 5.2: Delete Questionários

Cancelar Questionários.

O *Endpoint* cancelar questionário irá servir para cancelar questionários que estejam no estado *Draft* ou *Ready to Use*.

Atualizar questionários.

O *Endpoint* atualizar questionários irá servir para realizar a atualização de questionários, no entanto apenas questionários que se encontrem no estado *draft* ou *Ready to use*, é que poderão ser utilizados. Um questionário que esteja no estado *Ready to Use* irá passar para o estado *Draft* pois será sempre necessário validar se as atualizações respeitam as regras do *endpoint* de inicialização de um questionário (Figura 5.3).

Get Questionários.

O Get questionários como o próprio nome indica, irá buscar os questionários. No entanto este get possui algumas propriedades especiais, ou seja, o get quando for realizado terá os seguintes campos, pPage corresponde ao número da página, pRecords corresponde ao numero de valores que vão ser retornados, pField corresponde ao valor que será procurado

```
3 references
public ActionResult<QuestionnaireResponseDto> Put(string pId, QuestionnaireRequestDto pQuestionnaireRequest)
{
    var questionnaireToUpdate = myService.GetQuestionnaireById(pId);

    if (questionnaireToUpdate == null)
    {
        throw new EntityXDoesNotExistsCustomException("Questionnaire");
    }

    if (questionnaireToUpdate.State != EQuestionnaireState.ReadytoUse.ToString() && questionnaireToUpdate.State !=
    {
        throw new UpdateActiveQuestionnaireCycleCustomException();
    }
    var questionnaireAdapter = new QuestionnaireAdapter();
    var questionnaireType = myDatabaseContext.EvaluationCycleTypeCollection.GetById(pQuestionnaireRequest.Type);
    questionnaireToUpdate = questionnaireAdapter.Adapt(pQuestionnaireRequest, questionnaireToUpdate);
    var questionnaireResponse = myService.UpdateQuestionnaire(questionnaireToUpdate);

    var questionnaireResponseDto = questionnaireAdapter.Adapt(questionnaireResponse, questionnaireType.Value);
    return Ok(questionnaireResponseDto);
}
```

Figura 5.3: Cancelar Questionários.

por todos os campos da entidade e por último o pQuery que irá permitir realizar *querys* a base de dados, para fazer uma pesquisa mais especifica por exemplo type = Permanent And/and Name = Avaliação de gerente.

Duplicar Questionários.

O *endpoint* duplicar questionários surgiu por causa da necessidade de reutilizar um questionário que esteja em uso por um Ciclo de avaliação, assim sendo ao duplicarmos um questionário, será possível ter uma cópia de um questionário que até ao momento não podia ser atualizado.

Criar Ciclos de Avaliação.

O *endpoint* criar Ciclos de avaliação será utilizado para criar os dois tipos de ciclos de avaliação que serão utilizados pela GNG, que são ciclos de avaliação experimentais e ciclos de avaliação permanentes, no entanto, estes ciclos são bastante distintos, ou seja, quando se estiver a criar um ciclo de avaliação do tipo permanente deverá ser definida uma data para o qual deverá ter início e fim. Será possível escolher se o ciclo de avaliação apenas será respondido por funcionários ou por funcionário e respetivo supervisor. Por

último será possível escolher diversas datas para o envio de notificações, quer para uns funcionários, como para os respetivos supervisores, no entanto quando se cria um ciclo de avaliação experimental não haverá necessidade de introduzir datas de início e de fim porque o ciclo estará sempre a correr o que irá substituir a data de início e de fim será a data de contratação e a data de fim de contrato, neste ciclos será sempre obrigatório o funcionário e o supervisor responderem aos questionários.

Iniciar ciclos de avaliação.

O *endpoint* para iniciar ciclos de avaliação será responsável por validar que o ciclo de avaliação tem todos os parâmetros necessários para ser iniciado, caso todos os parâmetros sejam respeitados o estado do ciclo ira passar para o estado Iniciado o que ira fazer com que o ciclo de avaliação não possa mais ser atualizado com a exceção da alteração de utilizadores associados.

Finalizar ciclos de avaliação.

O *endpoint* para finalizar ciclos de avaliação será utilizado para alterar o estado do ciclo de avaliação para finalizado, ou seja, não poderão ser realizadas mais respostas ao ciclo. No entanto este *endpoint* não será acionado manualmente, mas sim através de uma DAG que irá correr todos os dias e validar se a data de fim chegou para algum ciclo de avaliação. Caso isso aconteça, o *endpoint* será executado. É preciso referir que este *endpoint* só tem efeito para ciclos de avaliação permanentes (Figura 5.4).

Cancelar ciclos de avaliação.

O *Endpoint* cancelar ciclos de avaliação irá servir para cancelar ciclos de avaliação que estejam no estado *Draft* ou Iniciados.

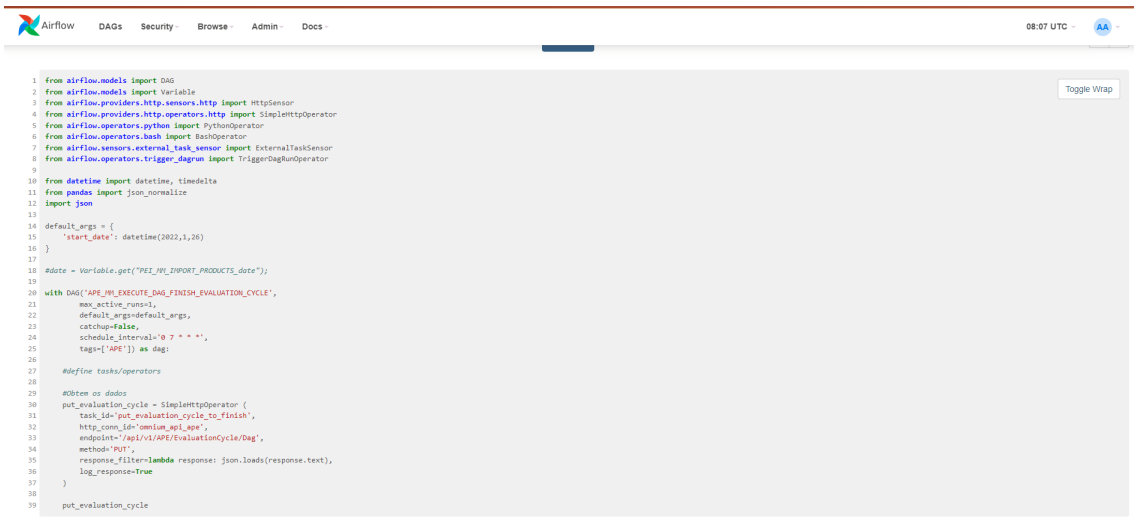

Figura 5.4: Consumidor de envio de notificações.

Apagar ciclos de avaliação.

O *Endpoint* apagar ciclos de avaliação irá servir para apagar ciclos de avaliação que estejam no estado Draft ou Iniciados.

Get Ciclos de avaliação.

O Get Ciclos de avaliação como o próprio nome indica, irá buscar os ciclos de avaliação. No entanto este get possui algumas propriedades especiais, ou seja, o get quando for realizado terá os seguintes campos, pPage corresponde ao número da página, pRecords corresponde ao número de valores que vão ser retornados, pField corresponde ao valor que será procurado por todos os campos da entidade e por último o pQuery que irá permitir realizar querys à base de dados, para fazer uma pesquisa mais especifica por exemplo type = Permanent And/and Name = Avaliação de gerente.

Envio de notificações em grupo.

O *endpoint* para envio de notificações será utilizado para agrupar e enviar todas as notificações de todos os ciclos de avaliação, que tenha chegado á data para o fazer. No caso de ciclos de avaliação permanente, serão enviadas as notificações quando as datas

selecionadas corresponderem á data atual. No caso de ciclos de avaliação experimentais as notificações serão enviadas, se a data de início de contrato, mais 30 dias, for igual á atual ou se a data de fim de contrato, menos 45 dias, for igual a data atual, tanto os 30 dias de início como os 45 dias antes do fim são dados parametrizáveis, na base de dados. Para além da criação do *endpoint* também foi criada uma DAG que irá correr diariamente e executar o *endpoint* explicado anteriormente (Figura 5.6).

Envio de notificações assíncrono.

O *endpoint* para envio de notificações assíncronas irá criar um evento que será consumido pelo rabbitmq, em seguida, esse mesmo evento irá despoletar a iniciação do *endpoint* para envio de notificações (Figura 5.5).

```
0 references
public async Task Consume(ConsumeContext<NotificationIntegrationRequestCreatedEvent> pContext)
{
    var sourcePlatform = myDatabaseContext.PlatformCollection.GetById("1");
    var integrationData = JsonSerializer.Deserialize<IntegrationData>(sourcePlatform.IntegrationData);

    var evaluationCyclesSize = myDatabaseContext.EvaluationCycleCollection.CountDocuments(new BsonDocument());
    var evaluationCyclesList = myDatabaseContext.EvaluationCycleCollection.GetEvaluationCyclesAsync(1, unchecked((int)evaluationCyclesSize), "Type", "asc", "", "");
    var pSortBy = "DisplayName";
    var pSortDirection = "asc";
    string pTerms = null;
    string pQuery = null;

    var userAdapter = new UserAdapter();
    var tenantName = myDatabaseContext.ParameterizationCollection.GetById("13");
    var tenantNameUrl = myDatabaseContext.ParameterizationCollection.GetById("14");
    var token = await myAccessTokenService.GetTokenAsync(tenantName.Value, tenantNameUrl.Value);
    var content = await myGateway.GetUsers(1, 1, pSortBy, pSortDirection, pTerms, pQuery, null, integrationData.BaseURL, token.Token);
    var userAACPResponseOne = JsonSerializer.Deserialize<UserAACPResponse>(content);

    var userList = await myGateway.GetUsers(1, (int)userAACPResponseOne.Count, pSortBy, pSortDirection, pTerms, pQuery, null, integrationData.BaseURL, token.Token);
    var userAACPResponseALL = JsonSerializer.Deserialize<UserAACPResponse>(userList);
    var users = userAACPResponseALL.Users;
    var dbUser = userAdapter.AdaptUserAACP(userAACPResponseALL);
    myNotificationService.AddBatchNotificationsAsync(evaluationCyclesList.Result.Item1, dbUser,token.Token);
}
```

Figura 5.5: Consumidor de envio de notificações.

Para que o envio de notificações não dependa de nenhum utilizador, será criada uma DAG que irá correr diariamente, para despoletar o envio de notificações assíncronas (Figura 5.6).

Create Resposta.

O endpoint Creat resposta será utilizado para instanciar na base de dados uma resposta(Figura 5.7).

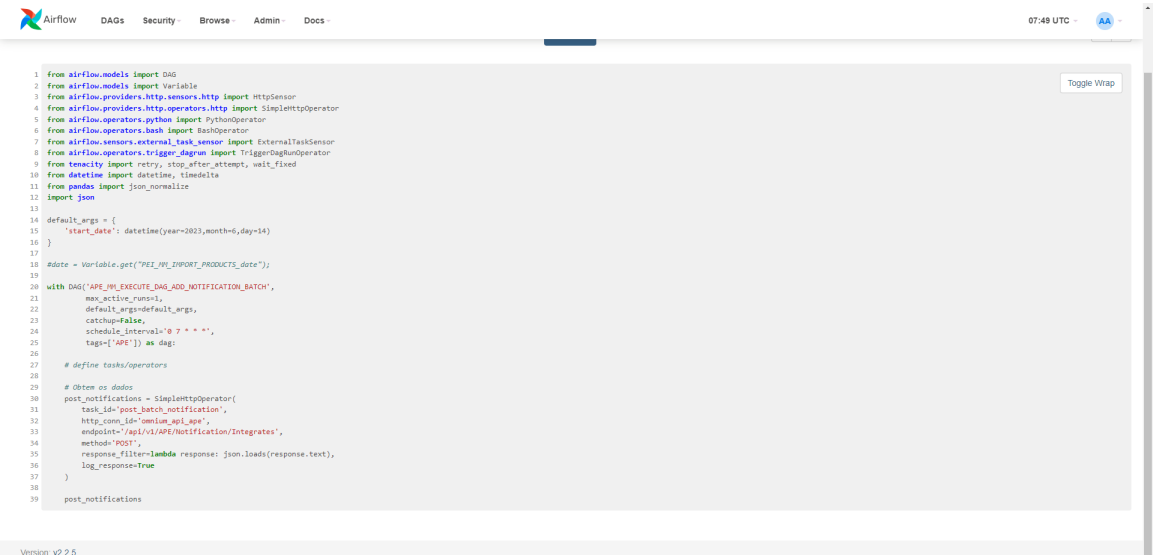

Figura 5.6: Dag para envio de notificações.

```
[ProducesResponseType(typeof(Error), StatusCodes.Status500InternalServerError)]
8 references
public ActionResult<AnswerResponseDto> PostAnswer(string pQuestionnaireId, string pEvaluationCycleId, string pUserId, string pTargetUserId = null)
{
    var dbQuestionnaire = myDatabaseContext.QuestionnaireCollection.GetById(pQuestionnaireId);
    if (dbQuestionnaire == null)
    {
        throw new EntityXDoesNotExistsCustomException("Questionnaire");
    }

    var dbEvaluationCycle = myDatabaseContext.EvaluationCycleCollection.GetById(pEvaluationCycleId);
    if (dbEvaluationCycle == null)
    {
        throw new EntityXDoesNotExistsCustomException("EvaluationCycle");
    }

    var dbUser = myDatabaseContext.UserCollection.GetById(pUserId);
    if (dbUser == null)
    {
        throw new EntityXDoesNotExistsCustomException("User");
    }

    if (pTargetUserId != null)
    {
        var dbTargetUser = myDatabaseContext.UserCollection.GetById(pTargetUserId);
        if (dbTargetUser == null)
        {
            throw new EntityXDoesNotExistsCustomException("User");
        }
    }

    if (!dbEvaluationCycle.QuestionnaireId.Equals(dbQuestionnaire.Id))
    {
        throw new QuestionnaireDoesNotBelongToTheEvaluationCycleCustomException(dbQuestionnaire.Id);
    }

    if (dbEvaluationCycle.State != EEvaluationCycleState.Started.ToString())
    {
        throw new CannotAnswerAnUnstartedEvaluationCycleCustomException();
    }

    var adapter = new AnswerAdapter();
    var answer = adapter.Adapt(dbQuestionnaire, dbEvaluationCycle, pUserId, pTargetUserId);
    dbEvaluationCycle.UpdateUserAnswer(pUserId, answer.Id, null, pTargetUserId);
    myDatabaseContext.EvaluationCycleCollection.Update(dbEvaluationCycle);
    var answerResponse = myService.AddQuestionnaireAnswer(answer, dbUser);

    return Ok(adapter.Adapt(answerResponse));
```

Figura 5.7: Criar Respostas.

Atualizar Respostas.

O *endpoint* atualizar resposta será utilizado para dar uma resposta a um determinado ciclo de avaliação. Este *endpoint* está preparado para receber várias respostas, no entanto, é preciso uma especial atenção, a partir do momento que o ciclo tiver terminado este *endpoint* não irá funcionar para o mesmo.

Finalizar Respostas.

Este *endpoint* irá alterar o status da entidade resposta para finalizado, ou seja, a partir desse momento não será possível realizar mais nenhum update á resposta e será neste *endpoint* em que será calculado o valor total obtido pelo funcionário.

Visualização de respostas.

O *endpoint* visualização de respostas será utilizado para visualizar as respostas dadas a um ciclo de avaliação pelo funcionário e pelo respetivo supervisor caso o ciclo de avaliação o permita (Figura 5.8).

```
[SwaggerOperation(Description = "This endpoint gets Answer", Tags = new string[] { "Answer" })]
[HttpGet]
[ProducesResponseType(typeof(AnswerResponseList), StatusCodes.Status200OK)]
[ProducesResponseType(typeof(Error), StatusCodes.Status400BadRequest)]
[ProducesResponseType(typeof(Error), StatusCodes.Status500InternalServerError)]
0 references
public async Task<ActionResult<AnswerResponseList>> Get([MinValue(1)] int pPage = 1, [MinValue(1)] int pRecords = 50, [SortBy("Name", "Type", "Year", "StartDate", "EndDate")] stri
{

    Tuple<List<QuestionnaireAnswer>, long> dbAnswers;
    if (!string.IsNullOrEmpty(pTerms) || string.IsNullOrEmpty(pTerms) && string.IsNullOrEmpty(pQuery))
    {
        var pFields = myDatabaseContext.ParameterizationCollection.GetById("4").Value;
        dbAnswers = await myDatabaseContext.AnswerCollection.GetQuestionnaireAnswerAsync(pPage, pRecords, pSortBy, pSortDirection, pTerms, pFields);
    }
    else
    {
        dbAnswers = await myDatabaseContext.AnswerCollection.GetQuestionnaireAnswerByQueryAsync(pPage, pRecords, pSortBy, pSortDirection, pQuery);
    }
    var response = new AnswerResponseList();
    response.Count = dbAnswers.Item2;
    var answerResponseList = new List<AnswerResponseDto>();
    response.Answers = answerResponseList;

    if (dbAnswers.Item1.Any())
    {
        var answerAdapter = new AnswerAdapter();
        foreach (var dbAnswer in dbAnswers.Item1)
        {
            answerResponseList.Add(answerAdapter.Adapt(dbAnswer));
        }
    }

    return Ok(response);
}
```

Figura 5.8: Get Respostas.

Get Historico.

O *endpoint* get histórico será utilizado para realizar o get dos ciclos de avaliação, para o qual precisa de responder e também os que já efetuou a resposta, conseguindo assim, obter um histórico para um utilizador.

Get analytics.

O *endpoint* get analytics será utilizado para recolher a informação relativa ao estado de respostas de um ciclo de avaliação, ou seja, o get irá trazer todos os utilizadores que têm de responder a um ou mais ciclos de avaliação. Este get também permitirá a aplicação das pQuerys (Figura 5.9).

```
var users = userAACPResponseALL.Users;
var locationList = myDatabaseContext.LocationCollection.GetLocationAsync();
var numberOfDaysAfterStartedWorking = Convert.ToInt32(myDatabaseContext.ParameterizationCollection.GetById("5").Value);
foreach (var ev in dbEvaluationCycles.Item1)
{
    if ((ev.State == "Started" || ev.State == "Finished") && ev.Users != null)
    {
        foreach (var user in ev.Users)
        {
            if (ev.Type == "1" && DateTime.Now >= ev.StartDate)
            {
                var userContent = users.FirstOrDefault(b => b.UserId == user.UserId);
                if (userContent.UserId != null)
                {
                    var dbUser = userAdapter.AdaptUserToAPEDto(userContent);
                    responseUserList.Add(userAdapter.Adapt(dbUser, user.UserScore, user.SupervisorScore, ev, locationList));
                }
            }
            else if (ev.Type == "2")
            {
                var userContent = users.FirstOrDefault(b => b.UserId == user.UserId);
                var admissionDate = OmniumGlobalization.DateTimeManager.FromUnixTimeToDateTime(userContent.AdmissionDate.Value);
                var resignDate = OmniumGlobalization.DateTimeManager.FromUnixTimeToDateTime(userContent.EndContractDate.Value);
                if (DateTime.Now >= admissionDate.AddDays(numberOfDaysAfterStartedWorking).Date)
                {
                    if (userContent.UserId != null)
                    {
                        var dbUser = userAdapter.AdaptUserToAPEDto(userContent);
                        responseUserList.Add(userAdapter.Adapt(dbUser, user.UserScore, user.SupervisorScore, ev, locationList));
                    }
                }
            }

        }
    }
}

var responseUserListWithPagination = responseUserList.Skip((pPage - 1) * pRecords).Take(pRecords);
var evaluationCycleUsersDto = new EvaluationCycleUsersDto { Count = responseUserList.Count, Users = responseUserListWithPagination };
return Ok(evaluationCycleUsersDto);
```

Figura 5.9: Get Analytics.

Get Users.

O *endpoint* get Users será utilizado para recolher as informações dos utilizadores para posteriormente poderem ser identificados nos ciclos de avaliação pelo nome, no entanto,

este get não será realizado à base de dados que foi referida no capítulo anterior, mas sim a uma base de dados externa devido á lei de proteção de dados.

Para garantir essa segurança, foi implementada uma validação, através de um *token*, o que garantirá a segurança das informações dos utilizadores (Figura 5.10).

```
0 references
public async Task<ActionResult<UserAPEResponse>> GetThroughAACP([MinValue(1)] int pPage = 1, [MinValue(1)] int pRecords = 50, [SortBy("Name", "Type", "Year", "StartDate", "EndDate", "DisplayName", "Code")] string pSortBy =
{
    var token = HttpContext.Request.Headers["Authorization"].FirstOrDefault().Replace("Bearer ", "");
    var sourcePlatform = myDatabaseContext.PlatformCollection.GetById("1");
    var integrationData = JsonSerializer.Deserialize<IntegrationData>(sourcePlatform.IntegrationData);

    var usersThatBelongToTheEvaluationCycle = new List<string>();
    var contentForAllUsers = await myGateway.GetUsers(1, 1, pSortBy, pSortDirection, pTerms, pQuery, null, integrationData.BaseURL, token);
    var userAACPResponseOne = JsonSerializer.Deserialize<UserAACPResponse>(contentForAllUsers);
    if (userAACPResponseOne.Count == 0)
    {
        var reponseEmpty = new UserAPEResponse { Users = null, Count = userAACPResponseOne.Count };
        return reponseEmpty;
    }
    var userList = await myGateway.GetUsers(1, (int)userAACPResponseOne.Count, pSortBy, pSortDirection, pTerms, pQuery, null, integrationData.BaseURL, token);

    if (pEvaluationCycleId != null)
    {
        var evaluationCycle = myDatabaseContext.EvaluationCycleCollection.GetById(pEvaluationCycleId);
        if (evaluationCycle != null)
        {
            foreach (var user in evaluationCycle.Users)
            {
                usersThatBelongToTheEvaluationCycle.Add(user.UserId);
            }
        }
    }

    var LocationList = myDatabaseContext.LocationCollection.GetLocationAsync();
    var userAdapter = new UserAdapter();
    var response = userAdapter.Adapt(userList, usersThatBelongToTheEvaluationCycle, LocationList);

    response.Users = response.Users.Skip((pPage - 1) * pRecords).Take(pRecords).ToList();

    return Ok(response);
}
```

Figura 5.10: Get Users.

5.2 Testes

Durante a fase de desenvolvimento foram criados testes unitários para todos os requisitos funcionais e para casos de erro, para garantir que as mensagens de erro eram percetíveis e o programa funcionava corretamente. Tendo os testes unitários todos desenvolvidos e todos *endpoints* concluídos houve uma passagem para produção da API. Em seguida foram realizados teste de integração tais como:

- Criar ciclo permanente sem utilizadores, sem notificações, sem questionário e guardar.
- Iniciar ciclo sem questionário.
- No ciclo de avaliação, as notificações devem ficar associadas ao utilizador selecionado e ao seu supervisor.

- Remover questionário de ciclo draft.
- Alterar questionário de ciclo iniciado
- Alterar notificações em ciclo draft (Tipo de ciclo Permanente).
- Duplicar um questionário.
- Visualizar respostas como supervisor sem o utilizador ter respondido.
- O responder aos ciclos de avaliação como supervisor.

Estes foram alguns dos teste de integração realizados. No total foram criados por volta de 140 testes de integração.

Em seguida foi disponibilizada uma demo para a loja da Levis do NortShoping onde foram criados três ciclos de avaliação e associados aos empregados da loja. Estes ciclos tiveram um período de dois meses para poderem ser realizados durante os quais os empregados irão receber notificações via email a notificá-los da necessidade de resposta ao ciclo de avaliação que lhes tinha sido designado, no entanto se este estivesse associado a dois ciclos, teria de receber dois emails. Durante este período foi se mantendo uma especial atenção ao *endpoint* das Analytics pois este permitia saber se os utilizadores já tinham iniciado a resposta ou se os seus supervisores já tinham realizado as avaliações dos mesmos.

5.3 Sumário

Neste capítulo, foram demonstrados os avanços alcançados ao longo do desenvolvimento deste projeto. Foram realizados testes minuciosos para assegurar o bom funcionamento da API, juntamente com os testes executados num ambiente de produção para obter feedback valioso, a fim de aprimorar ainda mais a API futuramente.

Capítulo 6

Conclusão e desenvolvimentos Futuros

6.1 Conclusão

O presente projeto teve como objetivo o desenvolvimento da API para a criação de ciclos de avaliação voltados à avaliação de funcionários, foi possível estabelecer uma estrutura eficiente para gerir de forma sistemática e eficaz o processo de avaliação de desempenho. Através da implementação cuidadosa de endpoints e funcionalidades, o projeto oferece uma solução flexível e escalável para a criação, monitorização e análise de ciclos de avaliação.

Durante a fase de desenvolvimento foram sempre encontrados novos desafios devido ás necessidades propostas pelo cliente, no entanto foi possível criar uma API que satisfez todas as suas necessidades e posteriormente a API foi construída de forma a que futuras empresas possam usufruir deste projeto sem ter qualquer tipo de dependência com o projeto da GNG.

6.2 Trabalho Futuro

Com a conclusão do projeto e da sua testagem foram observados alguns pontos que seriam uma melhoria necessária para o futuro da API.

- Adicionar a possibilidade de questionários que aceitem uma foto como resposta.
- Adicionar novos ciclos de avaliação.
- Adicionar nova versão para Espanha.

Bibliografia

[1] URL: https://omnium-retail.com.

[2] URL: https://www.gng.pt/en/.

[3] W. Khan, W. Ahmad, B. Luo e E. Ahmed, "SQL Database with physical database tuning technique and NoSQL graph database comparisons", em *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 2019, pp. 110–116. DOI: 10.1109/ITNEC.2019.8729264.

[4] URL: https://cassandra.apache.org/_/index.html.

[5] URL: https://www.mongodb.com/cloud/atlas/lp/try2?utm_source=google&utm_campaign=gs_emea_portugal_search_core_brand_atlas_desktop&utm_term=mongodb&utm_medium=cpc_paid_search&utm_ad=e&utm_ad_campaign_id=12212624551&adgroup=115749716383&gclid=Cj0KCQiAw9qOBhC-ARIsAG-rdn6pDPPt4ITETPamshu0ZSVlp_VuAeW2ykisDlLKrCu5BJbRYey7128aAlmoEALw_wcB.

[6] URL: https://visualstudio.microsoft.com/vs/features/.

[7] URL: https://www.postman.com/.

[8] URL: https://www.rabbitmq.com/.

[9] C. P. George, D. Z. Wang, J. N. Wilson, L. M. Epstein, P. Garland e A. Suh, *A Machine Learning Based Topic Exploration and Categorization on Surveys*, 2012. DOI: 10.1109/ICMLA.2012.132.

[10] URL: https://www.typeform.com/?_gl=1*zb1gl6*_up*MQ..&gclid=CjwKCAjws9ipBhB1EiwAccEi1PKHFyp2AwsK81eadrNyX5xSEuuSk2baXNP-ocrDeIHP3r4RKF2nJxoCiQkBwE&gclsrc=aw.ds&tf_campaign=EUROPE-Brand-Core-English-Combined&tf_source=google&tf_medium=paid&tf_content=150888936711_654632140542&tf_term=typeform.

[11] Y. Chaiyo e R. Nokham, *The effect of Kahoot, Quizizz and Google Forms on the student's perception in the classrooms response system*, 2017. DOI: 10.1109/ICDAMT.2017.7904957.

[12] B. H.T. e P. G., *Course End Survey: A Step Towards Quality Measure and Assurance*, 2017. DOI: 10.1109/MITE.2017.00009.

[13] J. Đurić e A. Mahmutović, *Software for Writing Online Exam with Video and Audio Surveillance - Cheatless*, 2021. DOI: 10.23919/MIPRO52101.2021.9596723.