IPG Politécnico da Guarda
Escola Superior de Tecnologia e Gestão

RELATÓRIO DE ESTÁGIO

Curso Técnico Superior Profissional
em Desenvolvimento de Aplicações Informáticas

José Miguel Murça Soares

julho | 2018

RELATÓRIO DE ESTÁGIO

CURSO TÉCNICO SUPERIOR PROFISSIONAL

DESENVOLVIMENTO DE APLICAÇÕES INFORMÁTICAS

JOSÉ MIGUEL MURÇA SOARES

JULHO 2018

ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Instituto Politécnico da Guarda

RELATÓRIO DE ESTÁGIO

JOSÉ MIGUEL MURÇA SOARES

RELATÓRIO PARA A OBTENÇÃO DO DIPLOMA DE

TÉCNICO SUPERIOR PROFISSIONAL EM DESENVOLVIMENTO

DE APLICAÇÕES INFORMÁTICAS

07/2018

ÍNDICE

ÍNDICE FIGURAS

Ficha de Identificação

Estagiário: José Miguel Murça Soares

Nº de aluno: 1012113

Curso: Técnico Superior Profissional em Desenvolvimento de Aplicações Informáticas

Instituição: Instituto Politécnico da Guarda

Morada: Avenida Dr. Francisco Sá Carneiro, n.º 50

Localidade: 6300-559 Guarda

Telefone: 271 220 100 **Fax:** 271 222 690

Data de início do estágio:19 /02/2017

Data de fim do estágio: 14/07/207

Duração do estágio: 750 horas

Docente Orientador: Pedro Manuel Pinto Teixeira

Grau Académico do Orientador: Mestre em Computação Móvel

Supervisor na Instituição: Diogo Aleixo

Cargo Supervisor: Team Líder GoContact, Guarda

Resumo

Este relatório apresenta o estágio curricular para a obtenção do grau de Técnico Superior Profissional em Desenvolvimento de Aplicações Informáticas e que inclui o desenvolvimento de uma aplicação Web. O estágio decorreu na empresa GoContact Guarda. Como objetivo principal, colocava-se o desenvolvimento de uma aplicação React e Node, uma Aplicação Web Cross-Platform tendo conhecimento prévio que se trata de uma importante plataforma utilizada por milhares de profissionais, e que tem as suas necessidades de reajustes.Intitulada de GoJira, esta aplicação comunica com a API de um projeto JIRA, que permite o tratamento dos dados para proporcionar uma Aplicação Responsiva de fácil utilização para a monitorização de qualquer projeto JIRA. Ao longo deste percurso, foi necessário a aquisição de formações especificas, recorrer a uma metodologia de desenvolvimento, delinear a funcionalidade do projeto de forma a determinar-se a arquitetura na qual o software seria desenvolvido.

Palavras-chave: Arquitetura Software, React, Node, Jira, Metodologia, Cross-Platform, API

Agradecimentos

Em primeiro lugar gostaria de deixar o meu reconhecimento ao Instituto Politécnico da Guarda e aos professores do TeSP Desenvolvimento de Aplicações informáticas pelo apoio e a partilha de conhecimentos.

Agradeço á empresa GoContact por me ter permitido realizar o estágio na empresa durante estes últimos meses e pela formação que me foi transmitida ao longo de todo o estágio.

Para a realização deste trabalho contámos com importantes apoios e incentivos sem os quais não teríamos tornado uma realidade, a quem nos cabe agradecer.

Ao Chefe de equipa da GoContact Guarda, Diogo Aleixo, agradeço a oportunidade e o privilégio de realizar este trabalho sob a sua supervisão, agradecendo a sua partilha de saberes, profícuos comentários e sugestões. Também a toda a sua equipa que me recebeu de braços abertos, mostrando-se disponíveis para ajudar no que fosse necessário.

Ao Professor Pedro Pinto, orientador deste relatório, pelos seus conselhos, disponibilidade e apoio pois estes foram uma base fundamental para a realização deste relatório.

E por fim, tendo consciência que sozinho nada disto teria sido possível, um agradecimento especial aos meus pais e irmã, pelo seu apoio incondicional, carinho e paciência demonstrados.

CAPÍTULO I- INTRODUÇÃO

O presente relatório tem por finalidade registrar a prática desenvolvida no Estágio, para a obtenção do grau de Técnico Superior Profissional em Desenvolvimento de Aplicações Informáticas do Instituto Politécnico da Guarda (IPG), no período de 1 de março de 2018 até 16 de Julho de 2018. Foi realizado em forma de Projeto intitulado "GoJira", com o desenvolvimento de uma plataforma que permitisse uma fácil monitorização de um Projeto JIRA, visando colocar em prática os conhecimentos adquiridos ao longo do curso Técnico Superior Profissional em Desenvolvimento de Aplicações Informáticas, com o intuito de me integrar no desenvolvimento de aplicações informáticas no mundo da programação.

O estágio decorreu em dias uteis entre as 9 e as 18 horas, na empresa GoContact Guarda, sediada no Polo da ESTG do IPG, onde contei com o apoio de todos os profissionais que lá exercem funções, que trabalham diretamente com o desenvolvimento de aplicações informáticas, tendo sido estes, que após varias utilizações, da plataforma *JIRA*, verbalizaram a necessidade de desenvolver uma aplicação para um projeto JIRA, surgindo então todo o meu projeto ao longo destes últimos meses, onde dediquei o meu tempo, aprofundei conhecimentos para poder criar GoJira, de forma a conseguir atingir os meus objetivos.

1.1 Objetivos do Estágio

O estágio teve como objetivo principal o desenvolvimento de uma aplicação React e Node, para tal foi necessário obter conhecimentos nas tecnologias e ferramentas utilizadas atualmente, dividindo-se em três objetivos principais.

1. **Adotar Metodologia de Desenvolvimento**

Adquirir conhecimento, utilizando as metodologias de desenvolvimento, de forma a compreender o impacto que estas têm no decorrer do desenvolvimento de qualquer tipo de software.

2. **Conceito de Arquitetura de Software**

Entender o conceito de Arquitetura de Software, quais os intervenientes, qual a importância e os cuidados a ter em conta na decisão de uma arquitetura para o projeto.

3. **Desenvolvimento de uma Aplicação**

Desenvolver uma Aplicação Web *Cross-Platform*, utilizando os conhecimentos obtidos nos objetivos anteriores.

Através da concretização deste estágio profissional estive em contato com a realidade da programação informática, momento fundamental para a minha formação enquanto futuro *developer*. Por meio da observação visualizei as competências que os profissionais desenvolvem, como também diferentes estratégias a utilizar. No decorrer deste estágio presenciei uma prática pedagógica supervisionada que me auxiliou na reflexão do projeto que estava a desenvolver. Assim sendo, através desta prática pude refletir sobre o mesmo, consciencializando-me das minhas capacidades, conhecimentos, e quais as melhores metodologias e estratégias a adotar no futuro.

1.2 Estrutura do Relatório

O presente relatório está dividido em cinco capítulos. Sendo o primeiro uma introdução ao trabalho desenvolvido e os seus objetivos. O Segundo capítulo visa explicitar o contexto do projeto fazendo uma introdução à plataforma JIRA e à Metodologia *Scrum*, bem como à formação fornecida o longo do estágio. No terceiro capítulo será feita uma breve introdução ás principais ferramentas utilizadas. No quarto capítulo é definida a arquitetura do projeto e o seu desenvolvimento, seguindo-se de uma breve demonstração.

Por fim, na Reflexão Final irei refletir sobre os objetivos alcançados durante o período de estágio, bem como a contribuição do mesmo, tanto a nível profissional como pessoal e as minhas limitações.

1.3 GoContact /Local de Estágio

O estágio decorreu na empresa GoContact Guarda, atualmente com quatro polos a nível nacional, nomeadamente Aveiro, Guarda, Porto e Lisboa, a GoContact têm vindo a revolucionar o mundo dos *Contact Centers*, oferecendo soluções integradas de última geração, oferecendo aos seus clientes (*Figura 1*) uma plataforma que fornece todas as funcionalidades de um *Contact Center* inovador.

O polo sediado na Guarda, conta com uma equipa de desenvolvimento de 5 profissionais na área de desenvolvimento de software.

Atualmente distinguida com o Prémio Gold na categoria Tecnologia, no âmbito dos APCC (Associação Portuguesa de Contact Centers) Best Awards 2018.

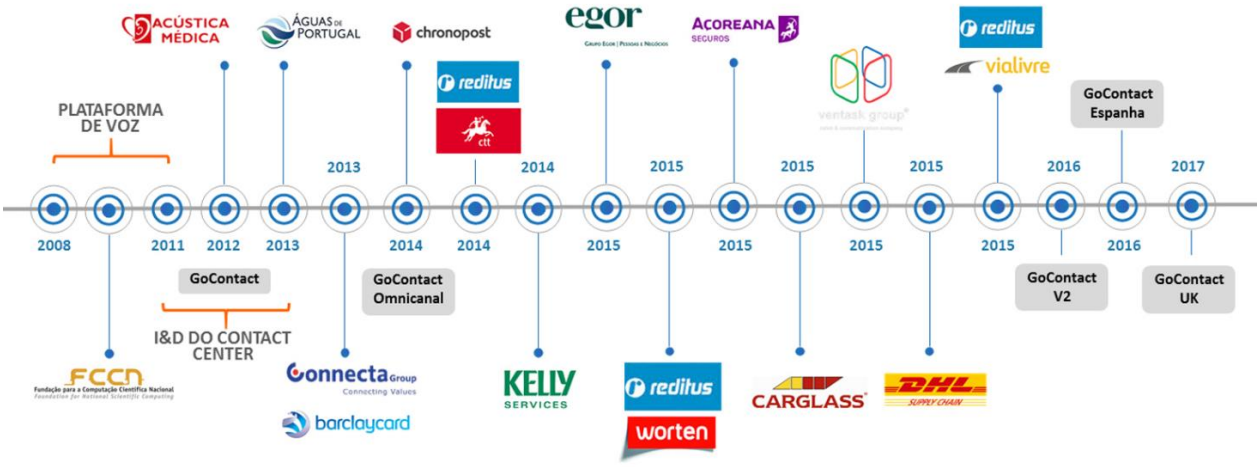

Figura 1 - História GoContact

1.4 Software JIRA e a sua importância

JIRA é um do software comercial desenvolvido pela empresa Australiana *Atlassian*, sendo uma ferramenta que permite a criação e monitorização de tarefas bem como o acompanhamento de projetos garantindo a gestão de todas as suas atividades num único lugar.

Desenvolvido e Lançado em 2002, JIRA é uma plataforma popular entre os Engenheiros de software, milhares de utilizadores utilizam a plataforma diariamente para planear, rastrear e lançar os seus produtos.

Um Projeto JIRA é um agrupamento de tarefas e é definido de acordo com a organização e as necessidades de cada, são exemplos de projetos JIRA: um projeto de desenvolvimento de software, uma campanha de marketing.

Através desta plataforma (*Figura 2*) todos os integrantes da equipa têm conhecimento de quais as suas Tarefas, prioridades e prazos, uma vez que contêm ferramentas poderosas e funcionalidades para cada fase de desenvolvimento do software.

No entanto e como estamos em constante mudança, com o objetivo de melhorar as nossas práticas é em volta desta importante plataforma que se desenvolve todo o meu projeto.

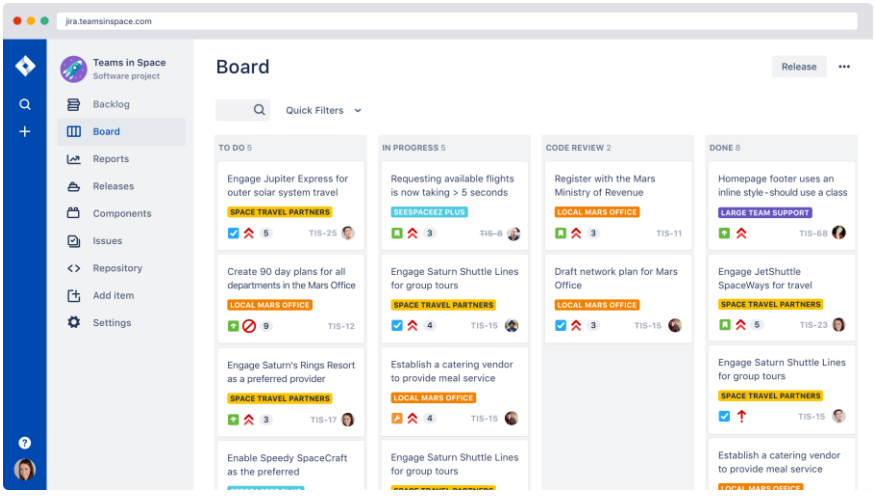

Figura 2 - Dashboard Atlassian Jira Software

CAPÍTULO II – CONTEXTO DO PROJETO

2.1 Metodologia

Metodologia é uma palavra derivada de "*método*", do Latim "*methodus*" que significa "*caminho ou a via para a realização de algo*". Método é o processo para se atingir um determinado fim ou para se chegar ao conhecimento, e metodologia é o campo em que se estuda os melhores métodos praticados em determinada área para a produção do conhecimento, sendo que cada área possui uma metodologia própria.

Adquirir uma metodologia de desenvolvimento apropriada é fundamental e deverá ser o primeiro passo a ser tomado no desenvolvimento de qualquer projeto, dividindo-o em várias partes com determinadas etapas tornando assim mais eficiente o planeamento e a gestão do mesmo.

Em muitos projetos o insucesso é frequente derivado ao facto de não seguirem uma metodologia apropriada ou até mesmo por ausência de uma, prazos não são cumpridos, orçamentos que são ultrapassados, o que leva a uma insatisfação por parte do cliente.

Estas metodologias não só tornam mais fácil o desenvolvimento de um projeto, mas também são uma mais valia para quem as adota, tornando-se um desenvolvedor sistemático ensinando-o a seguir um modelo de desenvolvimento, resultando em um trabalho não só mais eficiente, mas também com maior qualidade.

E como forma de cumprir com os objetivos a que me propus inicialmente para este estágio tive como base a metodologia *Ágile* utilizando a técnica *Scrum* à qual faço uma breve descrição seguidamente, bem como a formação complementar que me foi cedida [1].

2.2 Ágile

"Intelligence is the ability to adapt to change." - Stephen Hawking

Agile refere-se a um conjunto de métodos e práticas baseadas nos valores e princípio expressos no Manifesto Ágil (*Agile Manifest*)", o que inclui por exemplo a colaboração, auto-organização, e equipas interdisciplinares.

Ágile é a tomada de decisão baseada nos valores e princípios que a equipa decidiu em seguir, como se confere no manifesto (*Manifesto for Agile Software Development*) [1]

2.2.2 Os Princípios *Ágile*:

- A prioridade é, desde as primeiras etapas do projeto, satisfazer o cliente através da entrega rápida e contínua de software com valor.
- Aceitar alterações de requisitos, mesmo numa fase tardia do ciclo de desenvolvimento. Os processos ágeis potenciam a mudança em benefício da vantagem competitiva do cliente.
- Fornecer frequentemente software funcional. Os períodos de entrega devem ser de poucas semanas a poucos meses, dando preferência a períodos mais curtos.
- O cliente e a equipa de desenvolvimento devem trabalhar juntos, diariamente, durante o decorrer do projeto.
- Desenvolver projetos com base em indivíduos motivados, dando-lhes o ambiente e o apoio de que necessitam, confiando que irão cumprir os objetivos.
- O método mais eficiente e eficaz de passar informação para e dentro de uma equipa de desenvolvimento é através de conversa pessoal e direta.
- A principal medida de progresso é a entrega de software funcional.
- Os processos ágeis promovem o desenvolvimento sustentável. Os promotores, a equipa e os utilizadores deverão ser capazes de manter, indefinidamente, um ritmo constante.
- A atenção permanente à excelência técnica e um bom desenho da solução aumentam a agilidade.

- Simplicidade – a arte de maximizar a quantidade de trabalho que não é feito – é essencial.
- As melhores arquiteturas, requisitos e desenhos surgem de equipas auto-organizadas.

2.3 Metodologia *Scrum*:

Scrum é uma estrutura metodológica (Técnica *Ágile*) que pode ser aplicado a qualquer desenvolvimento de produto ou projeto, é uma técnica que consiste em uma determinada organização. [1]

2.3.1 Scrum Cargos e Responsabilidades

No *Scrum* existem 3 cargos específicos [1] :

- **Product Owner** - É o dono e visionário do produto, responsável por o *Product Backlog* (Conjunto de todas as funcionalidades de um projeto) e também responsável pelas despesas. Valida o trabalho das equipas no final de cada Sprint, aceitando-o ou rejeitando.
- **Scrum Master** - É o Responsável por manter a Equipa de desenvolvimento produtiva, certificando-se que a equipa tem tudo o que é necessário para o desenvolvimento do projeto.
    - **Developer Team** - Equipa composta de 5 a 9 elementos, são os responsáveis por entregar o trabalho definido em cada *sprint*.

2.3.2 *Scrum* – Meetings

Scrum Meetings são diversas práticas utilizadas por equipas que seguem os princípios e valores *Ágile* [1] :

- **Sprint Planning** - Reunião de equipa para o planeamento do que será completado no próximo Sprint. (Duração aproximadamente de 1 hora por cada semana do Sprint)
- **Daily Stand-up** - Pequena reunião de no máximo 15 minutos para a equipa se manter em sincronização, nomeadamente o que cada um fez no dia anterior e o que irá fazer nesse mesmo dia, também (se for caso) devem indicar se algo está a bloquear o seu trabalho.
- **Sprint Review** - Reunião onde a equipa mostra o que foi completado nesse Sprint e receber o *feedback* do *product owner*.
- **Sprint Planning** - Rever o que correu melhor e pior para poder melhor em Sprints futuros. O objetivo é melhorar a cultura dos integrantes da equipa e os seus processos. A sprint *retrospective* tem por norma 60 minutos

2.3.3 User Stories & Product Backlog

Todas as funcionalidades a serem implementadas no projeto são denominadas de *User Stories*, e ao conjunto de todas estas *User Stories* chama-mos de *Product Backlog* (*Figura 3*) .

Os *Sprints* é o agrupamento de *User Stories* que residem no *Product Backlog*, estes *sprints* tem uma duração compreendida entre 2 a 30 dias, e é importante o resultado do mesmo ser um produto funcional [1].

Figura 3 - - User Story & Product Backlog

2.4 Formação Complementar

Entende-se como definição de formação "o ato ou efeito de formar ou de se formar", formação é um conjunto de conhecimentos específicos que são ministrados ou adquiridos, por forma a adquirir ou actualizar conhecimentos profissionais ou relacionados com uma atividade.

Ao longo dos meses em que desenvolvi o meu estágio, e para que fosse possível o desenvolvimento desta aplicação, para além de todos os conhecimentos adquiridos ao longo destes últimos anos durante a formação profissional, foram-me fornecidas formações online, as quais fundamentais para o sucesso final, entre elas destaco:

2.4.1 The Web Developer Bootcamp – Udemy

Desde o mais simples até aos mais avançados e complexos tópicos "*The Web Developer Bootcamp"* introduz-nos ás Tecnologias de desenvolvimento Web, um curso *Full Stack* completo para iniciantes abrangendo ferramentas e tecnologias que tiveram um impacto significativo durante o desenrolar deste projeto, tais como: Javascript, HTML, CSS, Bootstrap, jQuery, Node, NPM, ExpressJS, REST… [2]

Duração do Curso: 43 Horas

2.4.2 The Complete Node.js Developer Course (2nd Edition) – Udemy

Este curso, capacita-nos desde a simples instalação e introdução Node até ao desenvolvimento de uma aplicação Chat em tempo real, através de pequenos exercícios que consistem na criação de pequenas aplicações, no final de cada capitulo utilizando todo o conhecimento obtido nas mesmas, o que resultou numa maior facilidade de desenvolvimento em todo este projeto. e na aquisição de conhecimentos para projetos futuros. [4]

Duração do Curso: 26.5 Horas

CAPÍTULO III - FERRAMENTAS UTILIZADAS

3.1 React

React ou *ReactJS* (*Figura* 4) é uma biblioteca Javascript *Open-Source* especificamente utilizada no desenvolvimento de interfaces (*Frontend*), responsável pela camada visível de uma aplicação Web ou móvel.

Desenvolvida por Jordan Walke, engenheiro de software da empresa Facebook, inicialmente implementada no Facebook em 2011 e mais tarde no Instagram 2012, é atualmente das bibliotecas mais populares no desenvolvimento de interfaces, *React* baseia-se em componentes, estes permitem a divisão da interface em partes independentes e reutilizáveis.

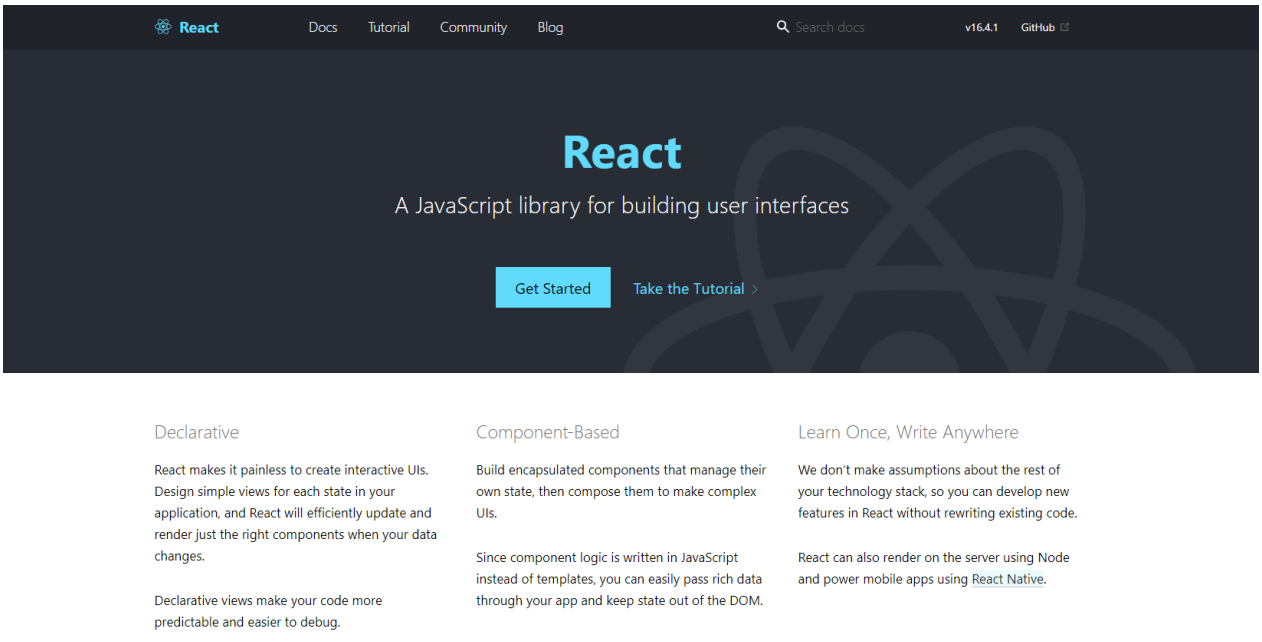

Figura 4 – Homepage React

3.2 Node

NodeJS ou Node (*Figura 5*) é um ambiente de desenvolvimento *Open-Source* baseado em Javascript frequentemente utilizada no desenvolvimento de serviços *backend*, sendo estes os responsáveis para o funcionamento da aplicação cliente. Conhecido não só pela sua rapidez e alta escalabilidade, mas também por possuir inúmeras bibliotecas (*packages*) *Open-source*.

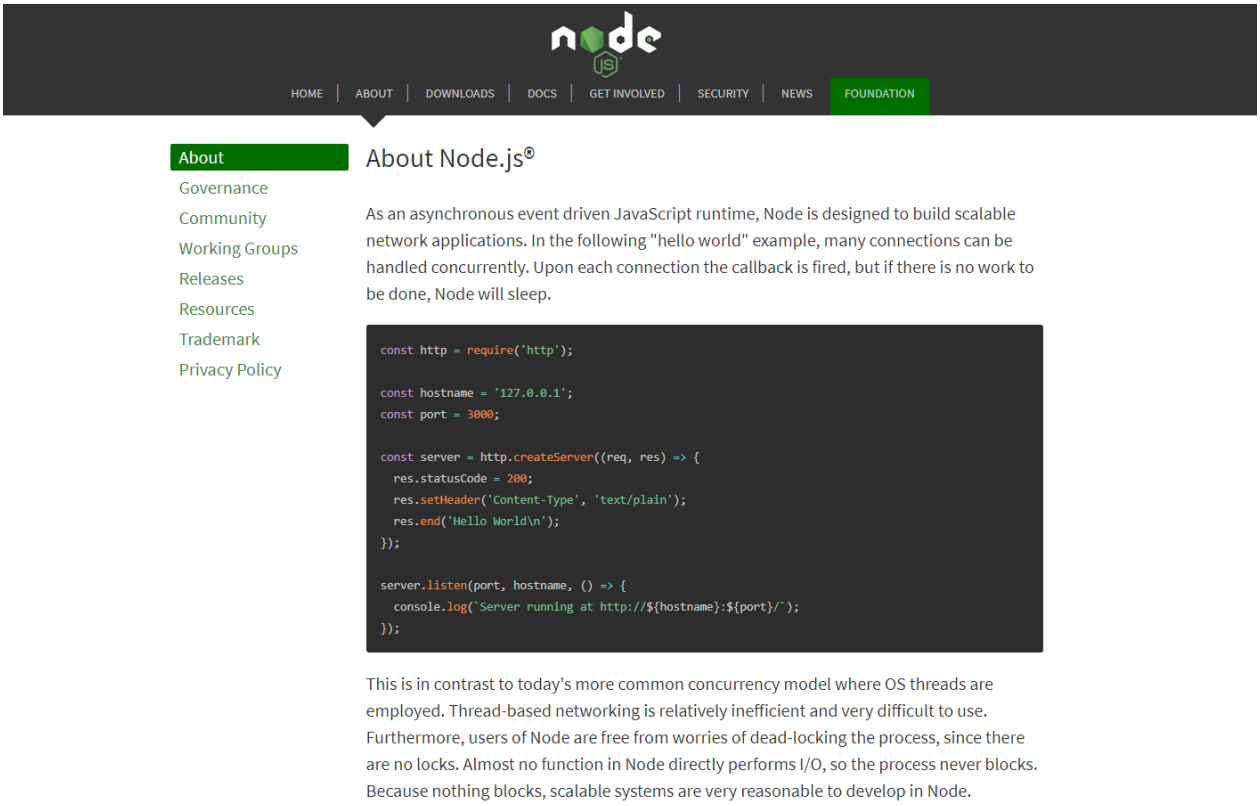

Figura 5 – Homepage Node

3.3 NPM (Node Package Manager)

NPM (*Figura 6*) é uma plataforma que conta com mais de 600,000 *packages*, estes são partilhados por toda a comunidade de Javascript *developers*. São "pedaços" ou "blocos" de código funcional, com um determinado objetivo que podemos utilizar nos nossos projetos. É também possível criar e partilhar packages com a comunidade.

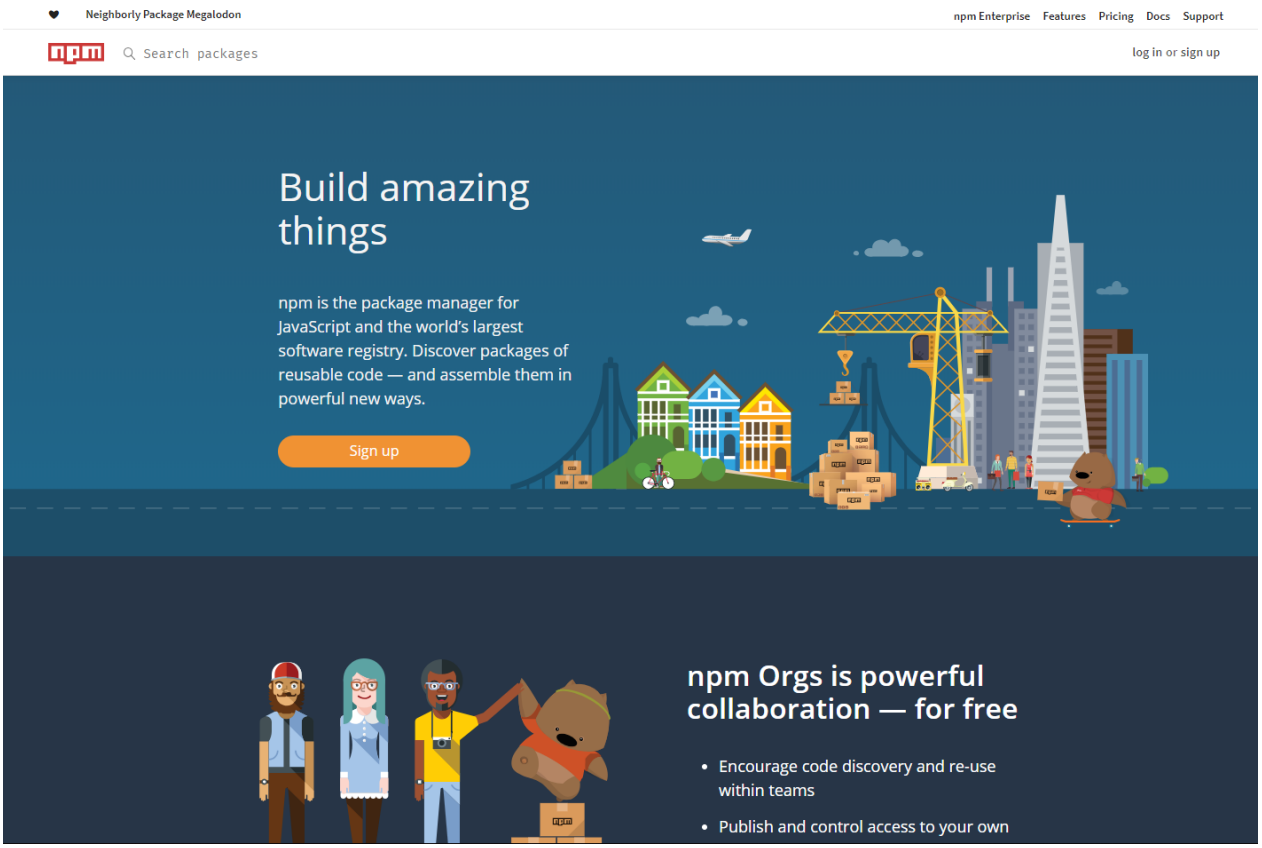

Figura 6 – Homepage NPM (Node Package Manager)

3.4 Postman

É uma poderosa ferramenta indispensável no desenvolvimento Web. Iniciada em 2012 *Postman* tem trazido todas as funcionalidades necessárias para o desenvolvimento e teste das *API´s*.

Postman (Figura 7) permite a documentação dos *Endpoints (*Endereço URL que permite aceder a um serviço fornecido pelo *backend)* através de coleções sendo possível partilha-las com toda a equipa.

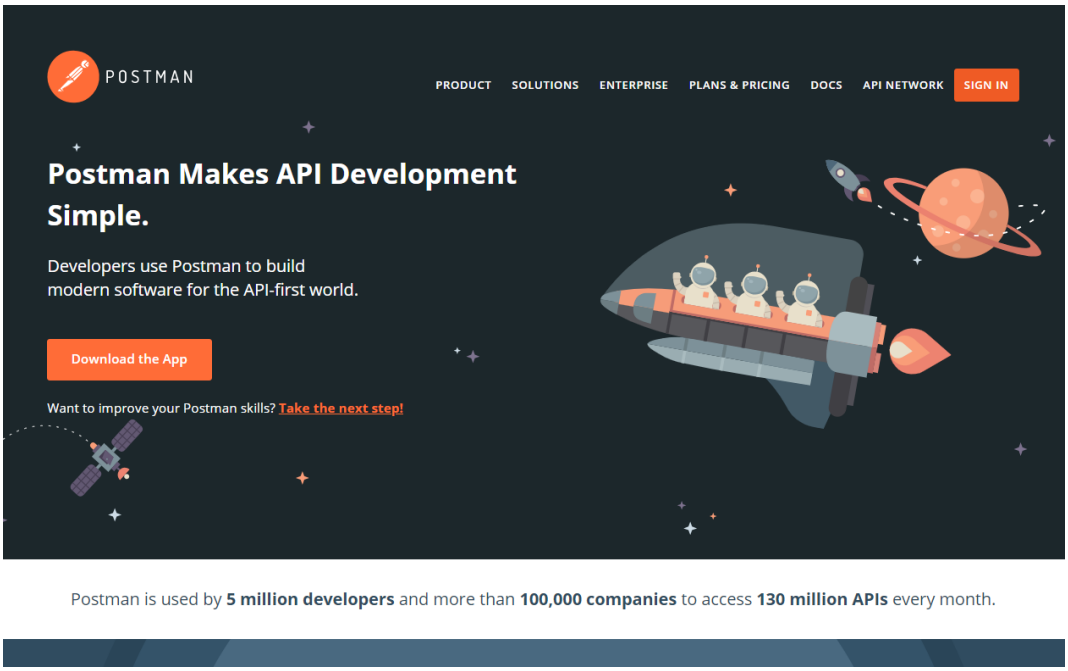

Figura 7 – Homepage Postman

3.5 ESlint

Eslint (*Figura 8*) é uma Ferramenta *Open-source* utilizada na validação de *Javascript* / *JSX* (Extensão de sintaxe *Javascript* utilizada em *React*), é executado dentro do editor de código detetando erros de sintaxe ou referencias. Não só deteta estes erros como também melhora a Qualidade do código em um determinado projeto que por sua vez vai melhorar a qualidade de código produzida por os seus utilizadores.

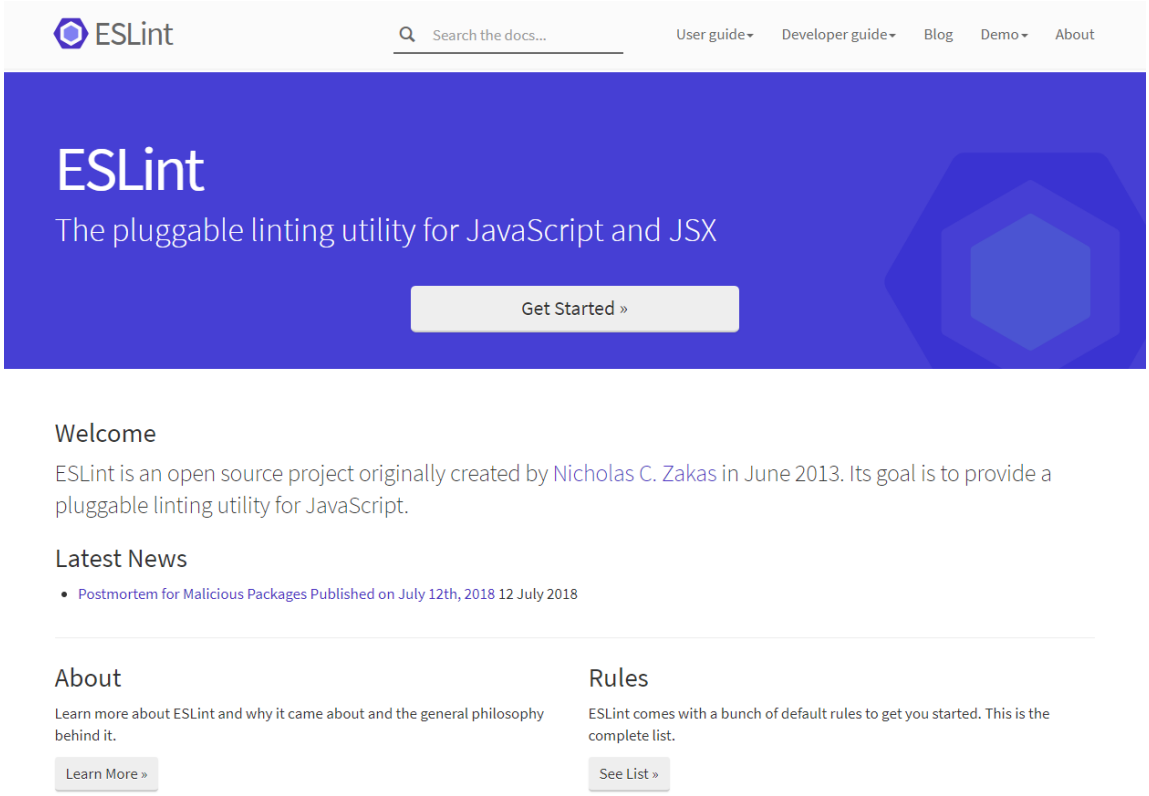

Figura 8 – Homepage Eslint

3.6 Git

Git (*Figura 9*) é um (VCS - version control system) Sistema de controlo de versão, que permite o rastreamento das alterações feitas ao nosso código e simplifica o processo de partilha de código com uma equipa.

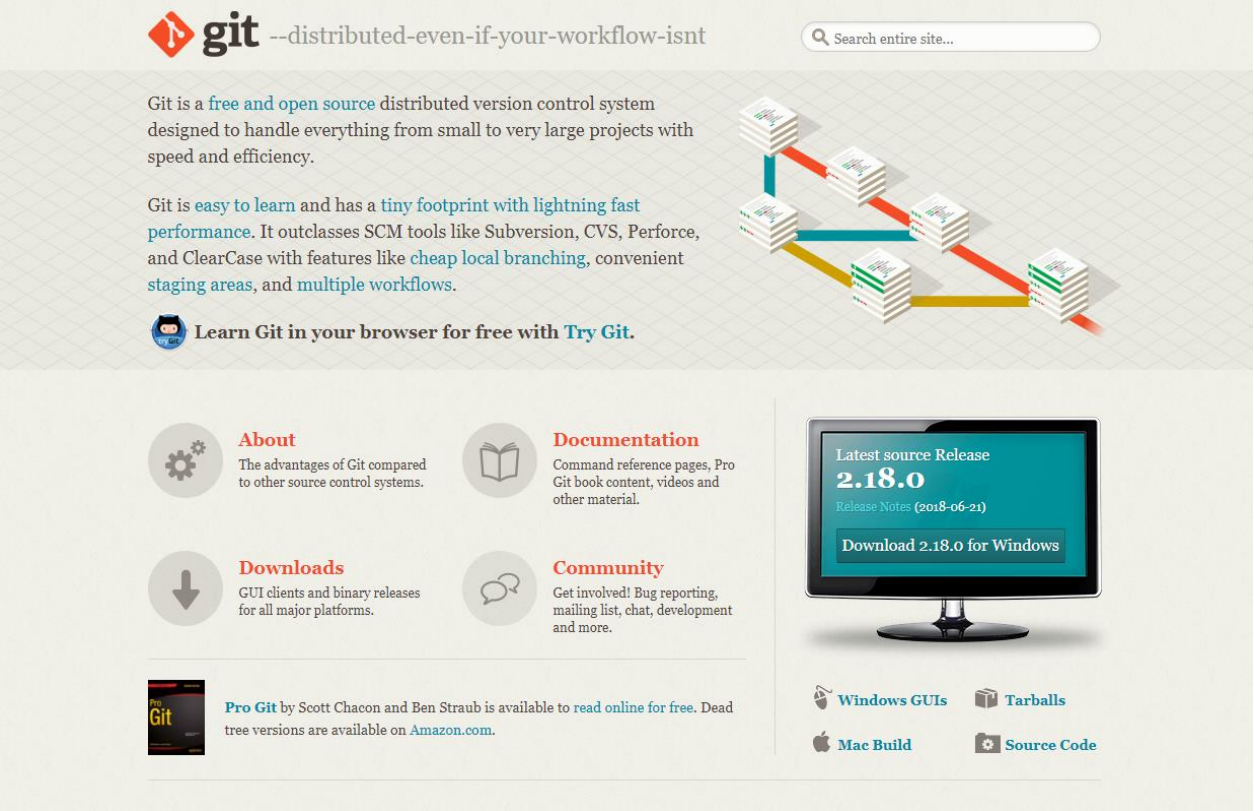

Figura 9 - Homepage git

3.7 GitHub

O GitHub (*Figura 10*) pode ser considerado uma "rede social" ou comunidade de programadores, onde podemos adquirir conhecimentos, partilhar e trabalhar juntamente com outros *developers* na construção de software.

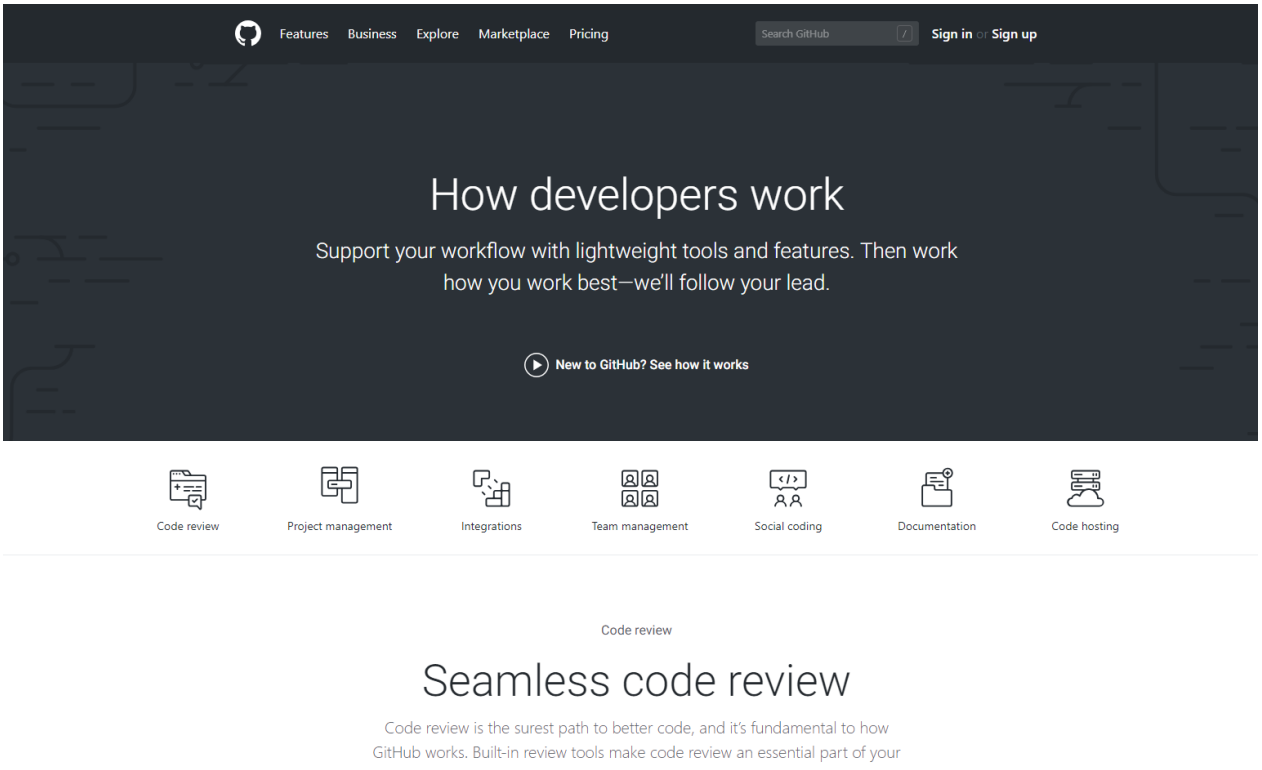

Figura 10 - Homepage GitHub

3.8 Stack Overflow

Stack Overflow (*Figura 11*) têm um papel fundamental no Mundo da programação, este serve como plataforma para que os utilizadores façam ou respondam a questões de desenvolvimento de software no caso de as mesmas ou questões semelhantes ainda não tenham sido questionadas.

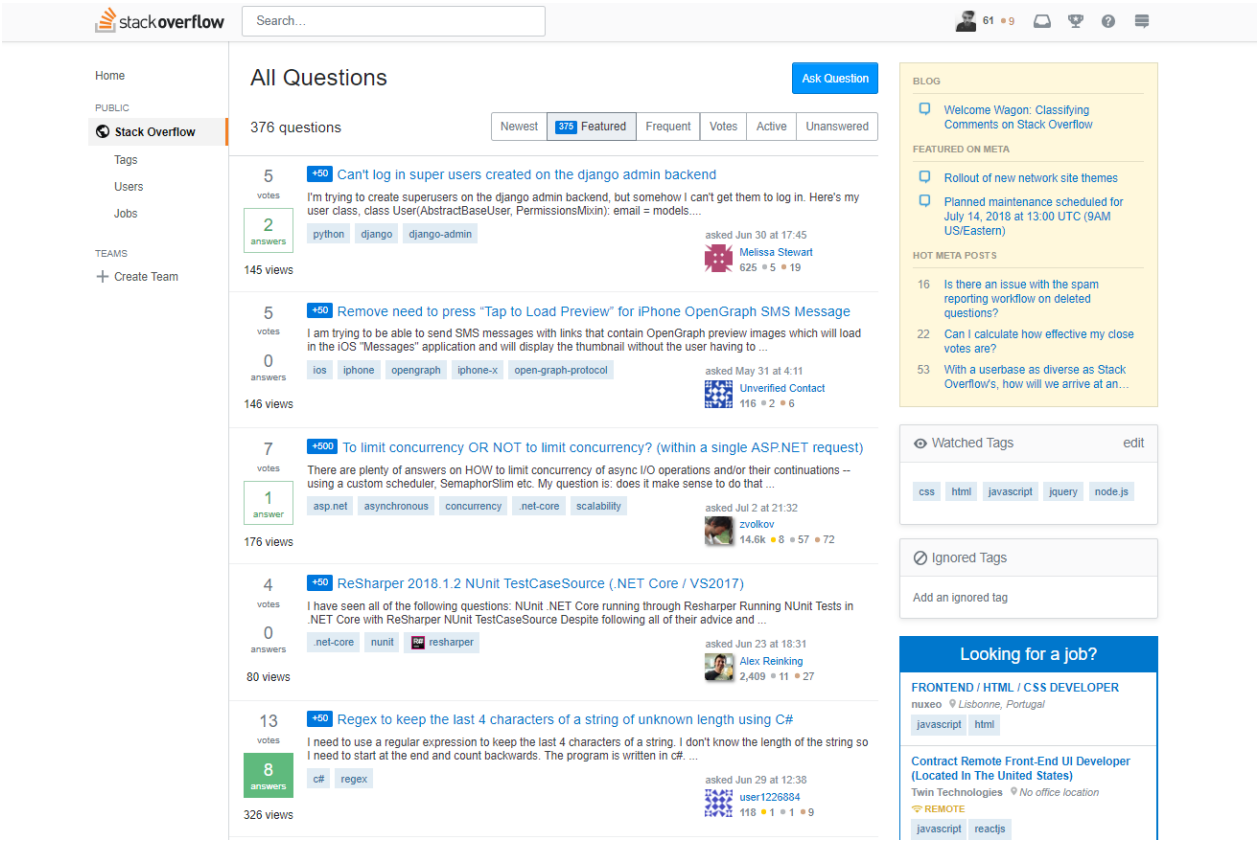

Figura 11 – Dahboard Stack Overflow

CAPÍTULO IV – Aplicação Desenvolvida

Neste capitulo vou tentar descrever o trabalho desenvolvido ao longo deste estágio, de forma a dar a conhecer a importância que este tem na prática de monitorização de um projeto, e que consiste no desenvolvimento de uma Aplicação Web, para a qual foi utilizando a metodologia de desenvolvimento *Scrum* e baseado numa Arquitetura de Software, arquitetura esta que irá ser analisada e definida após a discussão das funcionalidades.

4.1 GoJira

GoJira (*Figura 12*) é o trabalho resultante de quatro meses de estágio, uma aplicação Web *Cross-platform* em tempo real que permite a fácil monitorização de um projeto JIRA. GoJira permite obter todas as informações essenciais de um projeto, como listar *sprints* e os seus estados, obter tarefas de um determinado *sprint*, obter informações detalhadas sobre essas tarefas, bem como um uma *timeline* para saber o seu percurso completo no projeto.

Figura 12 - Logótipo GoJira

4.2 Funcionalidades da Aplicação

No processo de desenvolvimento de sistemas, podemos definir funcionalidade como um comportamento ou uma ação para a qual possa ser visualizado um início e um fim, algo que possa ser executado.

E como forma de cumprir o objetivo a que me propus, na meta de uma aplicação, numa fase inicial, foram decididas as funcionalidades (*User Stories*) que a aplicação deveria ter, constituindo desta forma o *Product Backlog* (Conjunto de *User Stories*) da Aplicação (*Figura 13*).

Passo então a enumerar as Funcionalidades (Requisitos) Funcionais:

- Quais os Sprints e os seus estados do Projeto
- Tarefas de um determinado Sprint
- Procura de um Sprint / Tarefa através do seu Sprint ID / ID Tarefa
- *Storypoints* distribuídos em um determinado Sprint
- Obter totais de sprints
- Tarefas e totais de tarefas atribuídas a um determinado *Assignee (*Utilizador*)*
- Estado das tarefas de um *Assignee*
- *Timeline* – saber o percurso completo de uma tarefa
- Para um intervalo de tempo, Tarefas atribuídas, concluídas, horas gastas, para um ou vários *Assignees*
- *Token Authentication (Autenticação á API Jira de um projeto através de Tokens)*

E as Funcionalidades (Requisitos) não Funcionais

- *Responsive*
- *User Friendly*

Demos início ao nosso projeto, preenchendo o nosso *product backlog* com todas as funcionalidades

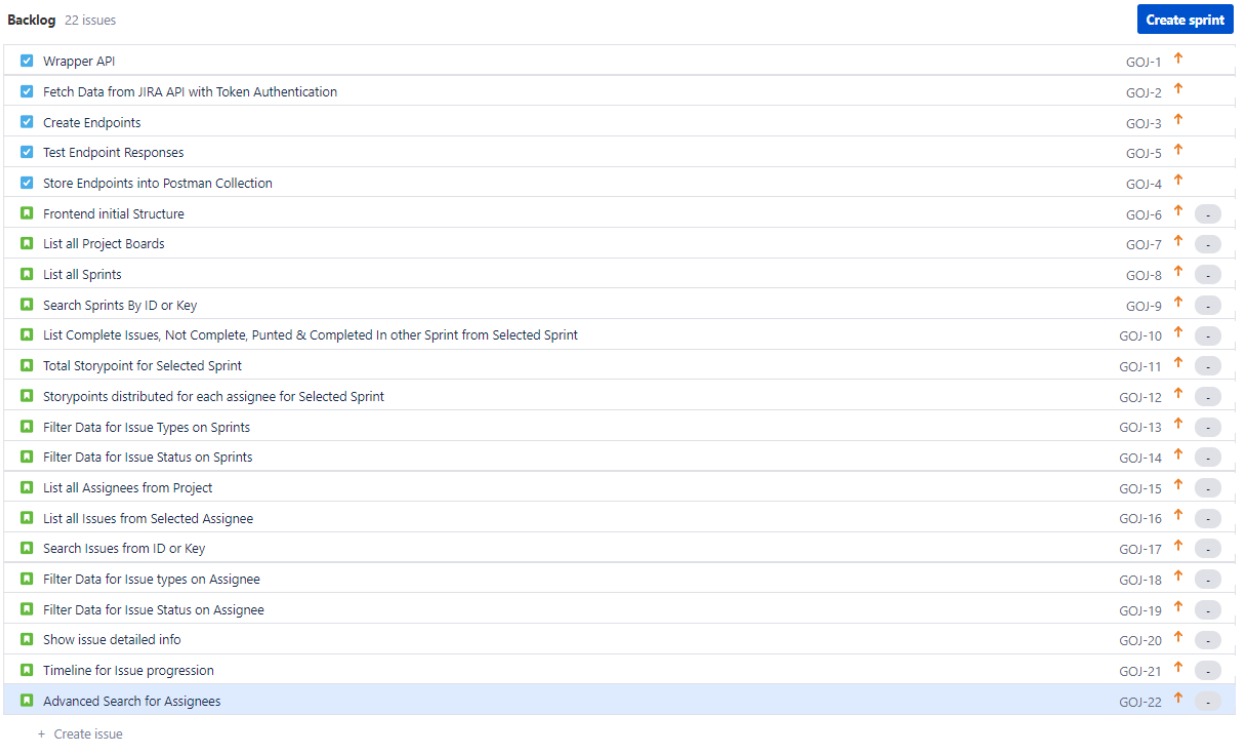

Figura 13 - Funcionalidades GoJira

4.3 Arquitetura de Software

Depois de determinada a complexidade e as funcionalidades do projeto, determinou-se a arquitetura na qual o software seria desenvolvido. A arquitetura define o modelo de software e o seu funcionamento. Responder a questões como por exemplo: Como é que as diferentes partes do sistema vão comunicar? Quais as linguagens de programação a serem utilizadas? Estas decisões são importantes e tornam-se cada vez mais difíceis fazer alterações a esta á medida que o projeto vai progredindo. Para isso é importante definir bem a arquitetura em uma fase inicial.

Desta forma também tornará o projeto inteiro mais fácil de entender, ajudando a tomar qualquer decisão necessária.

Foi então discutida a arquitetura da aplicação a desenvolver tendo em conta as funcionalidades a implementar, optando assim por utilizar a biblioteca *React* para o desenvolvimento do *Frontend*, e *Node* para o *Backend* (*Figura 14*).

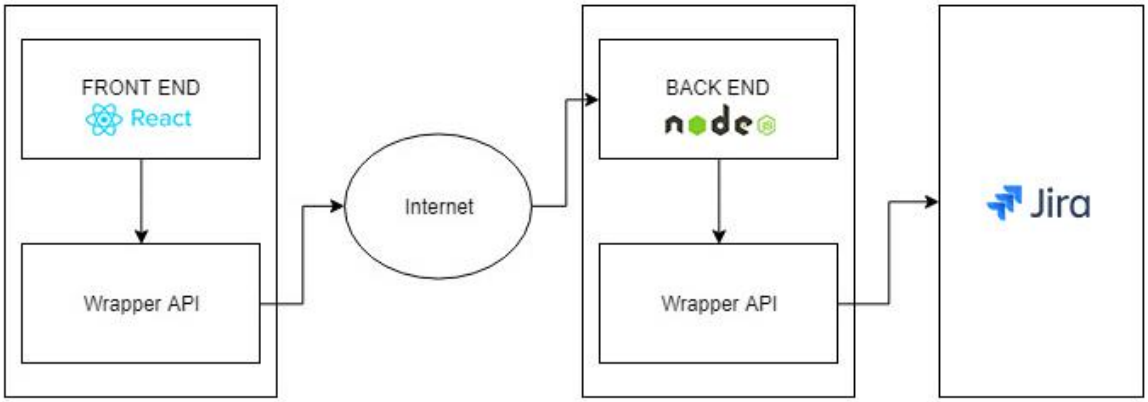

Figura 14- Arquitetura Software GoJira

4.3.1 Frontend React

Atualmente existem mais de 200.000 websites / aplicações a utilizar a *framework React*, existindo assim vários motivos para utilizá-la, Seguem-se então alguns dos motivos que tornam esta *framework* das mais utilizadas.

- **Desenvolvimento eficiente** - *React* permite a criação de código limpo, e fácil de entender, *React* utiliza componentes, ou seja, divide o projeto em várias partes, sendo estas reutilizáveis. Dividindo o projeto em componentes, também torna mais fácil fazer qualquer alteração necessária ao código.
- **Desempenho** - Esta *framework* foi desenvolvida a pensar no melhor desempenho possível, permitindo assim criar aplicações bastante rápidas, mesmo sendo aplicações de grande escala.
- **React Native** - Utilizando a mesma ideologia, *React native* é uma *framework* para o desenvolvimento mobile que se baseia também na utilização de componentes, sendo assim ao adquirir conhecimentos em React, também estamos a adquirir conhecimentos de programação *mobile* que podem vir a ser bastante uteis no futuro.

4.3.2 Backend Node

A cada dia que passa novas tecnologias vão evoluindo e ganhando atenção no desenvolvimento de software, com estas necessidades as tecnologias vão sendo polidas e melhoradas em termos de capacidades, disponibilizando mais soluções para as diversas aplicações.

- **Desenvolvimento Web com Node** - Primeiramente o node permite desenvolver para a web, *mobile* ou aplicações com apenas uma linguagem de programação, o Javascript.
- **Milhares de bibliotecas existentes** - Existem milhares de bibliotecas prontas a ser utilizadas para qualquer projeto.

4.4 REST

Rest (Representational State Transfer) é um tipo de comunicação por protocolo *HTTP*.

No caso da nossa aplicação, tendo um Cliente-Servidor terá de existir uma forma de comunicação entre os dois, esta comunicação acontece utilizando o protocolo *HTTP*, na parte do *backend* definimos uma série de serviços que podem ser pedidos pelo cliente através de pedidos *HTTP*.

4.5 Wrapper API

Quando existem determinadas funções a ser executadas com alguma complexidade, podemos recorrer a um *Wrapper*, esta é geralmente uma classe abstraída de todo o projeto, quando for necessário utilizar esta "Lógica" complexa em qualquer parte do nosso projeto, irá ser mais fácil, sendo apenas necessário chamar a função que está contida no Wrapper. Caso seja necessário mudar esta lógica, será apenas necessário alterar o *Wrapper*.

4.6 Planeamento e desenvolvimento da aplicação

As funcionalidades foram divididas por quatro sprints, como já foi referido anteriormente, sprints na metodologia *Scrum* são ciclos com duração entre 1 a quatro semanas onde a(s) equipa(s) trabalham para completar as tarefas atribuídas nesses *Sprints* .

4.6.1 Desenvolvimento do 1º Sprint

Desenvolvimento de todas as tarefas de *Backend*, criação dos *Endpoints* e documentação dos mesmos com o auxílio da ferramenta *Postman* (*Figura 15*)

Objetivos do *Sprint* :

- Estabelecer comunicação com a API Jira através de uma autenticação por *Tokens*
- Criar todos os *Endpoints* necessários
- Testar as respostas de todos os *Endpoints*
- Documentar *Endpoints* em uma Coleção *Postman*
- Criação do *Wrapper*

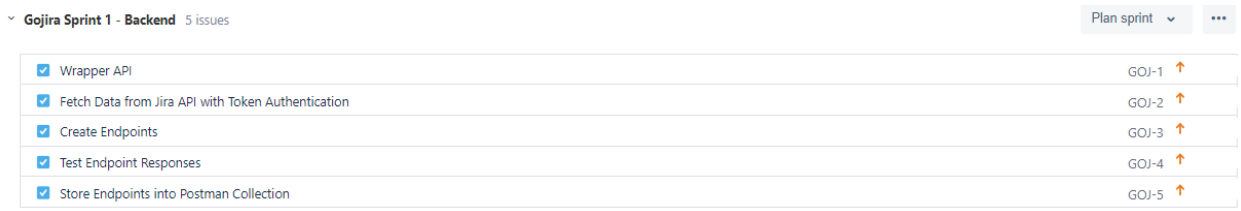

Figura 15 – Tarefas a Completar no Sprint Nº1

Como resultado do *Sprint* (*Figura 16*) existe agora um *backend* funcional, com todos os *endpoints* necessários ao desenvolvimento da aplicação criados e documentados.

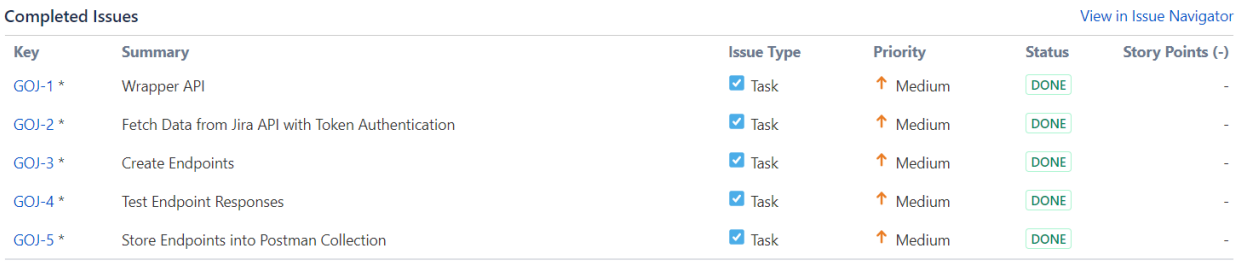

Figura 16 – Tarefas Completas do Sprint Nº1

4.6.2 Desenvolvimento do 2º Sprint

Desenvolvimento do *Frontend*, nomeadamente a monitorização dos Sprints (*Figura 17*).. No final deste segundo *Sprint* deverá ser possível monitorizar todos os *Sprints* de um Projeto (*Figura 18*).

Objetivos do *Sprint*:

- Desenvolvimento da estrutura inicial do *Frontend*
- Listar todas as *Boards* de um Projeto
- Listar todos os *Sprints*
- Procura de um *Sprint* através do seu ID ou *Key*
- Listar Tarefas Completas, Não completas, Tarefas eliminadas e tarefas completas em um outro *Sprint*, para um *Sprint* Selecionado
- Filtrar dados por tipos de tarefa para um determinado *Sprint*
- Filtrar dados por estado de tarefa para um determinado *Sprint*
- Obter Total de *Storypoints* para o *Sprint* Selecionado
- Obter *Storypoints* distribuídos por Utilizador para um *Sprint* selecionado

Figura 17 - Tarefas a Completar no Sprint Nº2

Completed Issues

View in Issue Navigator

Key	Summary	Issue Type	Priority	Status	Story Points (-)
GOJ-6 *	Frontend initial Structure	Story	Medium	DONE	-
GOJ-7 *	List all Project Boards	Story	Medium	DONE	-
GOJ-8 *	List all Sprints	Story	Medium	DONE	-
GOJ-9 *	Search Sprints By ID or Key	Story	Medium	DONE	-
GOJ-10 *	List Complete Issues, Not Complete, Punted & Completed in other Sprint from Selected Sprint	Story	Medium	DONE	-
GOJ-11 *	Total Storypoints for Selected Sprint	Story	Medium	DONE	-
GOJ-12 *	Storypoints distributed for each assignee for selected Sprint	Story	Medium	DONE	-
GOJ-13 *	Filter Data for issue Types on Sprints	Story	Medium	DONE	-
GOJ-14 *	Filter Data for issue Status on Sprints	Story	Medium	DONE	-

Figura 18 - Tarefas Completas no Sprint Nº2

4.6.3 Desenvolvimento do 3º Sprint

Continuação do desenvolvimento *Frontend*, agora respetivamente á monitorização das tarefas (*Figura 19*).

Figura 19 - Tarefas a Completar no Sprint Nº3

Objetivos do *Sprint*:

- Listar todos os Utilizadores de um Projeto
- Listar Tarefas para um determinado Utilizador
- Procura de uma Tarefa através do seu ID ou *Key*
- Filtrar dados por tipos de tarefa para um determinado Utilizador
- Filtrar dados por estado de tarefa para um determinado Utilizador
- Mostrar Informações detalhadas de Tarefa

Completed Issues

View in Issue Navigator

Key	Summary	Issue Type	Priority	Status	Story Points (-)
GOJ-15 *	List all Assignees from Project	Story	Medium	DONE	-
GOJ-16 *	List all Issues from Selected Assignee	Story	Medium	DONE	-
GOJ-17 *	Search Issues from ID or Key	Story	Medium	DONE	-
GOJ-18 *	Filter Data for issue types on Assignee	Story	Medium	DONE	-
GOJ-19 *	Filter Data for issue Status on Assingee	Story	Medium	DONE	-
GOJ-20 *	Show issue detailed info	Story	Medium	DONE	-

Figura 20 - Tarefas Completas no Sprint Nº3

4.6.4 Desenvolvimento do 4º Sprint

O quarto e último sprint foi destinado ás ultimas duas tarefas do projeto que consistem no desenvolvimento de uma pesquisa avançada e na construção de um *timeline* para todas as tarefas (*Figura 21*).

Objetivos do *Sprint*:

- *Timeline* que permita verificar a progressão de uma Tarefa
- Construção de uma Pesquisa avançada que permita obter dados para múltiplos *Assignees* e filtrar os mesmos para um intervalo de datas

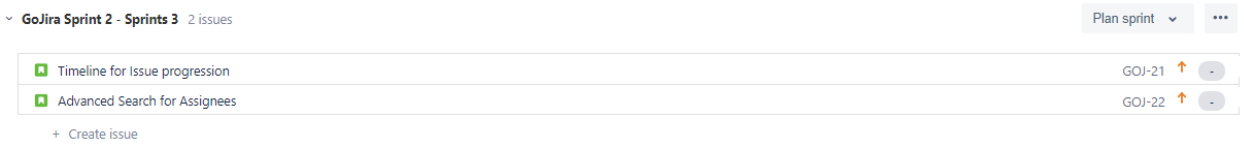

Figura 21 - Tarefas a Completar no Sprint Nº4

Completed Issues

View in Issue Navigator

Key	Summary	Issue Type	Priority	Status	Story Points (-)
GOJ-21	Timeline for issue progression	Story	Medium	DONE	-
GOJ-22	Advanced Search for Assignees	Story	Medium	DONE	-

Figura 22 - Tarefas Completas no Sprint Nº4

No final deste quarto e último Sprint, podemos dar como concluída a primeira versão de GoJira, todas as funcionalidades em *backlog* foram implementadas, todos os *sprints* foram concluídos dentro do tempo especulado e sem quaisquer imprevistos, tudo isto graças á utilização dos conhecimentos adquiridos não só das Metodologias mas como também da arquitetura e formações.

4.7 Verificação e correção de erros Eslint Airbnb

No desenvolvimento de todo o *Backend* e *Frontend* foi utilizada a Ferramenta *Eslint* (*Figura 23*), neste caso o Plugin para *Visual Studio Code* (Editor Código *Open-Source*), que permitiu com que o código não tivesse qualquer tipo de erros utilizando um conjunto de regras que podem ser definidas adaptando-se a qualquer projeto, para melhorar a qualidade de código através de boas práticas de programação foi recomendado utilizar um conjunto de regras pré-defenidas disponibilizadas pela *Airbnb*.

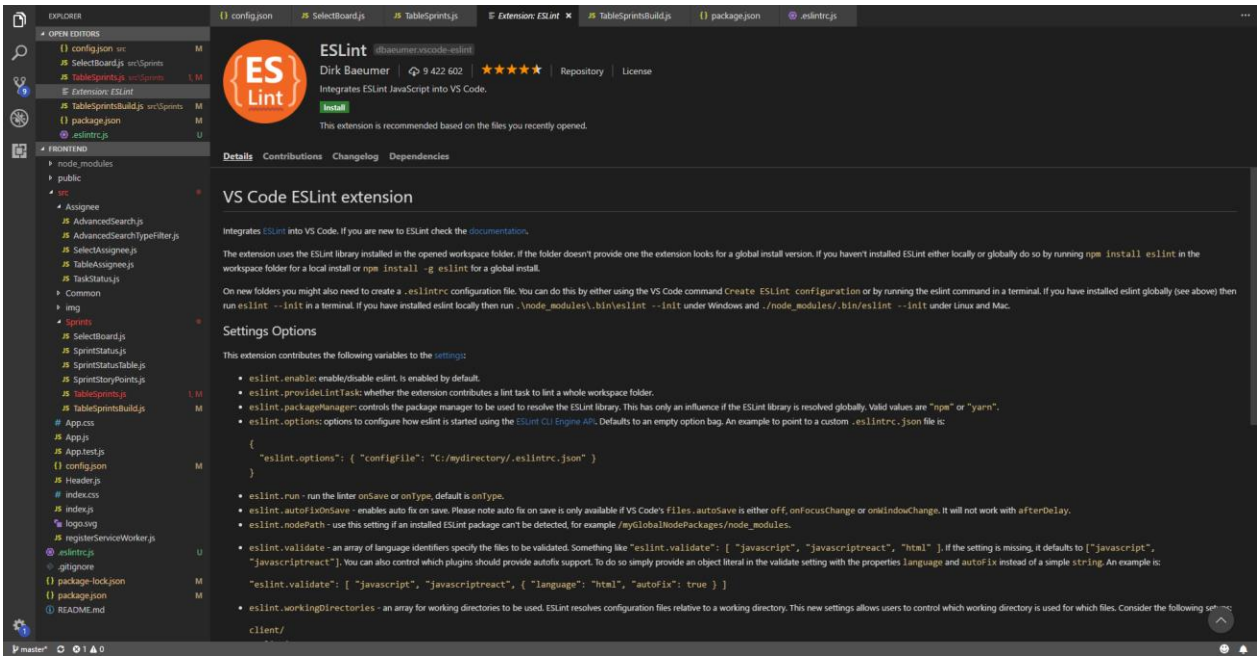

Figura 23 - Plugin Eslint Visual Studio Code

4.8 Aplicação desenvolvida

4.8.1 Exemplo Monitorização de Sprints

Podemos assim utilizar a *API* do Nosso Projeto JIRA para monitorizar o nosso próprio projeto.

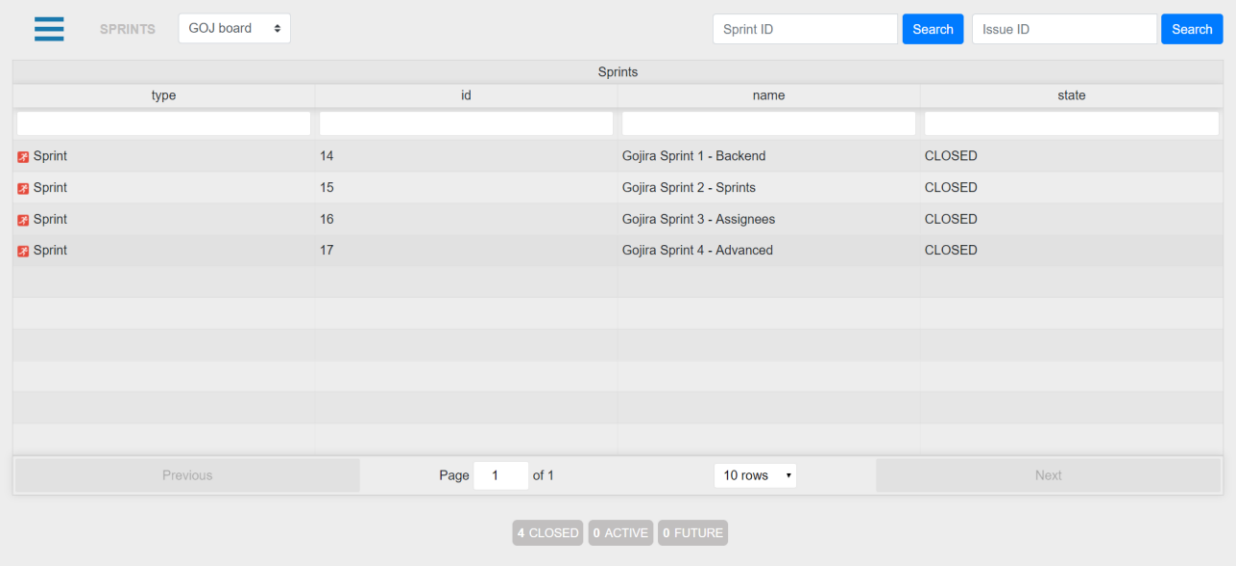

Figura 24 - GoJira Listagem de Sprint

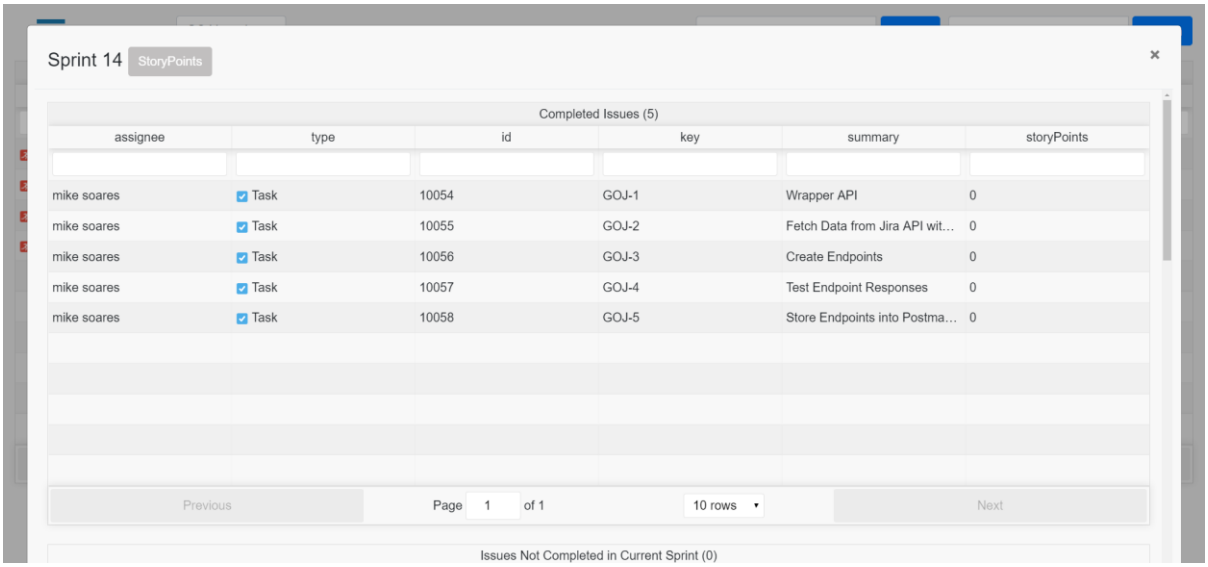

Figura 25 - GoJira Tarefas do Sprint Backend

Podemos verificar os Sprints do nosso projeto (*Figura 24*) quatro *Sprints*, todos eles concluídos, podemos também verificar as tarefas que foram atribuídas a um determinado *Sprint* (*Figura 25*)

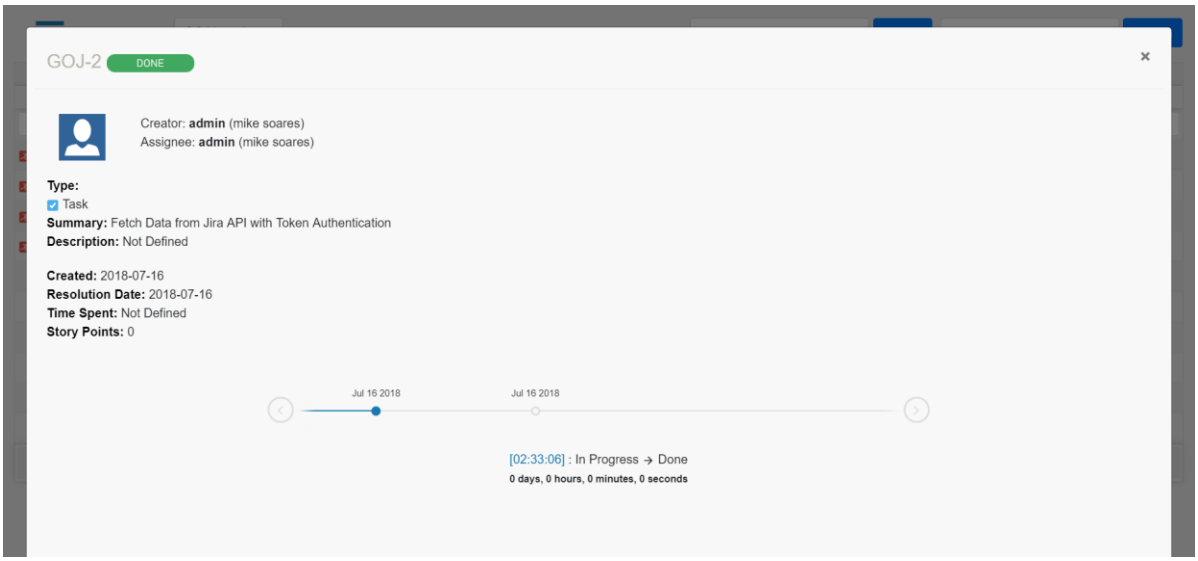

Figura 26 – GoJira Detalhes de Tarefa

GoJira permite também verificar todos os detalhes de uma tarefa, quem criou, a quem foi atribuída, Descrição, data de criação, data de resolução, bem como os *Storypoints*, a Timeline permite-nos verificar todo o percurso de uma tarefa, no caso do nosso projeto (*Figura 26*).

O percurso das tarefas em GoJira é curto, apenas temos dos estados, a atribuição da tarefa e a sua resolução , não sendo possível demonstrar todo o seu potencial, mas em projetos de maior complexidade onde existem diversas equipas, onde a tarefa passa por diversas fases, havendo a possibilidade de esta voltar para estados anteriores se assim for necessário. Como por exemplo, no caso da existência de uma equipa de testes, a tarefa é desenvolvida, mas chumba na fase de testes, esta volta para o desenvolvimento. Sendo nestes casos uma mais valia saber todo o percurso das tarefas, e o tempo entre estes estados.

4.8.2 Exemplo Monitorização de Assignees

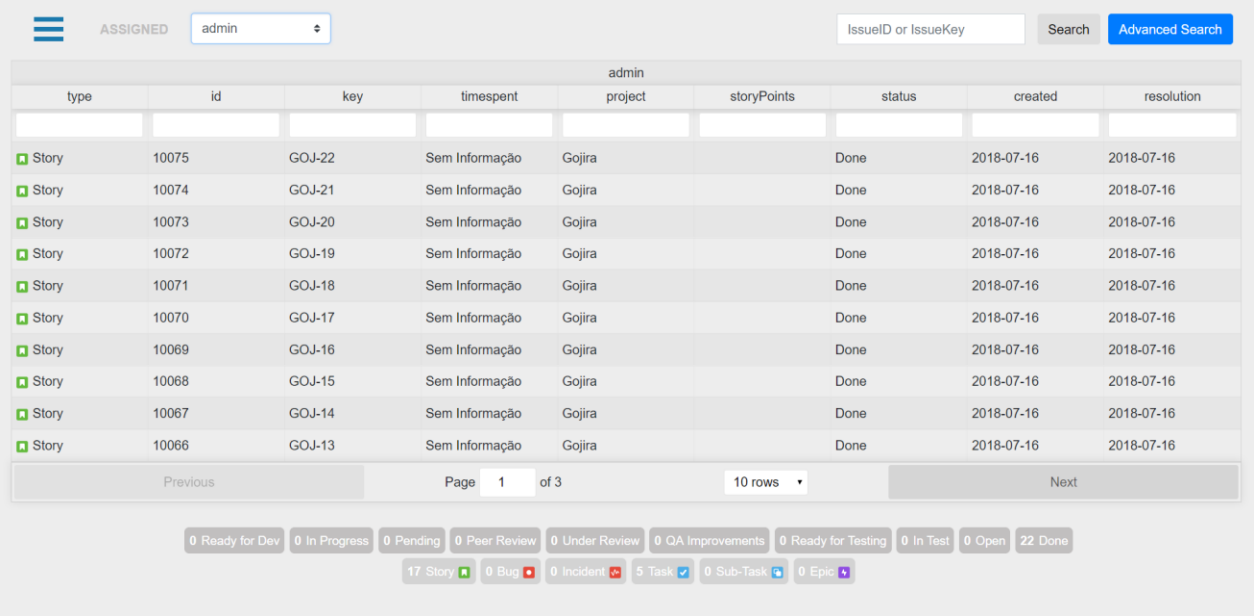

Figura 27 – GoJira Listagem de tarefas por Assignee

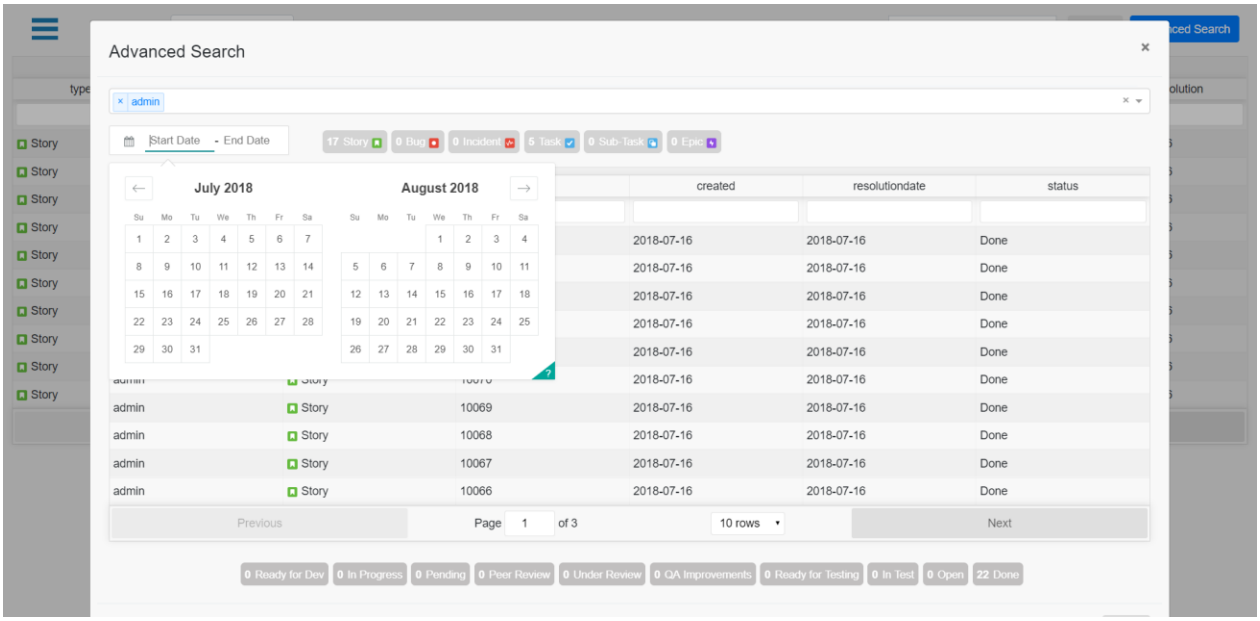

Figura 28 - GoJira Procura Avançada

Selecionando um *Assignee* (Utilizador) podemos verificar todas as tarefas que lhe foram atribuídas, podemos também filtrar por estados ou tipos de tarefa (*Figura 27*)

A Procura avançada permite selecionar múltiplos *Assignees* e definir um intervalo de datas, por exemplo: As tarefas do utilizador XPTO desde 20 de Janeiro 2017 até 16 de Julho 2018 (*Figura 28*).

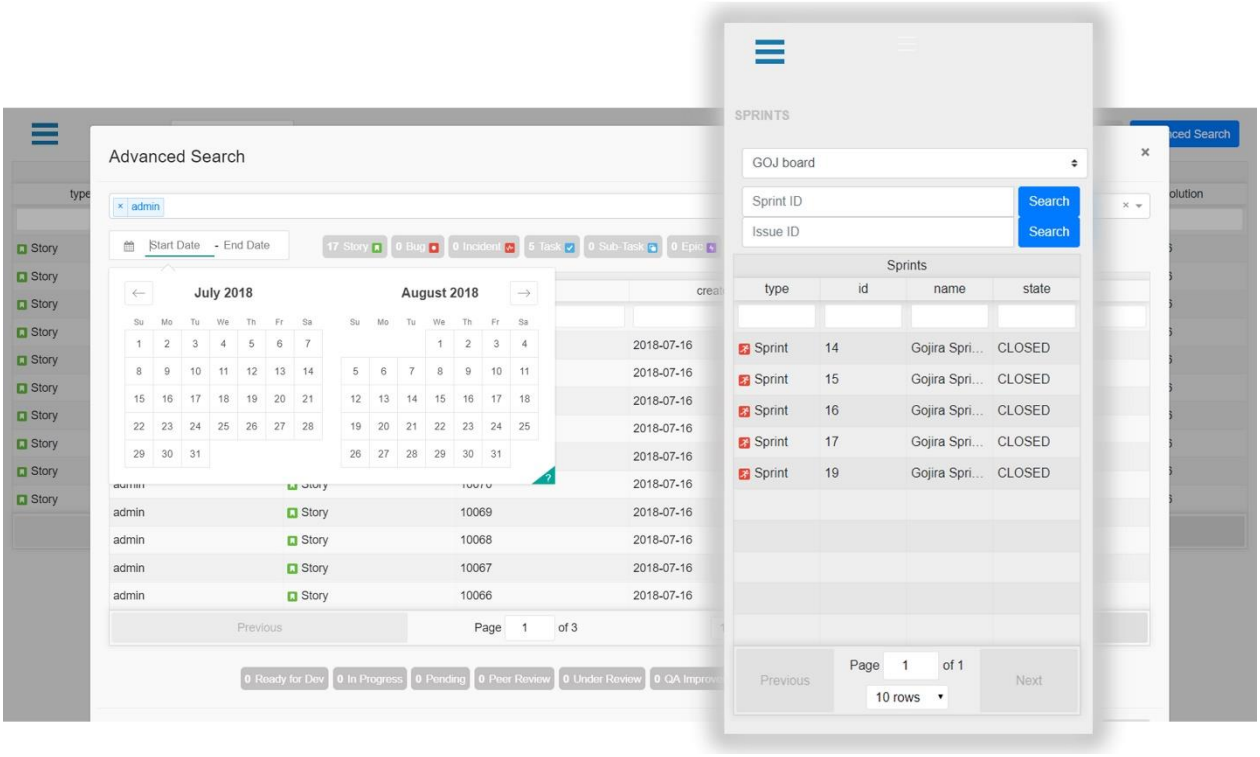

Figura 29 – GoJira Reponsive Design

Toda a Aplicação foi desenvolvida utilizando *Bootstrap* o que tornou a aplicação responsiva adaptando-se a qualquer tamanho de ecrã (*Figura 29*).

4.9 GoJira no GitHub

Por fim, depois de uma aplicação completamente funcional e sem erros foi altura de disponibilizar todo o projeto em um Repositório GitHub (*Figura 30*) [4].

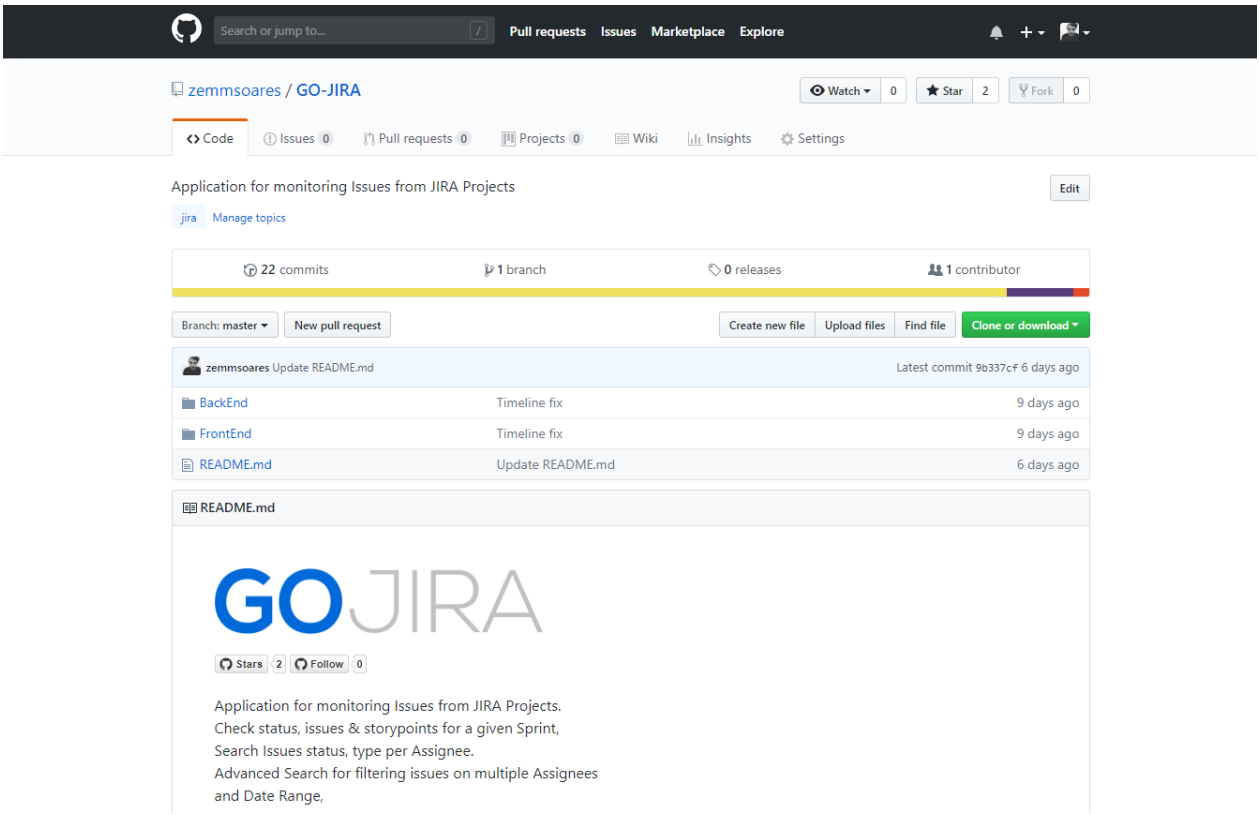

Figura 30 – Repositório GoJira no GitHub

Reflexão Final

Considerações finais

Aqui são apresentadas as conclusões finais de todo o trabalho desenvolvido ao longo destes quatro meses de estágio, uma breve reflexão sobre os objetivos alcançados, a contribuição deste projeto, quer a nível profissional, como a nível pessoal, e as suas limitações.

Chego a esta fase do meu percurso como aluno, e sinto que atingi com sucesso os objetivos a que me propus inicialmente, com o desenvolvimento de uma aplicação *React* e *Node*, que me permitiu adquirir conhecimento, utilizando as metodologias de desenvolvimento, percecionando o impacto que estas têm no decorrer do desenvolvimento de qualquer tipo de software. Não menos importante foi entender o conceito de Arquitetura de Software, quais os intervenientes, qual a importância e os cuidados a ter em conta na decisão de uma arquitetura para o projeto, conseguindo desenvolver

uma Aplicação Web *Cross-Platform*, intitulada de GoJira.

Como foi referido no início deste relatório, JIRA é um software comercial que permite a monitorização de tarefas e o acompanhamento de projetos garantindo a gestão de todas as suas atividades num único lugar, servindo diariamente milhares de utilizadores, para planear, rastrear e lançar os seus produtos. No entanto, tem também as suas limitações, nomeadamente no que diz respeito à facilidade de monitorização de um projeto, surgindo assim a necessidade de desenvolver esta aplicação.

GoJira permite obter todas as informações essenciais de um projeto, como listar os sprints e os seus estados, obter tarefas de um determinado sprint, obter informações detalhadas sobre essas tarefas, bem como um uma *timeline* para saber o seu percurso completo no projeto.

Durante este estágio profissional estive em contato com a realidade da programação informática, observando as competências que os profissionais desenvolvem, como também diferentes estratégias a utilizar em determinadas situações. O facto deste estágio apresentar uma prática pedagógica supervisionada foi uma mais valia para a reflexão do projeto que estava a desenvolver.

Tive as minhas dificuldades, os meus momentos de desespero, mas que consegui ultrapassar, com o empenho e dedicação, através de muita pesquisa, formação para a obtenção de conhecimento,

conhecimento este que me encaminhou para o sucesso desta importante aplicação. Fico com a sede de mais conhecimento nesta área, de melhorar sempre mais, a nível profissional este foi o início no mundo da programação, a nível pessoal o inicio de um grande sonho.

Limitações

Na concretização deste Relatório de Estágio Profissional surgiram algumas dificuldades, nomeadamente no que diz respeito à elaboração de trabalhos escritos, onde tenho alguma fragilidade. Outra adversidade sentida está relacionada com a questão do tempo, pois durante o período de estágio dediquei-me preferencialmente ao Estágio Profissional, nomeadamente ao desenvolvimento da aplicação, do que propriamente à realização deste Relatório.

GLOSSÁRIO DE CONCEITOS

Api : Conjunto de *Endpoints*

Assignee : Responsável por determinada tarefa

Backend : Responsável por fornecer serviços ao *Frontend* (aplicação cliente)

Cross-Platform : Multiplataforma, implementado de forma a ser utilizado em diversas plataformas.

Developer : Que ou o que desenvolve

Endpoints : Endereço URL que permite ao cliente aceder a um determinado serviço do *Backend*

Feedback : Comentário á reação de um determinado produto

Framework : É um conjunto de conceitos usado para resolver um problema de um domínio específico

Frontend : É a parte que interage diretamente com o cliente recebendo a informação através do *Backend*.

Full Stack : É aquele que lida com o desenvolvimento do *Backend* e *Frontend*

Meetings : Encontros / Reuniões

Node : Ferramenta Baseada em Javascript para o desenvolvimento do *Backend*.

Open-Source : Modelo de desenvolvimento que promove o licenciamento livre, para que qualquer consulte, examine ou modifique o produto

React – Biblioteca Javascript baseada em componentes para o desenvolvimento de *Frontend* para a Web.

React native – Biblioteca Javascript baseada em componentes para o desenvolvimento de *Frontend* para Aplicações Mobile

Scrum – Técnica popular da metodologia *Ágile*

Sprints – Agrupamento de tarefas a serem desenvolvidas por uma determinada equipa com um prazo de entrega estipulado

Storypoints – Pontos atribuídos a uma tarefa

Timeline - Linha cronológica.

Packages – Pacotes ou Blocos de Código que podem ser utilizados e partilhados em diversos projetos

BIBLIOGRAFIA

[1] Simplilearn “Guide to Scrum Methodology” [Online] https://www.simplilearn.com/free-scrum-methodology-guide-pdf

[2] The Web Developer Bootcamp [Online] https://www.udemy.com/the-web-developer-bootcamp/learn/v4/overview

[3] The Complete Node.js Developer Course (2nd Edition) [Online] https://www.udemy.com/the-complete-nodejs-developer-course-2/learn/v4/content

[4] Repositório GoJira GitHub [Online] https://github.com/zemmsoares/GO-JIRA