

DEPARTAMENTO DE ELETRÓNICA, TELECOMUNICAÇÕES E INFORMÁTICA



**SERVICE STATUS UA**

ENGENHARIA INFORMÁTICA

***Autores:***

Eleandro Laureano, 83069

Frederico Avó, 79900

Gonçalo Pereira, 93310

José Mello, 87456

***Orientador:***

Prof. Cláudio Teixeira

**Junho, 2021**

# **Keywords**

* Status
* Service
* State

# **Agradecimentos**

Somos extremamente gratos ao nosso orientador Cláudio Teixeira por orientar-nos da melhor forma possível e também ao seu colega de trabalho Dimitri Silva que nos deu o input necessário e mostrou-se muito útil para ajudar-nos durante o desenvolvimento deste projeto.

# **Abstract**

\*\*\* por escrever \*\*\*

*Obs*: Definição de *Abstract*:

*“...An abstract is a condensed description of the project and should include the key points of the introduction, methods, results and conclusion.The results may be summarized qualitatively but avoid quantitative summaries.Do not use acronyms, abbreviations or references.Readers look at the abstract first: do your best to make a good impression. Use words carefully to capture reader interest and make an effort to show that the subject (problem) covered by your work is important…”*

[**Keywords**](#_6k1634o9r0hi) **2**

[**Agradecimentos**](#_8o5w3p38nq59) **3**

[**Abstract**](#_na2s5e4827hd) **4**

[**Abreviações**](#_w6eh04bsyghi) **9**

[**Introdução**](#_53qe8zfxna4k) **11**

[**1.1 Contexto**](#_8kmtawgnoghw) **11**

[**1.2 Motivação**](#_5vhj81asyy9g) **11**

[**1.3 Objetivos**](#_1p9byf7ugji2) **11**

[**1.4 Estrutura**](#_88x7ihqw7ikl) **12**

[**Requisitos e Arquitetura**](#_mhnn0mh92pxt) **13**

[**2.1 Requisitos do Sistema**](#_abpqcdepx7c7) **13**

[**2.2.1 Requisitos Funcionais**](#_c0erucmzvc1y) **13**

[**2.2.3 Requisitos Não Funcionais**](#_2upipn906qel) **13**

[**2.2.4 Atores e Casos de Uso**](#_rb0nvobpi7h) **14**

[**2.2 Arquitetura do Sistema**](#_44ksyvwoj53k) **15**

[**2.2.1 Domain Model**](#_b18xv276mdsp) **15**

[**2.2.2 Arquitetura**](#_pbvpb0b8pxg5) **16**

[**Estado da Arte**](#_hn7itqxamnum) **17**

[**3.1 Projetos Relacionados**](#_jia1pou9yv08) **17**

[**3.1.1 AWS Service Health Dashboard**](#_fe5kgyf259qa) **18**

[**3.1.2 Google Cloud Status Dashboard**](#_1gsmjdglcyi2) **18**

[**3.2 Tecnologia Utilizada**](#_p3aejpk2oqo3) **19**

[**3.2.1 REACT JS**](#_dzdgr5u2kubb) **19**

[**3.2.2 .NET CORE & C#**](#_lxhc7geffkls) **19**

[**3.2.3 Persistence: SQL**](#_o6kzo6jz9lf5) **19**

[**3.2.1 Modelo Tecnológico**](#_1wdrlr377as6) **19**

[**3.2.2 Ferramentas: Discord, Google Docs and Microsoft Teams**](#_4dnr7c4uy1wk) **20**

[**3.2.3 Ferramentas: Microsoft Azure and Github**](#_7byvz0c9nrj4) **21**

[**Implementação**](#_chsq8dqlqzd4) **21**

[**Conclusão**](#_plch1fjynxyd) **22**

[**5.1 Sumário**](#_c64q85uz50n0) **22**

[**5.2 Resultados**](#_t35t71ud9284) **22**

[**5.3 Trabalho Futuro**](#_rn69xe52hpd2) **23**

[**Referências**](#_e1u27bkufuts) **23**

# **Abreviações**

* **API**  - Application Program Interface
* **SCOM -** System Center Operations
* **MVC -** Model View Controller
* **UI -** User Interface
* **REST -** Representational State Transfer
* **JS -** Javascript
* **BD -** Base de Dados

# **Introdução**

# **1.1 Contexto**

Este projeto foi desenvolvido para a cadeira de Projeto em Informática no curso de Engenharia Informática da Universidade de Aveiro. Além disso, foi orientado pelo Prof. Cláudio Teixeira.

Sendo assim, o objetivo foi de desenvolver uma aplicação web que mostrasse a comunidade sobre os períodos de indisponibilidade dos serviços académicos da Universidade de Aveiro, bem como os períodos de manutenção dos mesmos e foi realizado durante o segundo semestre do ano letivo 2020/2021.

# **1.2 Motivação**

Então, dada a quantidade e a complexidade dos serviços oferecidos à comunidade académica, sentia-se a necessidade de melhorar a comunicação entre eles, relativamente ao estado de operação dos serviços e dos sistemas disponíveis dentro da Universidade. E, para melhorar essa comunicação seria necessário informar aos utilizadores das manutenções agendadas acerca de certos serviços e sistemas que poderão ou não causar a sua indisponibilidade. De igual modo, a aplicação web também poderá ser utilizada para informar em tempo real de problemas que estejam a ocorrer em determinados serviços que causem a sua indisponibilidade.

# **1.3 Objetivos**

Os objetivos para este projeto são de construir uma aplicação web que consiga fornecer informação relevante acerca dos serviços e sistemas da Universidade de Aveiro, relativamente à sua indisponibilidade. Além disso, deve fornecer ao utilizador a possibilidade de subscrever a um determinado serviço, de acordo com a sua preferência, com o objetivo de ser notificado em relação à sua disponibilidade.

# **1.4 Estrutura**

\*\*\* escrever acerca da estrutura/organização do documento (seções e subseções) \*\*\*

# 

# **Requisitos e Arquitetura**

# **2.1 Requisitos do Sistema**

Nesta seção, iremos falar dos requisitos do nosso sistema, tanto os funcionais como os não funcionais e explicar detalhadamente a arquitetura desenhada especificamente para o nosso sistema.

# **2.2.1 Requisitos Funcionais**

# Histórico de falhas de cada serviço

* Apresentar uma estimativa do tempo de resolução do problema
* Notificações sobre os períodos de indisponibilidade dos serviços
* Autenticação somente a nível do administrador da aplicação web

# **2.2.3 Requisitos Não Funcionais**

* **Performance:** é o quão rápida a aplicação web responde às ações efetuadas pelos utilizadores.
* **Prioridade**: Média
* **Usabilidade:** é a capacidade do sistema providenciar aos utilizadores condições para serem capazes de realizar as tarefas de forma eficaz e eficiente enquanto desfrutam da sua experiência.
* **Prioridade**: Média
* **Integridade dos dados:** é a capacidade de garantir a precisão e consistência dos dados apresentados.
* **Prioridade**: Alta
* **Intuitivo:** a aplicação web deve ser auto-explicativa.
* **Prioridade**: Média

# **2.2.4 Atores e Casos de Uso**

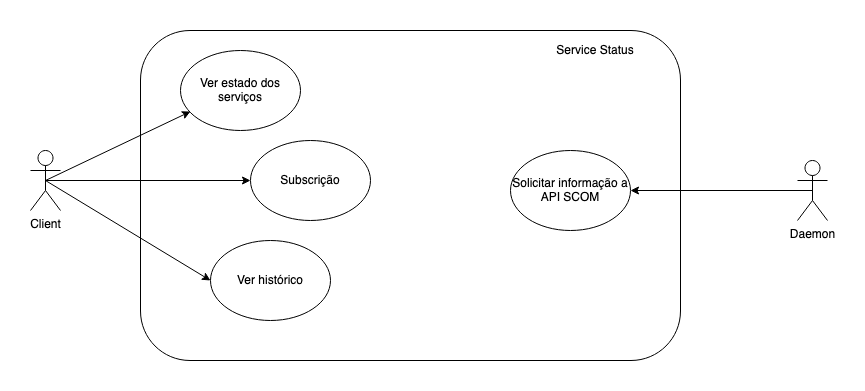
# Neste projeto, os utilizadores considerados como público-alvo são qualquer pessoa dentro da comunidade académica da Universidade de Aveiro, tal como, docentes, estudantes, investigadores, entre outros que são os clientes da aplicação web. Apesar disso, a aplicação web está disponível para utilizadores fora da comunidade académica da Universidade de Aveiro.

Sendo assim, os atores do sistema são:

* **Clientes:** os utilizadores do sistema que fazem parte da comunidade académica.
* **Daemon:** é um dispositivo externo que é responsável por fazer pedidos a API dentro de um intervalo de tempo.

Além disso, os use cases são os seguintes:

* Ver o estado dos serviços.
* **Prioridade**: Alta
* Subscrição.
* **Prioridade**: Alta
* Ver histórico.
* **Prioridade**: Média
* Solicitar informação à API SCOM.
* **Prioridade**: Alta



**obs: colocar uma legenda “Fig. 1 - Use Cases”**

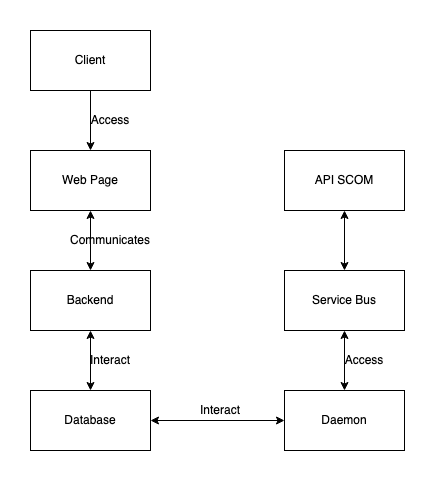
# **2.2 Arquitetura do Sistema**

Nesta seção, iremos abordar o que foi utilizado e implementado no nosso projeto, mostrando diagramas para ajudar a complementar e melhorar a sua compreensão.

# **2.2.1 Domain Model**

Em seguida, temos o seguinte diagrama para ajudar na compreensão do seguinte processo:

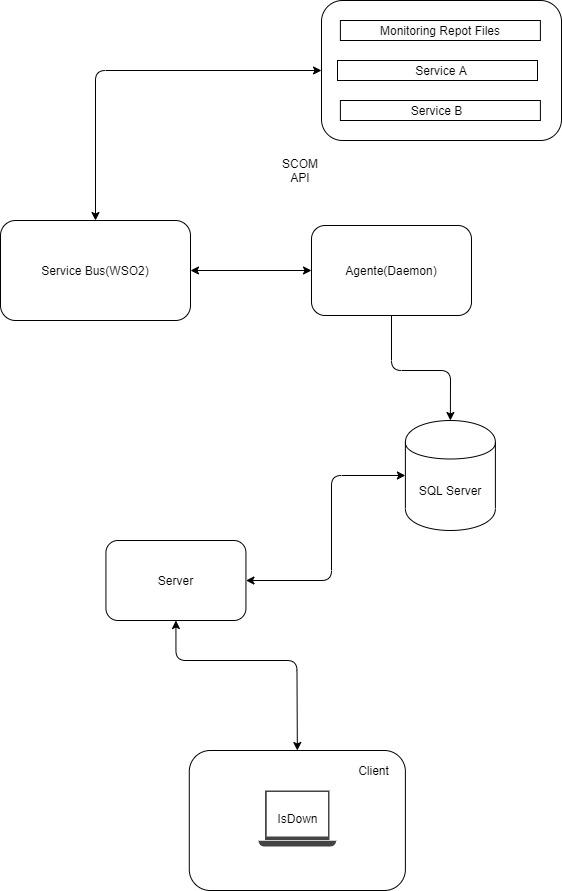
O cliente consulta a informação que está disponível ao interagir com a aplicação web. Esta aplicação web comunica com o Backend para a obtenção dos seus dados que já estão guardados na BD. Estes dados, que foram guardados pelo Daemon, que age como um Agente, sendo um dispositivo externo que corre em background e faz pedidos de 2 em 2 minutos ao Service Bus, que passa as mensagens através de filas, para atualizar os dados dos serviços e sistemas na BD, provêm da API SCOM.



**obs: colocar uma legenda “Fig. 2 - Domain Model”**

# **2.2.2 Arquitetura**

Em seguida, este próximo diagrama relativamente a arquitetura, representa essencialmente como o nosso sistema irá funcionar



**obs: colocar uma legenda “Fig. 3 - Architecture”**

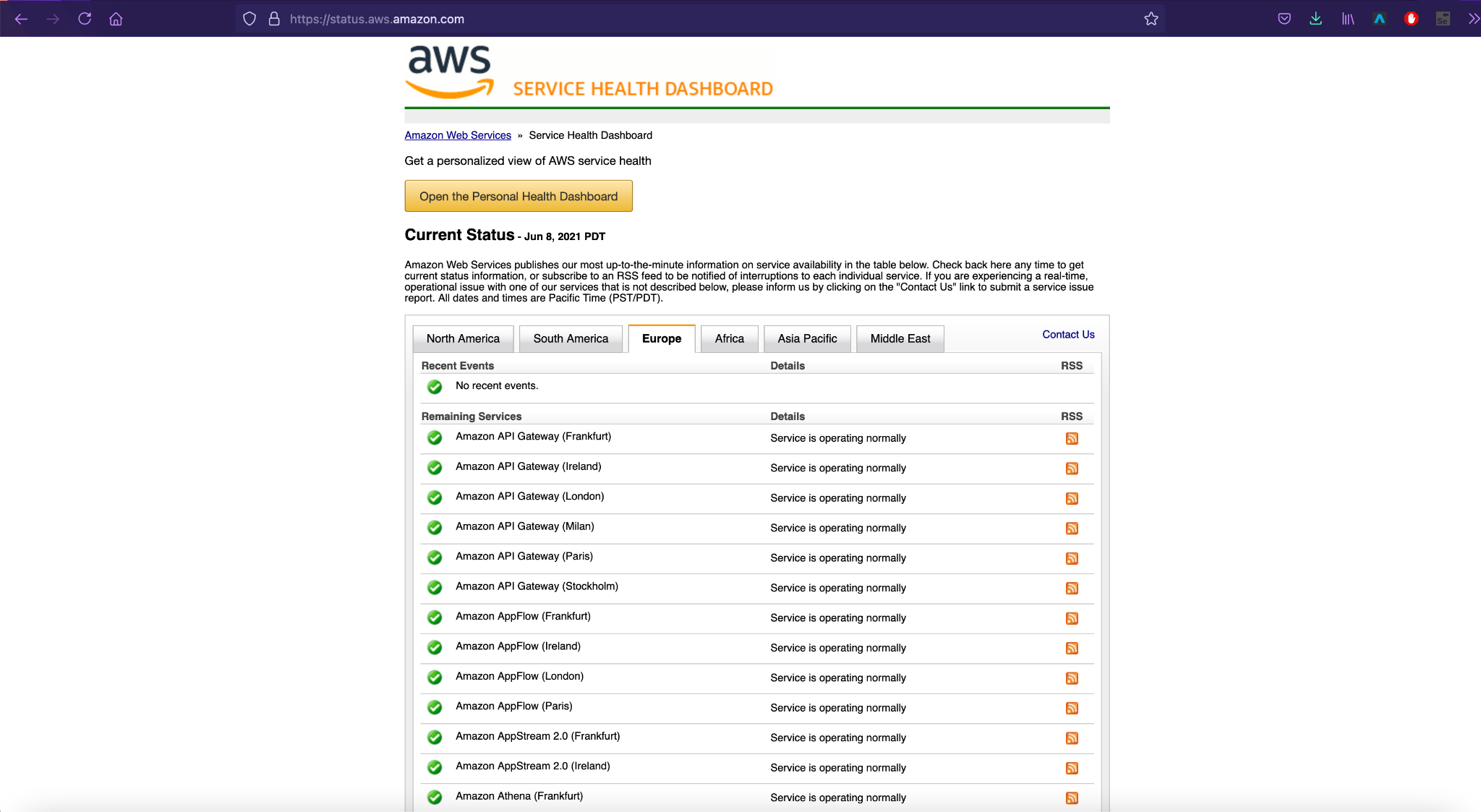
# **Estado da Arte**

# **3.1 Projetos Relacionados**

No que diz respeito aos projetos relacionados, irá mostrar-se em seguida exemplos de aplicações web que são idênticas a nossa. A principais diferenças são, que na nossa aplicação web é possível consultar a estimativa de tempo para a resolução de um problema, dada a indisponibilidade de um serviço e a subscrição para a receção de notificações de um determinado serviço.

# **3.1.1 AWS Service Health Dashboard**

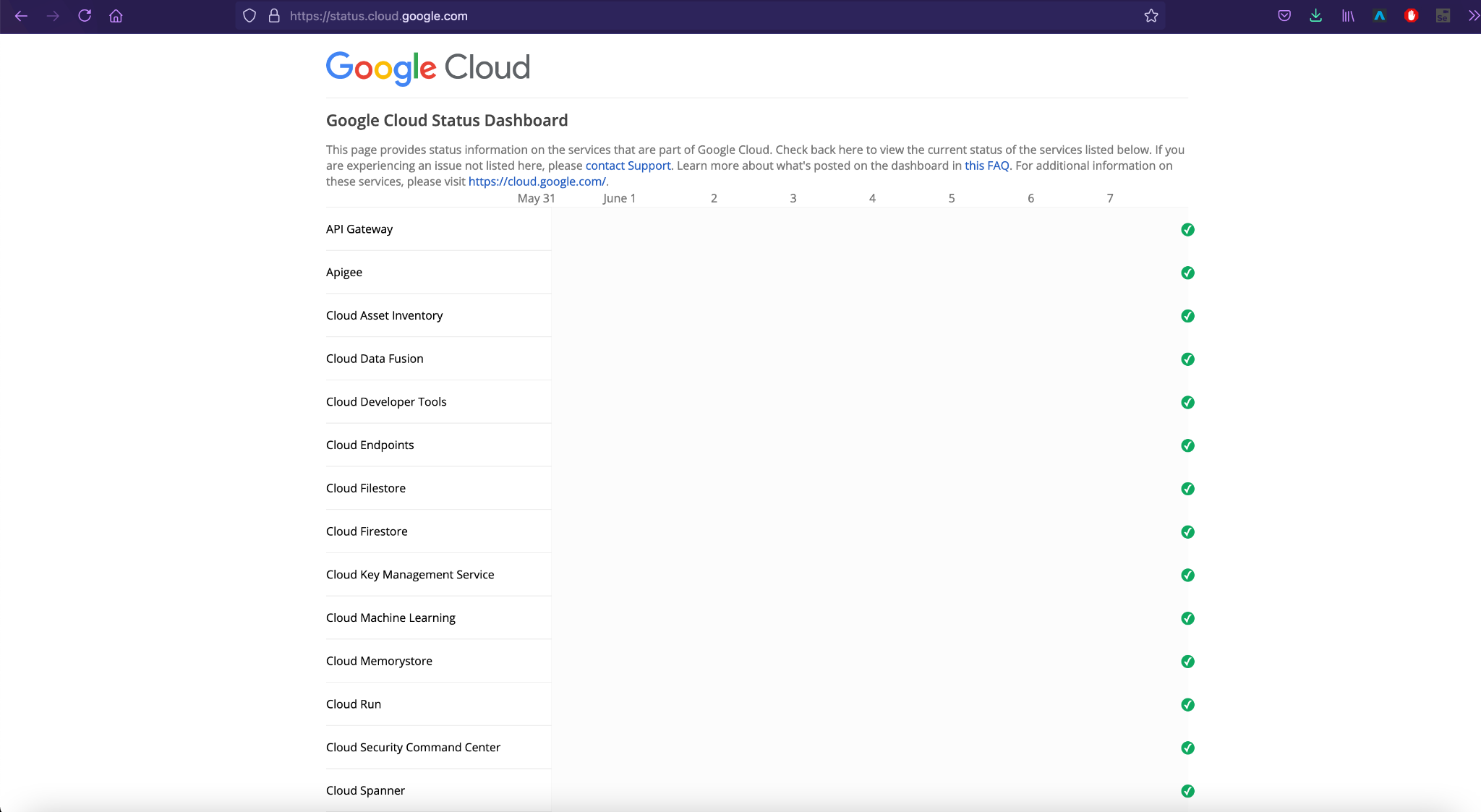
É uma plataforma da Amazon, com o propósito de ver o estado dos serviços proporcionados por eles e que estão distribuídos por diferentes localizações, desde a América, tanto do Sul como do Norte, passando pela Europa, África até o Oriente Médio. Então, o problema acaba por ser que não envia notificações de serviços específicos para os seus clientes e o histórico dos serviços encontra-se no fundo da página principal ao invés de ser numa página em separado.



**obs: colocar uma legenda “Fig. 4 - AWS”**

# **3.1.2 Google Cloud Status Dashboard**

É uma plataforma proporcionada pelo Google, que também fornece informação relativamente ao estado dos seus serviços. Além disso, possui um histórico de falhas onde especifica o tempo em que o serviço estava indisponível e o mostra de uma forma muito mais prazerosa ao olho do utilizador, dada a sua simplicidade e brevidade, e isto deve-se também ao facto de não possuir uma página inicial muito extensa. Apesar disso, esta plataforma não providencia uma estimativa de tempo para a resolução de um problema e não é possível a subscrição de determinados serviços para a recepção de notificações.



**obs: colocar uma legenda “Fig. 5 - Google Cloud”**

# **3.2 Tecnologia Utilizada**

Nesta seção, irá se proceder a uma explicação da tecnologia utilizada para a criação e desenvolvimento deste projeto.

# **3.2.1 REACT JS**

\*\*\* escrever acerca da utilização do ReactJS \*\*\*

# **3.2.2 .NET CORE & C#**

\*\*\* escrever acerca da utilização de .NET Core \*\*\*

# **3.2.3 Persistence: SQL**

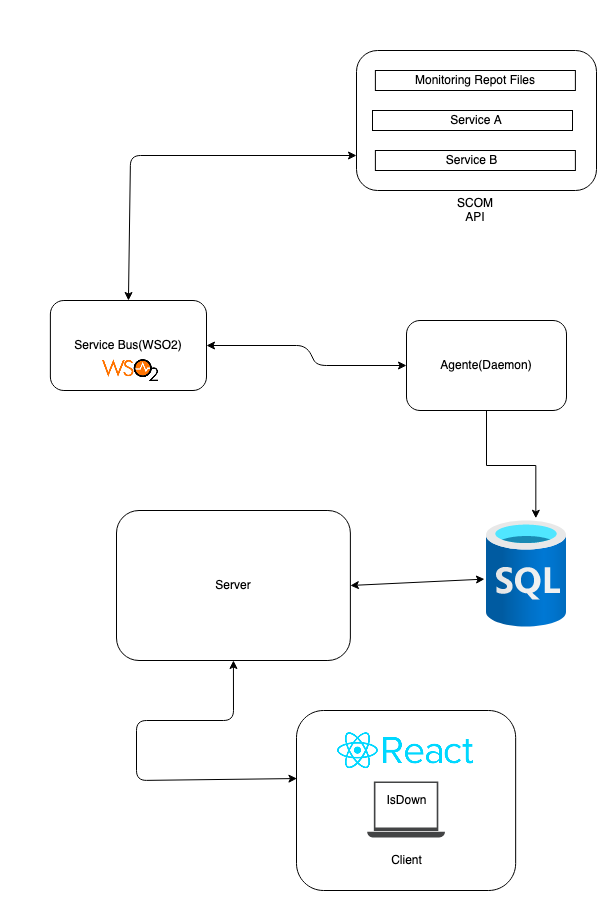
\*\*\* escrever acerca da criação da BD em SQL\*\*\*

# **3.2.1 Modelo Tecnológico**

O diagrama a seguir, representa a arquitetura do nosso sistema, assim como as tecnologias utilizadas em cada camada.

Para a camada do cliente, a aplicação web foi desenvolvida em **ReactJS** que irá comunicar com o Backend, indo recolher os dados na BD em **SQL** da **Microsoft Azure** que foram guardados pelo Agente, ou seja, pelo Daemon, recebendo assim mensagens a partir das filas do **Service Bus WSO2** com os dados recolhidos da **API SCOM**.

**Obs: Falta indicar o .NET CORE e C# no diagrama!**



**obs: colocar uma legenda “Fig. 6 - Modelo Tecnológico"**

# **3.2.2 Ferramentas: Discord, Google Docs and Microsoft Teams**

Em relação às ferramentas de colaboração entre a equipa, temos em primeiro, o **Discord** que inicialmente foi uma plataforma criada para comunidades de jogos, hoje em dia, acaba por ser para muito mais além disso e é bastante útil para outras comunidades e para a comunicação contínua entre elementos de um grupo, contendo a partilha de documentos, a criação de canais tanto para texto ou para voz, que acaba por ajudar aos elementos do grupo a se distribuírem se estiverem a trabalhar em partes diferentes do projeto. Além disso, inclui chamadas, videochamadas incluindo partilha de ecrã e por fim a integração de serviços externos como o **Github** ou **Gitlab**. E por isso, foi a nossa escolha como plataforma principal de comunicação.

Em seguida, a plataforma de escolha para a partilha de documentos tais como, as apresentações em Power Point e o Relatório do trabalho, utilizou-se o **Google Docs** pela familiaridade que os elementos do grupo têm e por ser uma plataforma que permite a edição dos documentos em tempo real, permitindo no final guardar os documentos em formato Word ou até PDF. Além disso, também é uma plataforma que funciona de forma síncrona e assíncrona.

Por fim, o **Microsoft Teams** foi a plataforma escolhida para as reuniões semanais com o nosso orientador, que suporta a partilha de documentos, a colaboração com um bate-papo, videochamadas e armazenamento de arquivos.

# **3.2.3 Ferramentas: Microsoft Azure and Github**

A seguir, no que diz respeito às ferramentas utilizadas para guardar a parte de construção de código, publicação do mesmo e criação de serviços, utilizamos tanto a **Microsoft Azure** como o **Github.** Em primeiro lugar, a **Microsoft Azure** é uma plataforma destinada à criação de aplicações e serviços que baseiam o seu funcionamento em conceitos de computação em nuvem. Além disso, serviu como plataforma para a publicação do nosso projeto e para a criação da **BD** em SQL, do Daemon como dispositivo externo e do Service Bus para a troca de mensagens.

Em segundo, temos o **Github** que é uma plataforma que serviu para hospedar o código desenvolvido no projeto e para controle de versões do mesmo.

# **Implementação**

# **4.1 Services**

\*\*\* escrever acerca da tabela services\*\*\*

# **4.2 DNS**

\*\*\* escrever acerca da tabela de DNS \*\*\*

# **4.3 Admin**

\*\*\* escrever acerca das funções do Admin\*\*\*

# **4.2 Notificações**

\*\*\* escrever acerca do serviço das Notificações\*\*\*

# **Conclusão**

# **5.1 Sumário**

\*\*\* overall \*\*\*

# **5.2 Resultados**

\*\*\* escrever acerca dos principais resultados\*\*\*

# **5.3 Trabalho Futuro**

\*\*\* escrever acerca das implementações futuras\*\*\*

# **Referências**

[1] Youtube, <https://www.youtube.com/>

[2] StackOverflow, <https://stackoverflow.com/>

[3] Microsoft, <https://docs.microsoft.com/pt-pt/>

[4] Microsoft Azure, <https://docs.microsoft.com/pt-pt/>

[5] MDN Web Docs, <https://developer.mozilla.org/en-US/>

[6] DZone, <https://dzone.com/>

[7] TutorialsPoint, <https://www.tutorialspoint.com/index.htm>

[8] StackExchange, <https://stackexchange.com/>

[9] Morioh, <https://morioh.com/explore?next=%2F>

[10] CodeSandBox, <https://morioh.com/explore?next=%2F>

[11] SQLHack, <https://www.sqlshack.com/>

[12] DigitalOcean, <https://www.digitalocean.com/community>