Universidade de Coimbra
Faculdade de Ciências e Tecnologia da Universidade de Coimbra
Departamento de Engenharia Informática
Engenharia de Software (PL6)
2018/2019

Arquitectura de Software
V 0.1

Informação do Documento			
Autor	Carla Mendes (carladrianamendes@gmail.com)	Data	12.11.2018

Objetivo

O objetivo deste documento é descrever na íntegra o software utilizado no processo de implementação do projeto da equipa OrcWall.

Link:

Change History			
V 0.1	Ínicio do documento	13.11.2018	Carla Mendes

Glossário	
API	Application programming interface / Interface de programas de aplicações.
OS	Operating System / Sistema Operativo
MEAN	MongoDB, Express.js, AngularJS, Node.js
DBMS	Database Management Systems / Sistema Gerenciador de Banco de Dados
SQL	Structured Query Language / Linguagem de Consulta Estruturada
NoSQL	Not Only SQL
ODM	Object Data Modeling /

Índice

Referências

ORCID API
https://members.orcid.org/api/about-public-api

Twitter API
https://developer.twitter.com/en/docs/tweets/search/overview

Node.js
https://nodejs.org/en/

Documento de Requisitos do OrcWall
https://github.com/carladrim/carladrim.github.io/blob/master/Requisitos/ES2018_ORCWall_DocumentoDeRequisitosV4.2.pdf

Lista de Imagens

1 Introdução

2 Possíveis Softwares

2.1 Possíveis Frameworks

2.1.1 Node.js

Node.js é uma plataforma para desenvolvimento de aplicações server-side baseadas em rede utilizando JavaScript e o V8 JavaScript Engine, ou seja com Node.js podemos criar uma variedade de aplicações Web utilizando apenas linguagem JavaScript.

Uma importante diferença entre Node.js e outras plataformas do mesmo género está no facto de esta ser *single thread.* Embora possa parecer uma desvantagem à primeira vista a verdade é que simplifica extremamente o desenvolvimento da aplicação.

2.1.2 Expressjs

Express é uma framework para aplicações web do Node.js flexível que fornece um conjunto robusto de recursos para aplicações *web* e *mobile*. Tem como linguagem JavaScript e funciona em vários OS, tem como plataforma o Node.js. Graças a estes dois *softwares* o *back-end* pode ser desenvolvido inteiramente em JavaScript. Em conjunto com a base de dados MongoDB e o *framework front-end* AngularJS formam o MEAN stack.

2.2 DBMS

2.2.1 MongoDB

MongoDB é um DBMS open-source orientado para documentos NoSQL. O termo NoSQL deve-se à ausência de SQL, ou seja, não assume as ideias do modelo relacional nem a linguagem SQL. Temos assim a primeira diferença entre os dois modelos em que o DBMS orientado para documentos lida com documentos e não com registos em tabelas bidimensionais. É uma das vantagens em usar um DBMS NoSQL, pois acelera o desenvolvimento da aplicação e reduz a complexidade das implementações.

2.2.2 Mongoose

Mongoose é uma biblioteca ODM para o MongoDB e Node.js. É responsáel pelas relações entre dados, oferece validação de esquemas e é usado para traduzir objetos em código e a representação desses objectos no MongoDB. Ou seja é responsável pela ligação entre a base de dados e o JavaScript.

2.3 Twitter API

2.3.1 Search Tweets API (Standard)

Para possibilitar a implementação de requisitos tal como o Re. 7 do Documento de Requisitos, que tem como objetivo implementar o feed do Twitter no *dashoard*. Para tal é possível implementar uma API do Twitter ou uma combinação de várias.

O Search Tweets API (Standard) permite encontrar tweets recentes ou populares e tem um comportamento semelhante, mas não igual, ao Search UI do Twitter. O Search tweets (Standard) procura entre os tweets dos últimos 7 dias.

2.4 Orcid API

A API Orcid permite sistemas e aplicações conectarem-se aos registos do Orcid, incluindo escrever e ler registos Orcid. A versão pública deste API permite ser autenticado com o ID do Orcid e procurar/obter dados públicos de ID's Orcid e informação feita pública por um titular de ID.

2.5 Autenticação

2.5.1 Passport.js

Passport.js proporciona várias estratégias de autenticação a partir de Facebook, Google+, Twitter, etc. É usado em conjunto com o Node.js.

2.6 Encriptação

2.6.1 BCryptHashing

De maneira a proteger a password de cada utilizador e a aumentar a segurança da informação de cada um destes, a password deve ser encriptada, para o efeito pode ser usado o BcryptHashing.

2.7 Testes

2.7.1 Jenkins

Através do Jenkins, é possível agilizar tarefas demoradas como a compilação de um projeto e a execução dos seus testes automaticamente. Com um servidor de integração contínua, essas tarefas são executadas a cada mudança no repositório de código e, em caso de erros de compilação ou falhas nos testes automáticos, os developers são alertados rapidamente. Jenkins é a nova versão (forkada) de Hudson.

2.7.2 SonarQube

O SonarQube é uma plataforma *open-source* que analisa de forma contínua a qualidade do código para realizar revisões automáticas que detetem *bugs*, vulnerabilidades de segurança, etc. Com base no código inserido, a plataforma aplica as regras pré-definidas e verifica se estão a ser cumpridas.

3 Softwares Escolhidos

Depois apresentados os possíveis *softwar*es às equipas de Implementação e Testes, e com a aprovação da Unidade de Gestão, foram escolhidos os seguintes *softwares*:

- **Framework**
  - Node.js
  - Express.js
- **DBMS**
  - MongoDB
  - Mongoose
- **Twitter API**
  - Search Tweets API
- **Autenticação**
  - Passport.js
- **Encriptação**
  - BcryptHashing
- **Testes**
  - SonarQube

4 Utilizadores do Sistema

4.1 Utilizador Anónimo

Qualquer utilizador na web que não tenha credenciais autenticadas para utilizar a plataforma.

4.2 Utilizador Registado

Utilizador que efetuou Registo na plataforma e cujas credenciais estão autenticadas.

5 Descrição da Arquitetura - Modelo C4

5.1 C4:Contexto do Sistema

O diagram de contexto do Sistema é usado para mostrar todos os utilizadores do Sistema que irão interagir com o projeto e as respetivas funcionalidades. É também usado para mostrar todos os sistemas externos que serão utilizados para suportar os requesitos do projeto, baseando-se no documento de Requisitos (V 4.1). Como se pode ver no diagrama seguinte, os Utilizadores do Sistema são os que estão explicitos no ponto 4. Um User Anónimo e um User Registado. Cada um deles tem um conjunto de ações que podem efetuar na plataforma. Essas ações estão descritas brevemente no diagrama, e detalhadamente no ponto anterior (4 – Utilizadores do Sistema) deste documento. No diagram é possível ver também os Sistemas Externos utilizados no projeto e uma breve descrição de como interagem com a plataforma. Uma explicação mais detalhada encontra-se no ponto 2 (Possíveis Softwares) e 3(Softwares Escolhidos) deste documento.

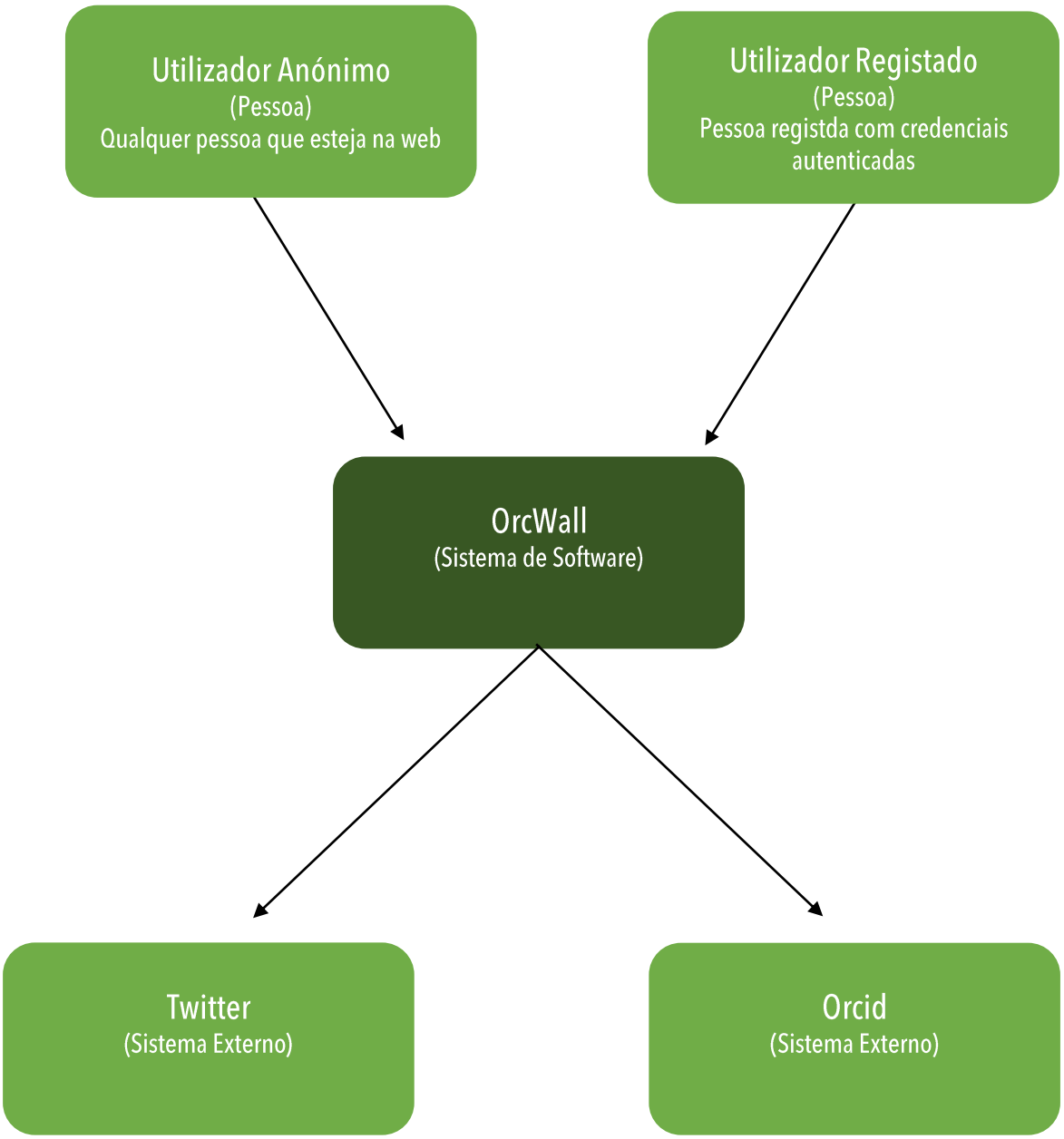

Figure #1 - C4: Diagrama de Contexto do Sistema para o projeto OrcWall

5.2 C4:Containers

O diagramas de Containers é uma representação de todos os containers inseridos no projeto em si e de que maneira os containers interagem entre eles e com os utilizadores do Sistema. Cada container tem uma breve descrição do seu uso na plataforma e como interage com esta, nomeadamente o protocolo de comunicação e o porte caso necessário. O diagrama apresenta também os protocolos de comunicação utilizados na interação com os sistemas exteriores escolhidos para o projeto. Uma explicação mais detalhada sobre o uso de cada container está presente no ponto 2 (Possíveis Softwares) e 3 (Softwares Escolhidos) deste document, assim como a razão pela qual o container foi escolhido.

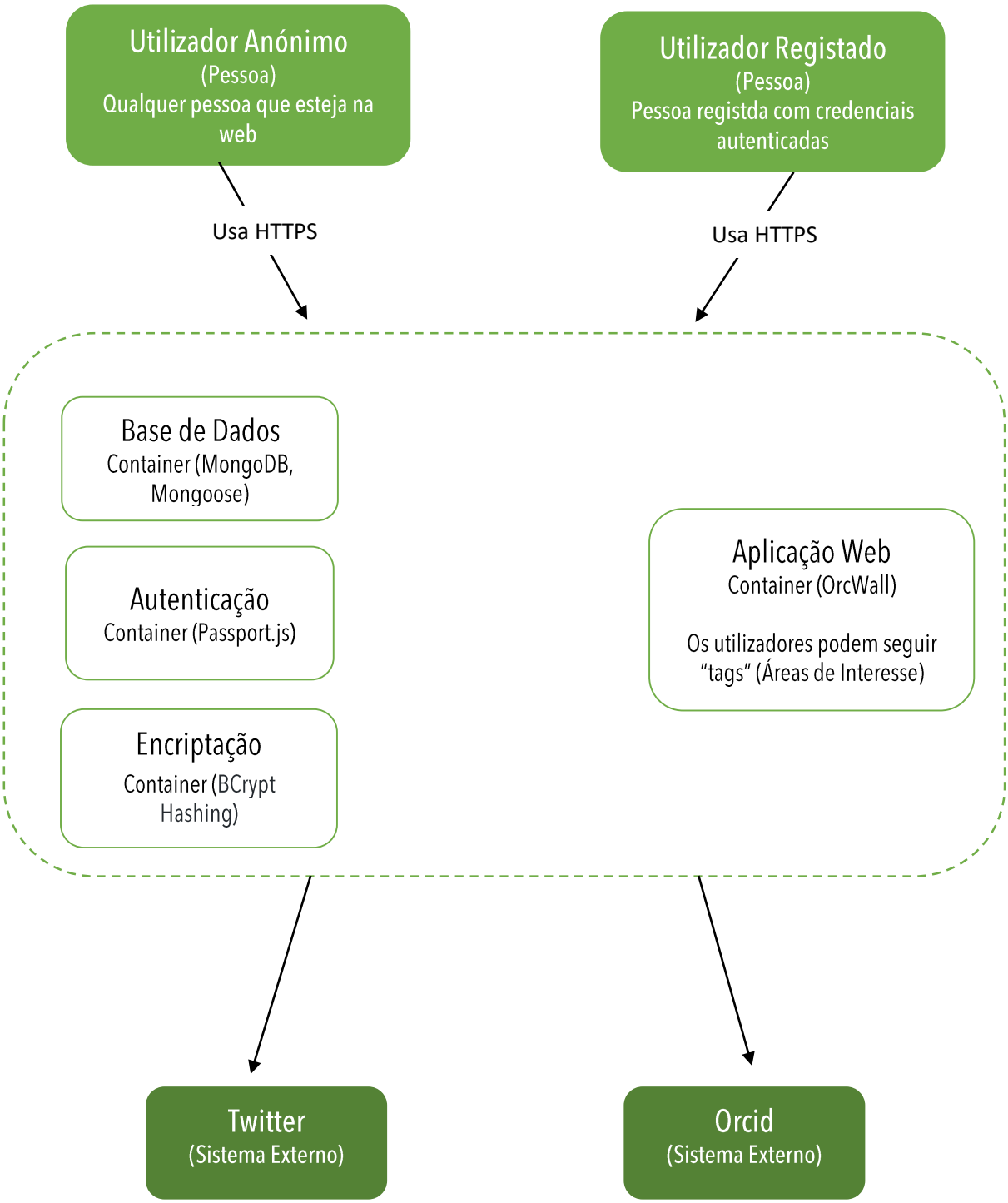

Figure #2 - C4: Diagrama de Containers do Sistema para o projeto OrcWall

5.3 C4:Componentes

O diagram de Componentes é uma representação de todos os components necessários para implmentar todos os requisites descritos no documento de Requisitos (V4.1) e como todos os componentes interagem entre si e com o container do projeto. Todos os componentes tem uma breve descrição de como eles irão interagir coma aplicação web em si e qual o seu uso no projeto, segue-se uma explicação mais detalhada.

Componentes:

- **Twitter API – Search Tweets API(Standard):** Esta API é utilizada para devolver ao utilizador os tweets mais recentes, relevantes às áreas de interesse (tags) escolhidas pelo utilizador.

- **ORCID API (pública):** A API do ORCID é o componente mais importante do sistema. Esta API permite ao utilziador autenticar o seu ID do ORCID, entrar no sistema a partir da sua conta ORCID, ler e retirar informação publicada no ORCID por qualquer utilizador do mesmo e procurar informação na API mencionada.

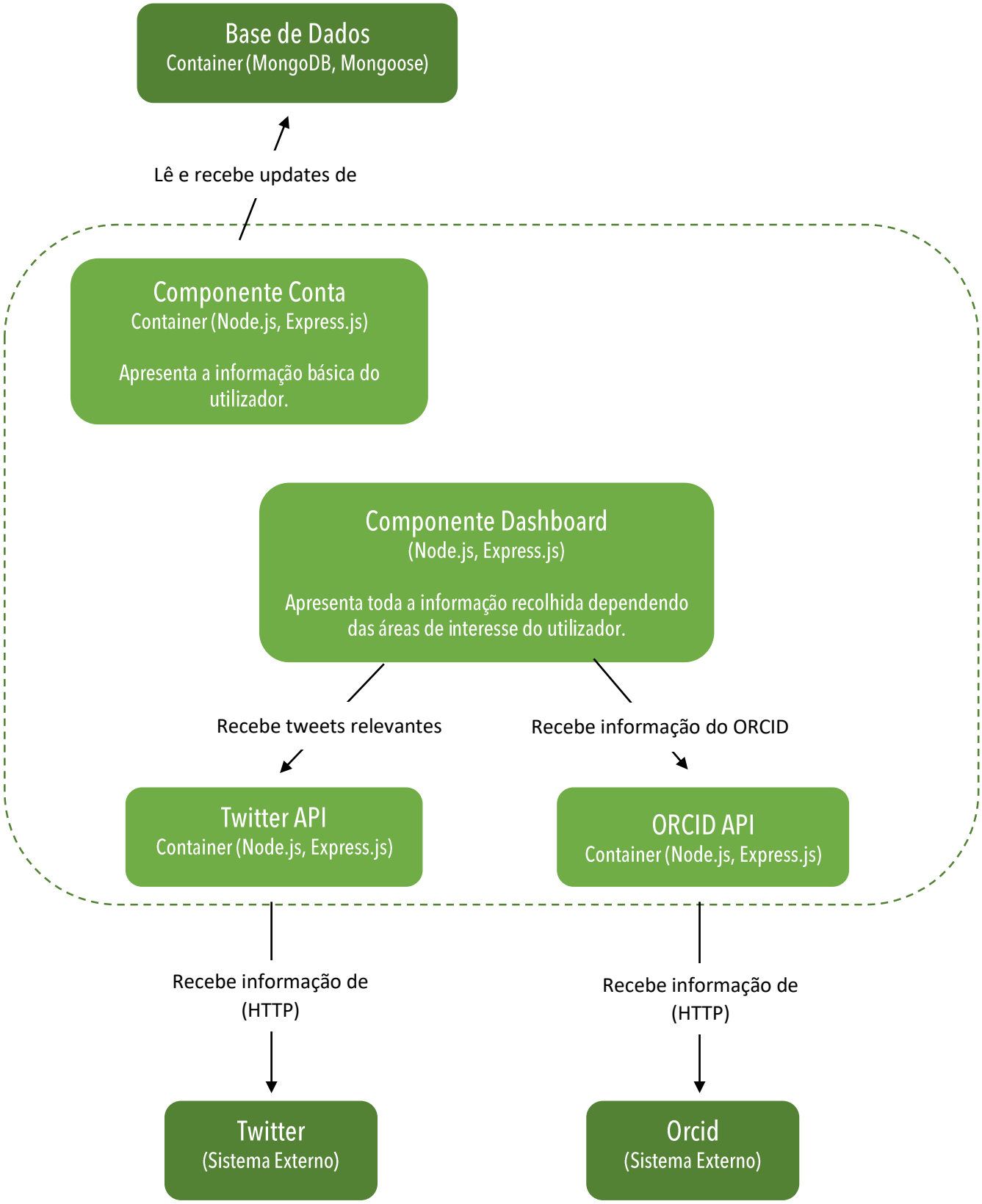

Figure #1 - C4: Diagrama de Componentes do Sistema para o projeto OrcWall

5.4 C4:Diagrama User - Componente

Este diagrama representa as relações que todos os users do Sistema irão ter com os componentes e controladores do Sistema. Isto foi explicado textualmente no diagram de Componentes representado em cima e como maneira de simplificar a legibilidade das interações existentes este diagram foi criado. Uma explicação mais detalhada desta relação está descrita na secção número 5.3 Componentes deste documento.

6 Base de Dados

6.1 Entidades

6.1.1 User

Existem dois tipos de users que foram brevemente descritos no pornto 4 – Utilizadores do Sistema deste documento:

- **Users Anónimos**

  Um user anónimo não está registado no sistema, as suas credenciais não estão validadas e é considerado um Guest User.
  O user anónimo apenas consegue aceder às páginas de login e registo.

- **Users Registados**

  Um user anónimo não está registado no sistema, as suas credenciais estão validadas. Todos os users registados conseguem aceder às informações básicas de cada user (exceto a password), em concreto as tags que cada user segue.

A entidade User tem os seguintes atributos:

- Email (PK)
- ID
- Password
- Primeiro Nome
- Útimo Nome
- Área de Atuação
- Instituição
- Áreas de Interesse (Tags)

Não existem outra entidades para além do User.

7 Conclusão

Para concluir este document é necessário indicar que esta pode não ser a melhor arquitetura nem a mais complexa mas é a arquitetura que mais se adequa às necessidades da equipa na melhor maneira. A unidade de Implementação, e as capacidades dos membros da unidade, teve um grande impacto na escolha dos Softwares de maneira a facilitar a implementação, o desenvolvimento da plataforma e a diminuir a complexidade da Arquitetura.