

TQS: Product specification report

André Alves [89334], Miguel Mota [89331], Luís Fonseca [89066], João Silva [88813]
v2020-05-03

1	Introduction	1
1.1	Overview of the project	1
1.2	Limitations	1
2	Product concept	2
2.1	Vision statement	2
2.2	Personas	2
2.3	Main scenarios	2
2.4	Project epics and priorities	2
3	Domain model	2
4	Architecture notebook	3
4.1	Key requirements and constrains	3
4.2	Architetural view	3
4.3	Arquitetura de instalação	3
4.4	System architecture	4
4.5	API for developers	4
5	API de integração	4
6	References and resources	4

1 Introduction

1.1 Overview of the project

O projeto no âmbito cadeira de Testes e Qualidade de Software tem como objetivo desenvolver um sistema de multicamadas com uma aplicação web e uma estratégia de *Software Quality Assurance* (SQA), seguindo todo um processo de engenharia software.

Com isto, surgiu Just Like Home, um sistema para locatários puderem arrendar os seus bens imóveis online e para que clientes que estejam à procura de uma residência consigam alugar de forma fácil e eficaz.

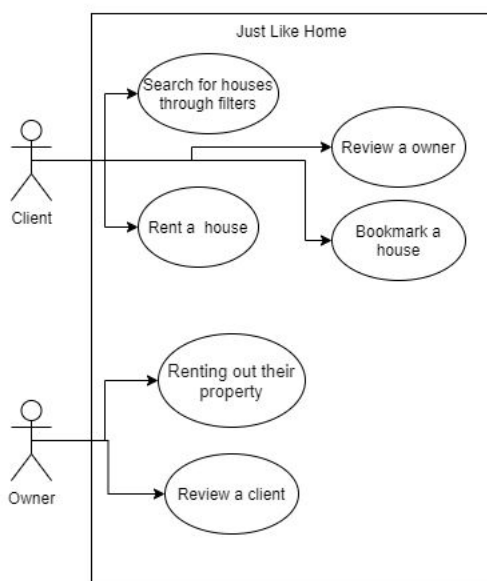
1.2 Limitations

Devido a vários motivos, existem obviamente algumas limitações existentes aquando da conclusão do projeto. Não obstante, tendo em conta o *scope* da cadeira, não se consideram críticas no que diz respeito ao produto como um todo, e às lições dele retiradas:

- O dono de uma casa após aceitar uma proposta não a pode cancelar.
- Não existe um sistema de mensagens entre os utilizadores dentro do nosso sistema.
- Os utilizadores que possuem anúncios de casa, conseguem dar update a qualquer especificação menos de *comodities*.

2 Product concept

2.1 Vision statement



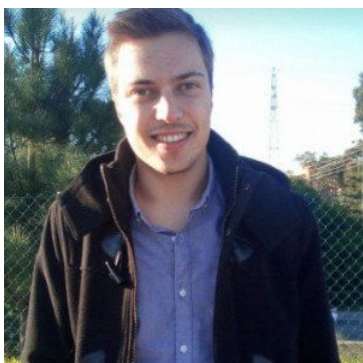
Há muitas plataformas de alugar casas/apartamentos online, porém na maioria a parte visual e lógica não são intuitivas, criando muitas vezes confusão por parte das pessoas que conseqüentemente desistem de usufruir deste serviço. Portanto, Just Like Home vem tentar simplificar este processo, tornando possível os locatários criarem anúncios das várias residências que têm para arrendar, que necessitam de várias especificações para que não crie dúvidas sobre o local. O cliente consegue fazer uma pesquisa personalizada através de vários filtros, para que consiga encontrar uma residência para alugar de forma fácil ou, ainda, marcar como favorito. No processo de alugar, o cliente faz um pedido com as respetivas datas de início e fim de estadia ao locatário, no qual, este é que decide se aceita ou não.

Por fim, o cliente consegue fazer uma review da residência onde esteve para que ajude nas decisões de próximas pessoas que estejam interessadas no mesmo local e o locatário consegue fazer uma review de um cliente,

procurando assim evitar, que por exemplo, um locatário faça contratos com clientes que possam criar

algum prejuízo no local que alugou, visando portanto a beneficiar ambas as partes. Em cima, encontra-se um diagrama de casos de uso onde se resume os vários tipos de utilização no sistema.

2.2 Personas



Persona 1 - O José Guerreiro é um professor de ensino básico recém-formado de 24 anos que vive no Porto, solteiro e tem dois irmãos. Há pouco tempo, recebeu uma proposta para trabalhar numa escola em Galegas em Leiria, para tal, necessita de uma residência.

Como é um jovem adulto, a tecnologia sempre foi uma constante na sua vida. Por isso, procura uma plataforma onde seja possível de forma fácil encontrar o apartamento ideal e a um bom preço nesta zona desconhecida. Para além disso, um dos seus irmãos está acabar o secundário com intuito de prosseguir para um ensino superior na Universidade de Aveiro.

Portanto, buscam uma plataforma confiável, eficiente e fácil para que consigam encontrar as suas segundas casas.



Persona 2 - A Joana Ferreira é uma mulher de 67 anos reformada que trabalhava numa lavandaria em Águeda no distrito de Aveiro. É uma senhora casada há 35 anos e tem um filho. Como o seu trabalho não lhe trazia muito rendimento financeiro e a sua reforma ainda é pior, decidiu investir na compra de dois apartamentos com o propósito de arrendar a turistas na época de verão ou a trabalhadores, de forma a ganhar um dinheiro extra.

Contudo, não tem tido muita adesão. O seu filho apenas lhe criou uma página no facebook para ajudar na divulgação. Mas, não tem atingido público suficiente. Portanto, precisa de ajuda nesta matéria de forma a chegar a várias pessoas que queiram alugar os seus

apartamentos para que consiga ter um melhor sustento na sua vida financeira.

2.3 Main scenarios

Case 01: O José e a Joana registam-se na plataforma para ter acesso a mais funcionalidades.

Case 02: O José pesquisa apartamentos de acordo com as suas preferências, de forma a encontrar um lugar à sua medida. Localização e duração parecem-lhe fatores importantes.

Case 03: O José encontra uma residência que lhe interessa e para tal indica data de início e data de fim da estadia para pedir ao proprietário.

Case 04: Após o termo da sua estadia, o José pretende dar uma boa review à casa onde esteve.

Case 05: A Joana quer criar anúncios dos seus apartamentos para arrendamento. Procurando promover a satisfação dos eventuais clientes, quer que estes saibam exatamente como é o local para onde estão a vir, de forma a não se virem a arrepender.

Case 06: Quando alguém mostra interesse num dos anúncios da Joana, esta verifica se o candidato tem um perfil válido segundo os seus padrões, decidindo aí se quer aceitar ou recusar o pedido dessa pessoa.

































Case 07: Quando o apartamento volta a ficar livre, a Joana trata ela mesma de o limpar e preparar para as próximas pessoas, aproveitando para averiguar em que condições o mesmo ficou, o que lhe permite dar reviews aos residentes anteriores.

Case 08: Na sua procura por um sítio para o seu irmão ficar, o José encontra vários sítios promissores. Deseja então guardá-los, para mais tarde ver com o irmão.

Case 09: Tanto o José como a Joana, fazem login no sistema, para que consigam criar anúncios, alugar bens imóveis, fazer reviews e adicionar apartamentos/casas aos favoritos.

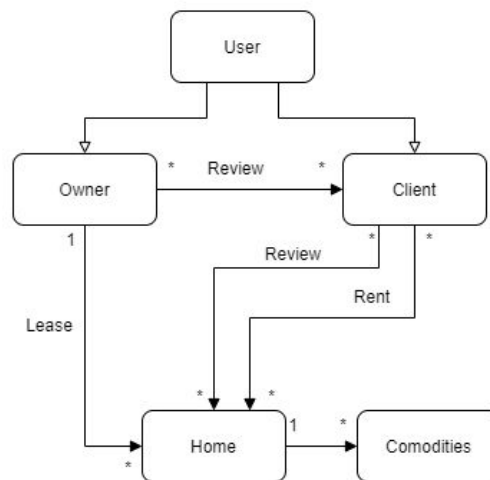
2.4 Project epics and priorities

Utilizou-se a plataforma Jira como backlog onde eram colocadas todas as user stories, procurando assim definir quais as user stories que devem ser implementadas em cada iteração, a user story que se encontrava mais em cima no backlog era a que tinha mais prioridade, ou seja, a que necessita de ser implementada em primeiro. O Jira apresenta também uma espécie “calendário” onde se pode visualizar os prazos em que os user stories foram e devem ser implementados, como se pode observar na figura:

Epic		ABR - JUN	JUL
<ul style="list-style-type: none"> Conceptualization <ul style="list-style-type: none"> Software Architecture Definition of Use Cases Definição de Personas 			
<ul style="list-style-type: none"> Case: Add property <ul style="list-style-type: none"> Insert Property Feature 			
<ul style="list-style-type: none"> Case: Search Property <ul style="list-style-type: none"> Controllers Testing House Controller Test Page of Property Specifications Search Properties Feature 			
<ul style="list-style-type: none"> Case: See User Profile <ul style="list-style-type: none"> User see their profile Endpoint for user information Test user information 			
<ul style="list-style-type: none"> Case: Rent Property <ul style="list-style-type: none"> Rent Property Testing Owner shouldn't be able to rent... Add validation when posting rent Ask Rent a property 			
<ul style="list-style-type: none"> Case: Bookmark House <ul style="list-style-type: none"> Check houses bookmarked Unbookmark a house Testin and logic bookmark house Bookmarker 			
<ul style="list-style-type: none"> Case: Register <ul style="list-style-type: none"> Ability to save a user password Register Feature 			
<ul style="list-style-type: none"> Case: Login <ul style="list-style-type: none"> Login Feature Test login behaviour 			
<ul style="list-style-type: none"> Case: Accept/Deny Rent <ul style="list-style-type: none"> Endpoint and logic to get all curr... Accept Guest on rent-> change... Deny Rent Owner accept rent requests 			
<ul style="list-style-type: none"> Android application <ul style="list-style-type: none"> Android 			
<ul style="list-style-type: none"> Testing Selenium 			
<ul style="list-style-type: none"> Case: Review user <ul style="list-style-type: none"> Review Client Feature Get review rating Frontend display reviews 			
<ul style="list-style-type: none"> Case: Review House <ul style="list-style-type: none"> Review Property Feature Show average rating on frontend Ability to rate when you've been to... 			
<ul style="list-style-type: none"> Case: Logout <ul style="list-style-type: none"> User Logout 			
<ul style="list-style-type: none"> Case: Posterior Commodities Increase <ul style="list-style-type: none"> Add Commodities Commodities 			
<ul style="list-style-type: none"> Case: Top 5 Houses <ul style="list-style-type: none"> Endpoint for top 3 rated houses Display top houses before specific... 			
<ul style="list-style-type: none"> Case: Owner update his own house <ul style="list-style-type: none"> Ability to update house values Update house 			

3 Domain model

No sistema existe um utilizador que pode ser um cliente ou um locatário, capaz de realizar diferentes tarefas, como se pode ver na figura abaixo. O cliente pode alugar várias residências. Adicionalmente, pode ainda realizar reviews a várias bens imóveis onde tenha estado. Cada residência está associada a um locatário, e este poderá fazer vários anúncios às diferentes residências, nas quais tem várias especificações - *comodities*, como por exemplo, uma piscina. O locatário pode também realizar várias reviews a clientes que tenham usufruído das suas residências. As residências se já estiverem com data prevista para arrendamento não serão mostradas no sistema.



4 Architecture notebook

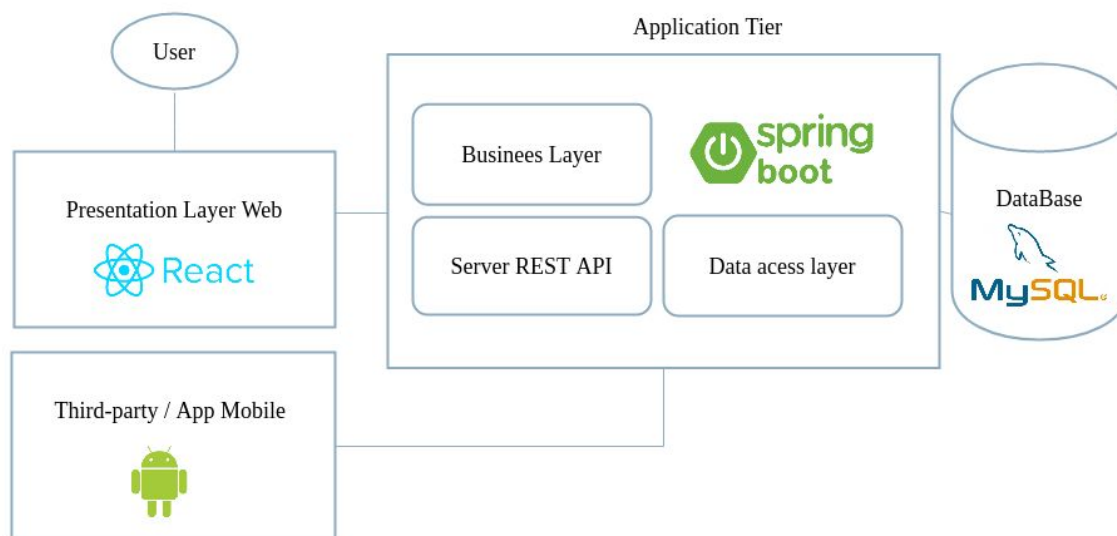
4.1 Key requirements and constraints

- Durante o desenvolvimento do projecto serão impostos métodos de Continuous Deployment através de containers Docker. Isto, para além de ser uma boa escolha em relação à performance e facilidade de deployment, garante também um processo mais fácil no caso de necessidade de mudança ou migração de server.
- O desenvolvimento do projecto será sujeito a métodos de Continuous Integration e Testing para facilitar a introdução de novas features.
- O produto terá de ser robusto e bem estruturado para que a sua manutenção ao longo do tempo seja feita com facilidade, devendo ser possível a novos membros da equipa perceber rapidamente a arquitetura e estrutura do produto.
- O produto final (versão estável) deverá ser capaz de servir um fluxo sazonal de clientes que será maior durante altura de férias (verão).
- A database deve ser persistente e manter os registos em caso de crash do servidor, uma vez a database estando em docker ter-se-á de usar volumes.
- O produto deve disponibilizar um website que será a principal Interface com o sistema para os utilizadores.
- Será necessária também uma API para que seja possível a sistemas externos usarem os serviços do produto para os seus fins, nas suas plataformas sejam elas quais forem.

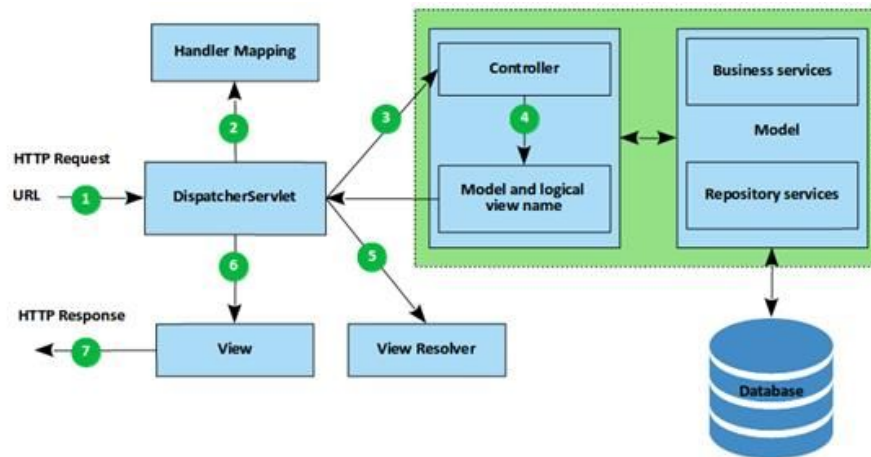
- A API terá de ser documentada para que clientes externos a consigam utilizar autónomamente.
- A utilização do website do produto deve ser regulada por identificação e password.
- O sistema deve garantir que nenhuma renda é efectuada sem o acordo de ambas as partes.

4.2 Architectural view

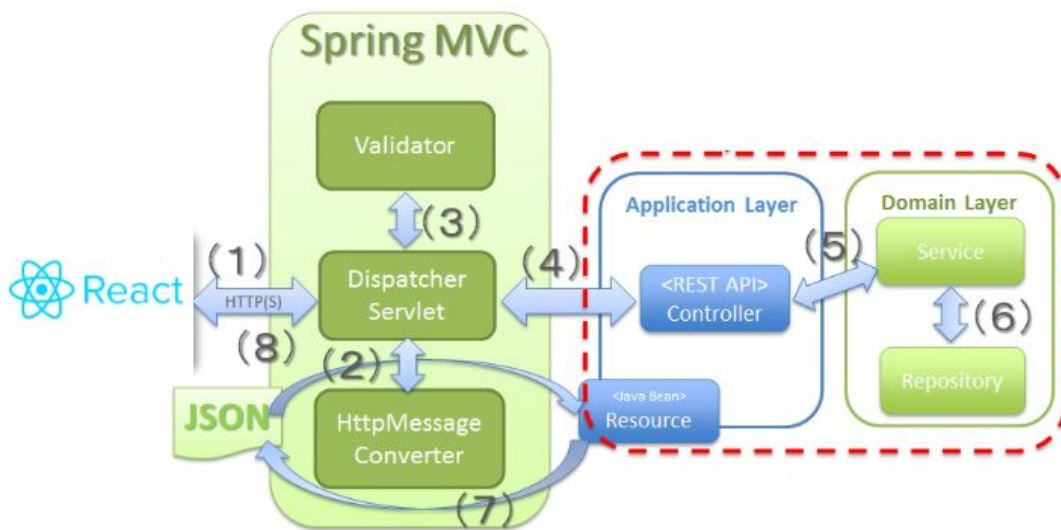
Como proposto o backend será implementado em **Spring Boot**, onde se encontra a parte lógica do nosso sistema e a REST API que poderá ser usada por outros desenvolvedores através, por exemplo de uma aplicação android. Será usado **MySQL** como sistema de base de dados, que devido ao seu carácter relacional é útil para a nossa modelação, caracterizada por várias relações, como por exemplo, a de uma casa estar associada a um locatário, simplificando posteriormente as queries. O sistema terá uma aplicação web desenvolvida em **React** onde se prevê que estejam todas as features implementadas do marketplace.



Por norma o SpringBoot usa uma arquitetura do tipo Model-View-Controller (representada na imagem em baixo).

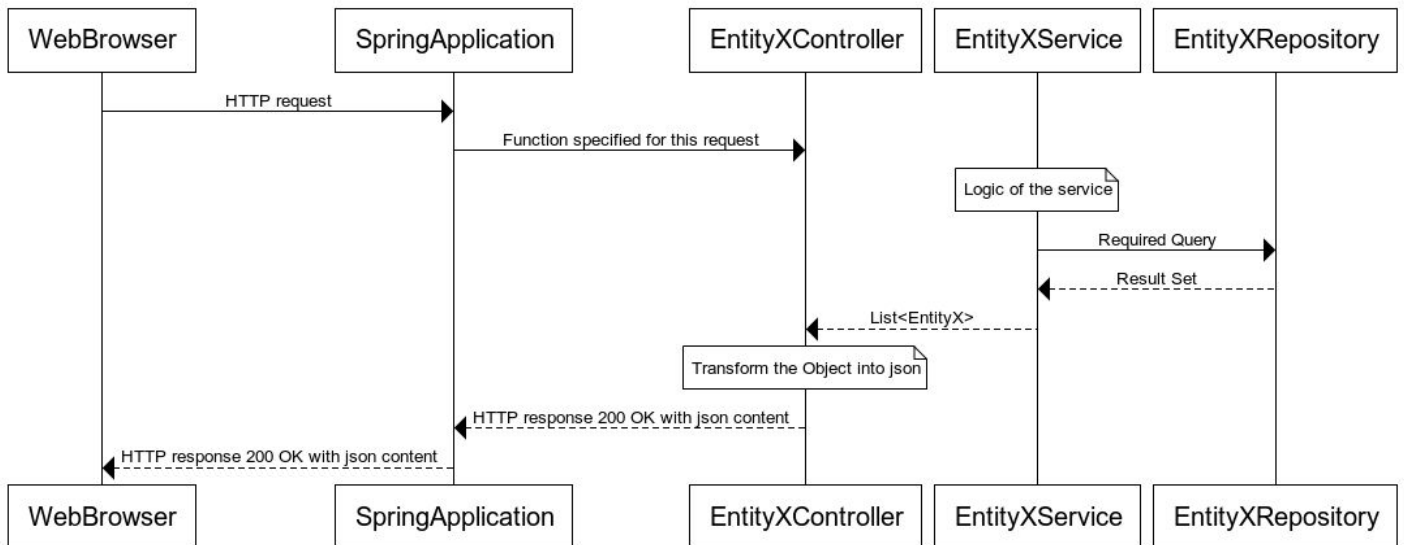


No entanto, no projecto o SpringBoot serve para se criar um RESTful Web Service , isto é em vez de o Spring-Boot gerar uma view html e enviar para o cliente, vai enviar uma resposta em JSON para o cliente onde a sua vista pega nestes dados e gera a view desejada.



Como se pode ver na imagem em cima os pedidos chegam ao dispatcher, são mapeados para o respetivo controller que posteriormente chama o serviço onde se encontra a lógica para responder ao pedido. O service faz chamadas aos repositories consoante os dados que precise , indo assim indiretamente aceder à base de dados. Após o service executar a lógica necessária para responder ao pedido devolve a resposta ao controller e este transforma os dados em json e envia para o dispatcher que por sua vez devolve ao cliente. Em baixo podemos ver o processo especificado de forma generalizada num diagrama de sequência.

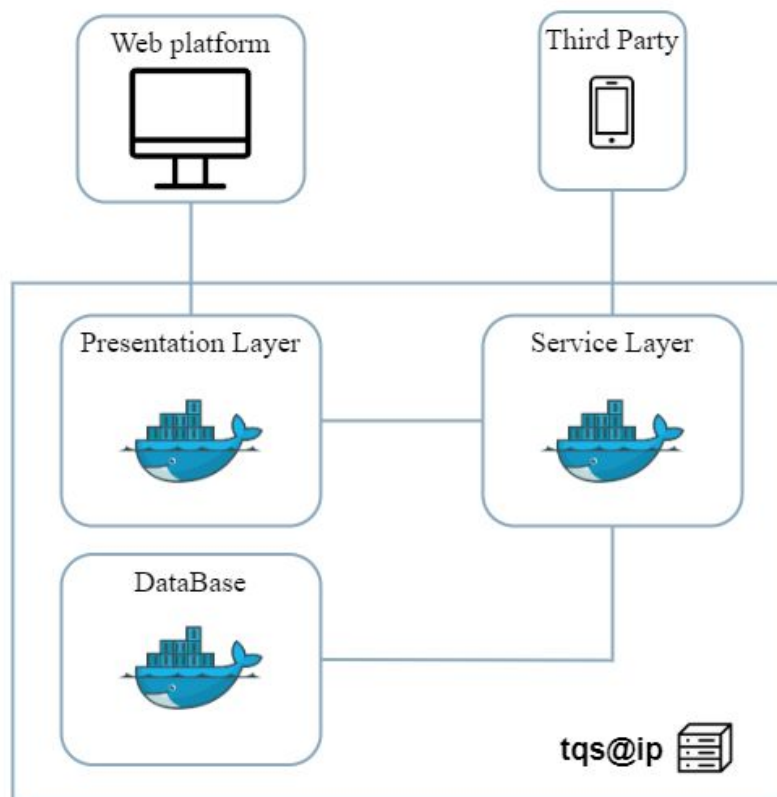
Sequence of a request



A aplicação android funciona de forma idêntica no entanto em vez de ser um webbrowser a fazer o pedido vai ser a aplicação.

4.3 Deployment architecture

A aplicação encontra-se disponível numa máquina virtual dos STIC, onde 3 containers de Docker, ou seja, um container para cada layer do sistema (Presentation Layer, Service Layer - Rest API e DataBase), isto para que se tenham todas as dependências necessárias para utilizar o nosso produto, sem ter qualquer problema. Pode-se observar o deployment através do diagrama:



Na figura abaixo demonstramos as imagens docker que correspondem ao sistema:

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
frontend	latest	2f0990a2b848	3 hours ago	2.02GB
backend	latest	443d12ab5aa5	3 hours ago	254MB
node	latest	91a3cf793116	12 days ago	942MB
openjdk	11-jre-slim	973c18dbf567	2 weeks ago	204MB
maven	3.6.0-jdk-11-slim	c7428be691f8	14 months ago	489MB
mysql/mysql-server	5.7	278f8eedd831	5 weeks ago	338MB

Na figura abaixo demonstramos os docker containers usando as respectivas imagens:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
6f1048e43e85	frontend	"docker-entrypoint.s..."	19 minutes ago	Up 19 minutes	0.0.0.0:80->3000/tcp
50b2e735d357	backend	"java -jar usr/local..."	23 minutes ago	Up 23 minutes	
806c5d928936	mysql/mysql-server:5.7	"/entrypoint.sh mysql..."	36 minutes ago	Up 36 minutes (healthy)	

De forma a que os dados mantenham persistência caso o docker com o mysql ,por alguma razão, deixe de estar ativo foi inicializado com um volume.

5 API for developers

JustLikeHome tem uma API divisível em 4 componentes: User, House, Reviews e Rents. Começando pelo User, é possível obter toda a informação do utilizador através do seu id, como por exemplo, as casas que tem nos favoritos, todos os alugueres que fez, todos os anúncios de bens imóveis que pôs à venda, etc. Além disso, a API fornece todos os utilizadores que estão registrados no sistema.

Em relação à House, um dos principais pontos a referir, é a capacidade de pesquisa, onde são retornadas várias casas tendo em conta a sua localização, número máximo de pessoas disponíveis e as datas da sua estadia. Também é possível ver toda a informação da casa através do seu id, onde se encontra todas as suas especificações, o seu locatário e reviews feitas. Por fim, adicionar os bens imóveis aos favoritos de um utilizador.

Através das Reviews é possível obter informação de todas as reviews feitas entre utilizadores (quando um utilizador aluga uma casa, o locatário pode fazer uma review à pessoa que esteve no seu bem imóvel), reviews feitas a casas e criação das mesmas.

Em relação às Rents, o principal ponto a referir é que fornece informação sobre os vários pedidos de aluguer às casas do locatário através do seu id e ver todos os aluguéis que foram aceites.

A API pode ser utilizada por outros desenvolvedores para servir de base a vários sistemas que considerem as funcionalidades suportadas úteis, podendo até posteriormente focar-se noutras funcionalidades não suportadas atualmente.

A documentação da API encontra-se em [http://192.168.160.52:8080/swagger-ui.html/](http://192.168.160.52:8080/swagger-ui.html#/) (relembrando que apenas acessível com a VPN da UA). Em baixo podemos ver uma porção do site onde a documentação se encontra.

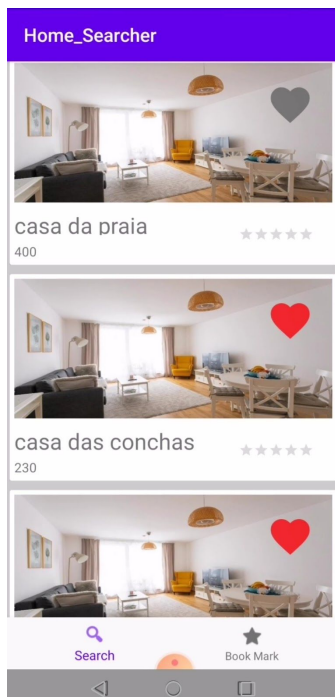
house-controller House Controller			▼
POST	/addBookmark	User bookmark a house as favourite	
POST	/addComoditie	Add a new comoditie to an existing house	
DELETE	/deleteBookmark/userId={userId}&houseId={houseId}	User stops bookmarking a house as favourite	
GET	/houses/city={city}&start={start}&end={end}&guests={guests}	Obtain all houses matching the passed parameters	
GET	/specificHouse/houseId={houseId}	Get all the information of a specific house	
GET	/topHouses	Get top rated houses	
PUT	/updateHouse	Update house values	
rent-controller Rent Controller			▼
PUT	/acceptRent	Owner accept a user rent request on a given house	
POST	/askToRent	Ask to rent a certain house	
PUT	/denyRent	Owner deny a user rent request on a given house	
GET	/onGoingRents/user={userID}	Return all on going rents on all houses of a given user	
GET	/pendingRents/user={userID}	Return all pending rents on all houses of a given user	

5.1 External Mobile App Client

Para demonstrar a expressividade funcionamento da API, foi desenvolvido um cliente fictício - “Home_Searcher”, que se trata de uma mobile app que utiliza a API para oferecer os seus serviços a consumidores.

Esta app invoca chamadas à API para Login e UserInfo o que possibilita ao utilizador ter acesso às suas casas favoritas (bookMark). Esta aplicação implementa também os métodos de procura

disponibilizados pela API para que o cliente possa pesquisar casas por região, número de ocupantes e data de estadia, assim como adicioná-la aos seus favoritos, estas mudanças são de facto efetuadas na conta do utilizador e serão visíveis no website.



6 References and resources

Para além dos documentos fornecidos ao longo das aulas a outra referência que teve grande impacto no desenvolvimento deste produto foi o <https://www.baeldung.com/> , cujos tutoriais forneceram bastante ajuda no desenvolvimento da aplicação.

As ferramentas que foram usadas ao longo do projecto foram: docker containers, mysql, spring-boot, junit5, selenium, katalon recorder, blazemeter, apache jmeter, mockito, hamcrest, sonarqube, sonarcloud, insomnia, intellij, jira, github, github actions, react.