



UNIVERSIDADE  
**LUSÓFONA**

HeatSystem  
Wakeboard

# Trabalho Final de curso

Relatório Intermédio 2º Semestre

João Luís (a21903149)

Nuno Capela (a21902803)

Pedro Serra

Trabalho Final de Curso | LEI | 25/06/2022

## **Direitos de cópia**

Software de cálculo para a gestão completa de uma competição de wakeboard, Copyright de Nuno Capela e João Luís, ULHT.

A Escola de Comunicação, Arquitectura, Artes e Tecnologias da Informação (ECATI) e a Universidade Lusófona de Humanidades e Tecnologias (ULHT) têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

# Índice

Índice.....	iii
Lista de Figuras e Tabelas.....	iv
Resumo.....	vi
Abstract .....	vii
1 Identificação do Problema .....	1
2 Levantamento e análise dos Requisitos .....	3
2.1 Requisitos Funcionais.....	3
2.2 Cumprimento de Requisitos.....	13
3 Viabilidade e Pertinência.....	14
4 Solução Proposta.....	15
5 Benchmarking.....	18
6 Método de Planeamento .....	22
7 Resultados .....	23
7.1 Plano de testes e validação .....	27
8 Conclusão e trabalhos futuros .....	31
Bibliografia .....	32

## Lista de Figuras e Tabelas

Fig. 1 – Login da nossa aplicação.	5
Fig. 2 – Adicionar atletas.	6
Fig. 3 – Editar atletas.	7
Fig. 4 – Apagar o(s) atleta(s).	8
Fig. 5 – Adicionar evento.	9
Fig. 6 – Editar evento.	10
Fig. 7 – Linguagens de programação mais usadas em 2021.	11
Fig. 8 – As várias tabelas existentes na nossa app.	12
Fig. 9 – Logout do utilizador.	14
Fig. 10 – Linguagens de programação mais usadas em 2021.	15
<b>Fig. 11 – Competição com 14 atletas.</b>	<b>16</b>
Fig. 12 – Filtragem de Heats na ronda “Semis”.	16
Fig. 13 – Filtragem de HeatsAthletes consoante a Heat.	16
Fig. 14 – Mensagem de Vencedor.	15
Fig. 15 – Criação de uma competição no software (No software já existente).	18
Fig. 16 – Criação de um evento no software (Na nossa solução).	18
Fig. 17 – Adição de atletas a uma competição (No software já existente).	19
Fig. 18 – Lista de atletas inscritos (No software já existente).	19
Fig. 19 – Adicionar o atleta a um evento (Na nossa solução).	19

Fig. 20 – Geração de baterias e visualização das mesmas (No software já existente).	20
Fig. 21 – Geração de baterias e visualização das mesmas (Na nossa solução).	20
Fig. 22 – Pontuação a ser atribuída (No software já existente).	21
Fig. 23 – Pontuação a ser atribuída (Na nossa solução).	21
Fig. 24 – Atletas de uma determinada competição.	23
Fig. 25 – Competição de 5 atletas.	24
Fig. 26 – Competição com os atletas ordenados por scores.	24
Fig. 27 – Atletas sem scores.	25
Fig. 28 – Atletas com os scores, e mensagem de quem ganhou.	25
Fig. 29 – Competição com 7 atletas.	26
Tab. 1 – Tabela de Heats (Snake Sys).	2
Tab. 2 – Tabela de Heats de LCQ.	2

## Resumo

O presente trabalho pretende o desenvolvimento de um sistema inteligente, que auxilie competições de Wakeboard fazendo, um organização específica, entre os diferentes atletas que estão a competir, através de baterias (heats), e uma atualização direta de pontuações/lugares para cada atleta, tem também a possibilidade de adicionar/remover/editar as várias entidades presentes no software, ou seja, entidades necessárias no desporto de Wakeboard. É possível fazer os mesmos processos para eventos/competições. Este software tem como objetivo ser utilizado pela International Waterski & Wakeboard Federation (IWWF).

O software irá ser desenvolvido em Django. Iniciando com a extração de uma base de dados onde estão armazenados os dados dos atletas. Após a extração estar concluída, passamos ao preenchimento das nossas tabelas com a informação que vai ser utilizada no processo.

Com as tabelas preenchidas devidamente, conseguimos criar os eventos/competições, bem como alterar dados de atletas, dependendo dos resultados obtidos no evento. Os eventos irão alterar o lugar dos atletas nos ranking existentes.

## **Abstract**

The present work intends to develop an intelligent system, which helps Wakeboard competitions, making a specific organization between the different athletes who are competing, through heats, and a direct update of scores/places for each athlete, you also have the possibility to add/remove/edit the various entities present in the software, that is, entities necessary in the sport of Wakeboard. It is possible to do the same processes for events/competitions. This software is intended to be used by the International Waterski & Wakeboard Federation (IWWF).

The software will be developed in Django. starting with the extraction of a database where the athletes' data are stored. After the extraction is complete, we proceed to fill in our tables the information that will be used in the process.

With the tables properly filled in, we were able to create the events/competitions, as well as change athlete data, depending on the results obtained in the event. Events will change the place of athletes in the existing rankings.





## 1 Identificação do Problema

Existe uma aplicação usada pela IWWF mas esta está desatualizada e requer um redesign um dos maiores problemas existentes é a descontinuidade da mesma, então será resolvido através de linguagens e frameworks recentes que possa dar continuidade ao software. Outro problema é uma interface pouco user-friendly e também pouco moderna que também vai ser tomada em conta bem como a dificuldade de uso da mesma. O projeto em si vai ter em conta vários aspetos sendo o mais importante a distribuição de atletas pelas heats:

### Como vai ser feita a distribuição de atletas?

A distribuição vai ser feita através de baterias (Heats), ou seja, sub-grupos de atletas. Esta distribuição vai ter em conta o lugar no ranking do respetivo atleta na lista da IWWF Wakeboard Boat, começando no topo da lista (jogadores com melhor ranking) para o fim da lista (jogadores com menor ranking) sendo sempre os que têm melhor pontuação que ficam para o final e os que têm menor começam primeiro. Os jogadores que não têm rank definido vão ser distribuídos de forma aleatória pelas baterias depois da distribuição dos jogadores com rank definido estar feita. Atletas que vão participar em competições de categoria diferente (dificuldade acrescida) vão ter 75% dos seus pontos no rank direcionados para a pré-distribuição.

Os números de atletas por bateria estão dependentes do total de atletas inscritos na competição que por norma não pode ultrapassar os 6 atletas.

Após estar concluída a fase de qualificação, um igual número de atletas das diferentes baterias vão passar para a próxima fase. Nesta fase a distribuição inicial já não tem papel no desenvolvimento da competição, esta vai depender do lugar que o atleta ficou nessa mesma bateria para depois ser feita a distribuição e assim sucessivamente nas rondas seguintes.

De todos os atletas que não se conseguiram qualificar na primeira fase de qualificação vão ter, entre si, uma última ronda onde terão hipótese de se qualificarem.

A distribuição dos atletas pelas diferentes baterias em todas as fases vai ser feita através do método Snake System.

### O que é o Snake System?

Vamos supor uma competição com 5 baterias e 24 atletas.

- A distribuição começa no topo da lista e o atleta com a melhor classificação vai ficar na bateria 5 na última posição.
- Em seguida, o segundo melhor atleta é colocado na bateria 4 na última posição. E assim sucessivamente.

- Quando a última posição da bateria 1 estiver preenchida, o próximo atleta é colocado na penúltima posição da bateria 1.
- Quando a penúltima posição da bateria 1 estiver preenchida, o próximo atleta é colocado na penúltima posição da bateria 2. E assim sucessivamente.

No final teríamos a seguinte distribuição de atletas:

Example with 24 riders, 5 heats (the number is the rank of the rider):

	Heat 1	Heat 2	Heat 3	Heat 4	Heat 5
1		24	23	22	21
2	16	17	18	19	20
3	15	14	13	12	11
4	6	7	8	9	10
5	5	4	3	2	1

Tab. 1 – Tabela de Heats (Snake Sys).

O mesmo acontece na LCQ, Last Qualification Heat:

LCQ Heat 1		LCQ Heat 2	
1		1	24
2 ▲	22	2	23
3 ▲	21	3	20
4 ▲	18	4	19
5 ▼	17	5	16

Tab. 2 – Tabela de Heats de LCQ.

Chegando ao fim do nosso TFC, e fazendo uma comparação ao que foi planeado no início do ano, podemos afirmar que fomos ao encontro de grande parte dos tópicos propostos, conseguindo assim fazer um software para as competições de WakeBoard muito mais atualizada, simples e fácil de se utilizar. O único tópico que não conseguimos desenvolver, foi a existência de dois métodos de distribuição dos atletas pelas diferentes heats, devido à escassez de tempo e à complexidade de cada um decidimos optar por realizar o mais utilizado atualmente, Snake System, deixando assim o Ladder System para ser desenvolvido posteriormente.

## 2 Levantamento e análise dos Requisitos

Este software permite aos responsáveis das competições (oficiais, júris, etc...) editar, criar e eliminar dados de eventos, atletas, e outras tabelas inseridas no software sendo estes os únicos capazes de os fazer.

Desta forma e para cada componente do sistema, foram identificados os seguintes requisitos:

### 2.1 Requisitos Funcionais

**RF01** - Autenticação do utilizador (login).

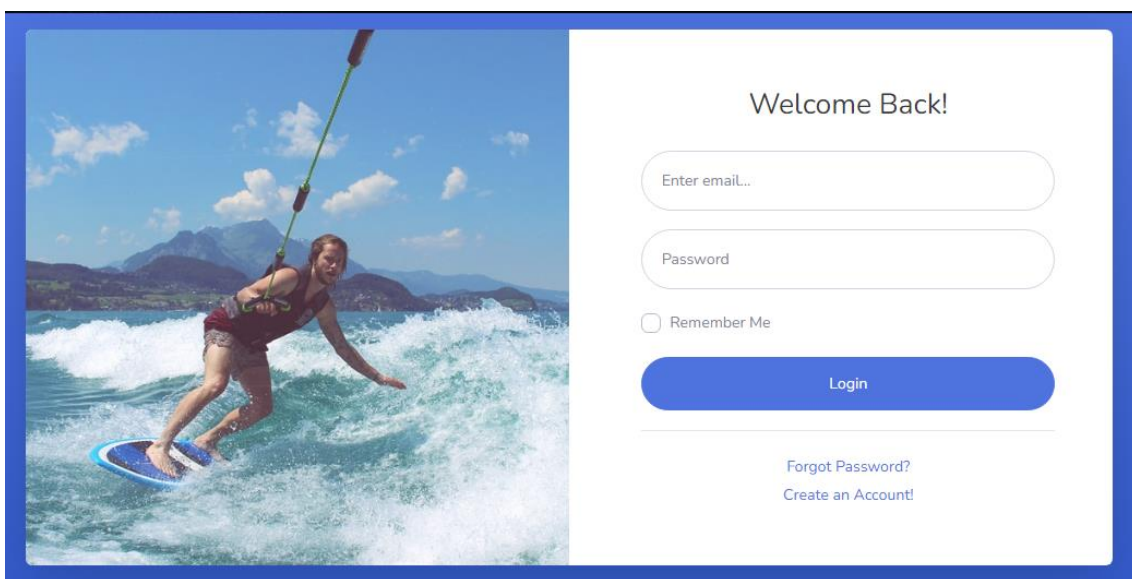


Fig. 1 – Login da nossa aplicação

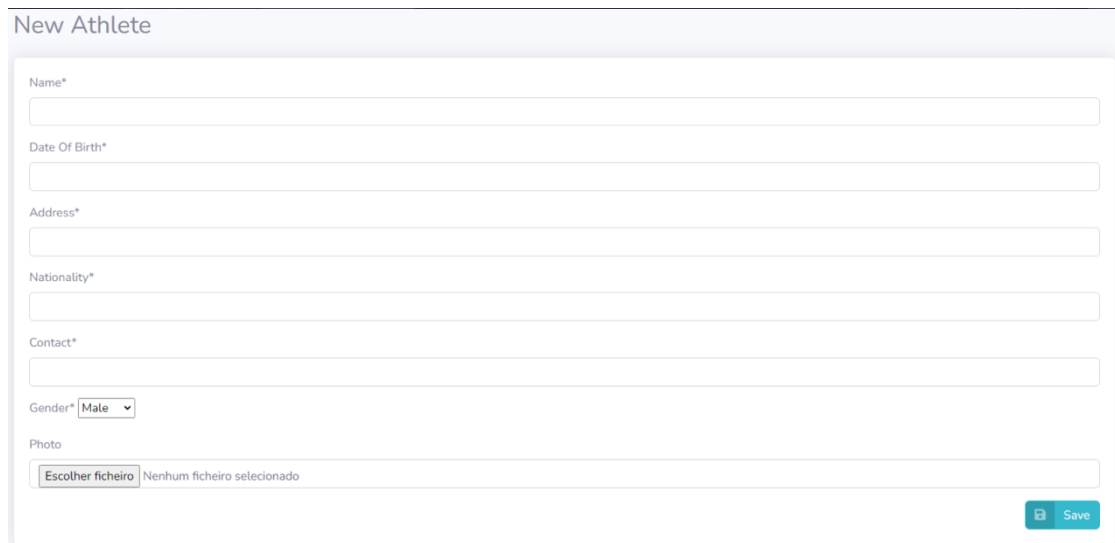
#### Pré-Condição

- Disponibilização do ecrã de login.

#### Critérios de aceitação

- A autenticação é feita através da introdução do Email e Password.
- Se as credenciais estiverem corretas o sistema deve permitir a autenticação.

## RF02 – Adicionar atleta(s).



New Athlete

Name\*

Date Of Birth\*

Address\*

Nationality\*

Contact\*

Gender\* Male ▾

Photo

Escolher ficheiro Nenhum ficheiro selecionado

Save

Fig. 2 – Adicionar atletas

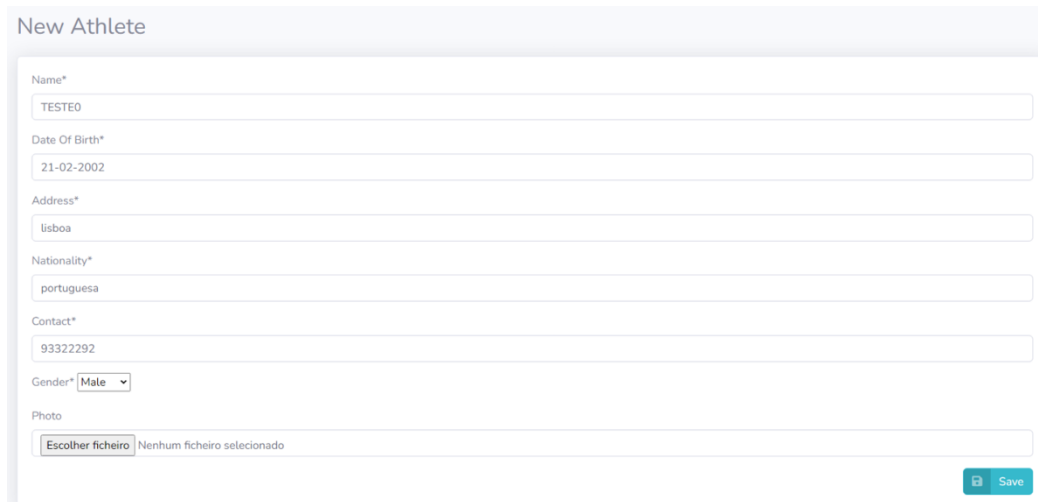
### Pré-Condição

- O utilizador deve estar autenticado (com permissões específicas).

### Critérios de aceitação

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a adição.

### RF03 – Editar dados de atleta(s).



The image shows a web form titled "New Athlete". It contains several input fields and a dropdown menu. The fields are: "Name\*" with the value "TESTEO"; "Date Of Birth\*" with the value "21-02-2002"; "Address\*" with the value "lisboa"; "Nationality\*" with the value "portuguesa"; "Contact\*" with the value "93322292"; "Gender\*" with a dropdown menu showing "Male"; and "Photo" with a file selection button labeled "Escolher ficheiro" and the text "Nenhum ficheiro selecionado". A "Save" button is located at the bottom right of the form.

Fig. 3 – Editar atletas

#### Pré-Condição

- O utilizador deve estar autenticado (com permissões específicas).
- O atleta tem que existir.

#### Critérios de aceitação

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a adição.

## RF04 – Apagar atleta(s)

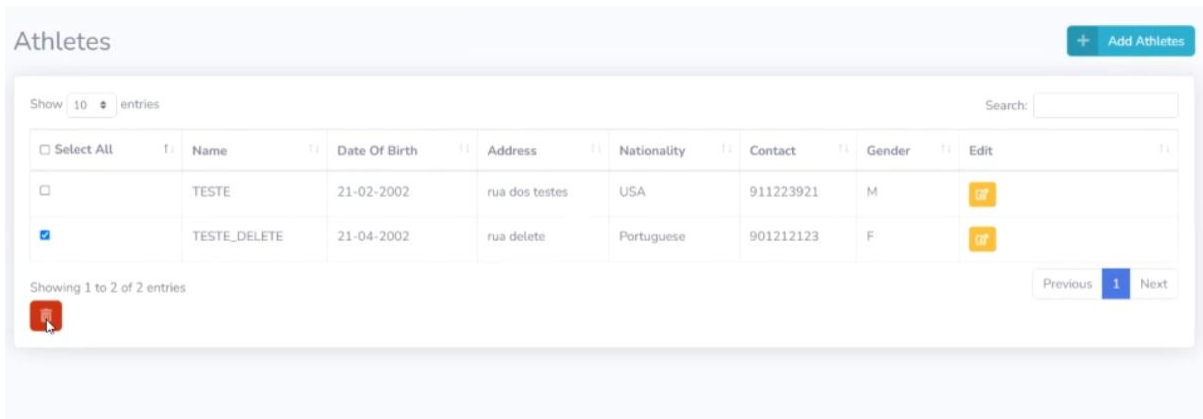


Fig. 4 – Apagar o(s) atleta(s)

### Pré-Condição

- O utilizador deve estar autenticado (com permissões específicas).
- O atleta tem que existir.

### Critérios de aceitação

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a adição.

## RF05 – Adicionar evento/competição

New Event

EventStar\*

Country\*

Address\*

Data start\*

Data end\*

HomologationType\*

Fig. 5 – Adicionar evento

### Pré-Condição

- O utilizador deve estar autenticado (com permissões específicas).

### Critérios de aceitação

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a adição.

## RF06 – Editar evento/competição

New Event

EventStar\* 4 Stars - International Events >4 countries / IWWF Regional Tour Minimum requirements - 10k USD cash prize

Country\*  
USA

Address\*  
rua da star

Data start\*  
03-04-2021

Data end\*  
06-04-2021

HomologationType\* Chief Judge Report

Save

Fig. 6 – Editar evento

### Pré-Condição

- O utilizador deve estar autenticado (com permissões específicas).
- O evento tem que existir.

### Critérios de aceitação

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a edição.



## RF07 – Eliminar evento/competição

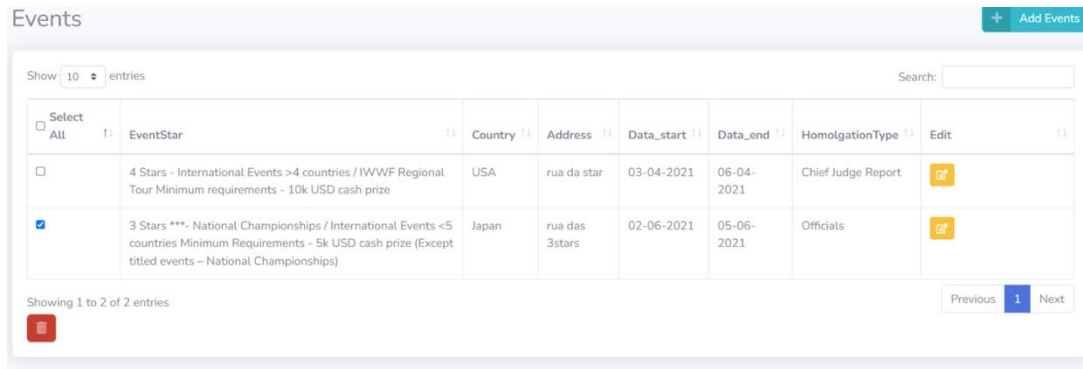


Fig. 7 – Eliminar evento.

### Pré-Condição

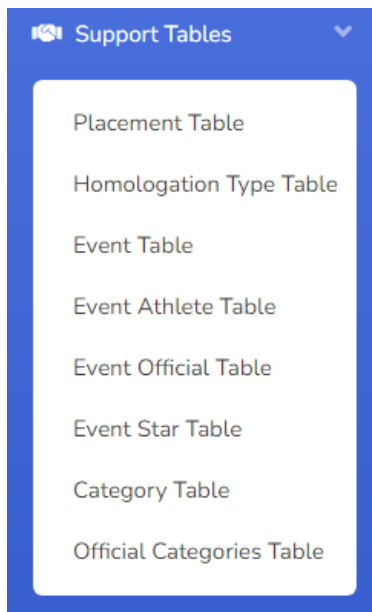
- O utilizador deve estar autenticado (com permissões específicas).
- O evento tem que existir.

### Critérios de aceitação

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a eliminação.

## RF08 – Adicionar elementos a Tabelas de suporte e Tabelas de Entidade

### Tabelas de Suporte:



### Tabelas de Entidades:

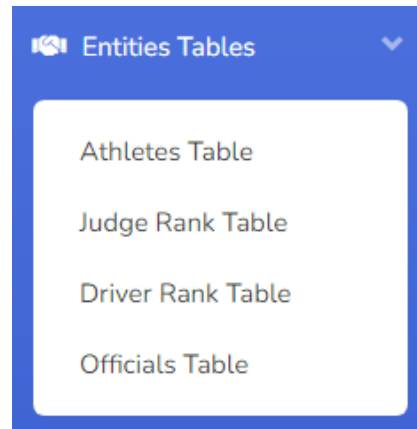


Fig. 8 – As várias tabelas existentes na nossa app.

### Pré-Condição

- O utilizador deve estar autenticado (com permissões específicas).

### Critérios de aceitação

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a adição.

## **RF09 – Editar elementos a Tabelas de suporte e Tabelas de Entidade**

### **Pré-Condição**

- O utilizador deve estar autenticado (com permissões específicas).
- O elemento tem que existir.

### **Critérios de aceitação**

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a edição.

## **RF010 – Eliminar elementos a Tabelas de suporte e Tabelas de Entidade**

### **Pré-Condição**

- O utilizador deve estar autenticado (com permissões específicas).
- O elemento tem que existir.

### **Critérios de aceitação**

- Utilizador autenticado com as devidas permissões.
- O utilizador especifica os campos acima mostrados de forma correta.
- Se as credenciais estiverem corretas o sistema deve permitir a eliminação.

## RF11 – Logout

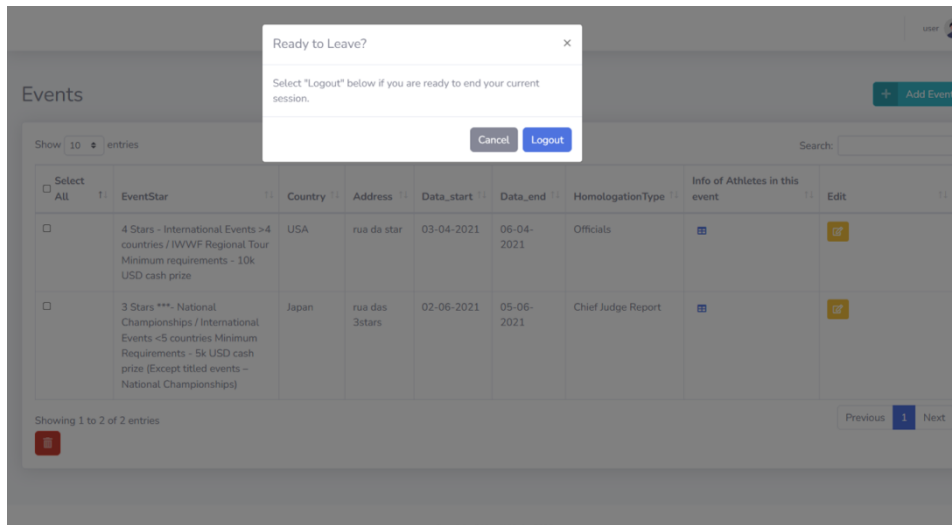


Fig. 9 – Logout do utilizador

### Pré-Condição

- Disponibilização do ecrã com opção de logout.

### Critérios de aceitação

- O utilizador tem que existir.
- O utilizador tem que estar previamente autenticado.

## 2.2 Cumprimento de Requisitos

Integral:

- RFO1
- RFO2
- RFO3
- RFO4
- RFO5
- RFO6
- RFO7
- RFO8
- RFO9
- RF10
- RF11

Modificado:

- RFO8

Na última entrega, ao olharmos para trás conseguimos verificar que fizemos todos os requisitos, desde criar um atleta e evento a saber quem ganhou numa determinada competição.

### **3 Viabilidade e Pertinência**

Tentar acabar com uma necessidade da IWWF, International Waterski & Wakeboard Federation (IWWF) que é o órgão regulador mundial de todos os esportes aquáticos rebocados e tem 90 federações afiliadas em todo o mundo. É reconhecida pelo Comitê Olímpico Internacional (COI) como a única autoridade que rege todos os esportes aquáticos rebocados, também é membro afiliado da Associação das Federações Esportivas Internacionais Reconhecidas do COI (ARISF), da Associação Global da Federação Internacional de Esportes (GAISF) e um dos sete esportes fundadores dos Jogos Mundiais, uma instituição bastante conceituada.

Este software é uma ferramenta flexível, rápida e sem custos de manutenção elevados. Está a ser desenvolvida em Python baseada na framework Django, sendo assim um software moderno/atualizado e é viável devido a isso mesmo, tendo assim possibilidades de ser continuado e usado no futuro.

Os resultados obtidos indicam uma boa abordagem da análise inicial tendo em conta a viabilidade da framework e linguagem utilizada para o desenvolvimento do software.

## 4 Solução Proposta

O primeiro ponto que tivemos em conta para escolher a tecnologia que íamos utilizar era o futuro do software, ou seja, escolher uma framework e uma linguagem moderna que possa ter continuidade num futuro. Escolhemos uma framework Django, utilizada na cadeira de PW (Programação Web), onde nos foi dado algumas “luzes” mas queríamos aprofundar o nosso conhecimento e prática na mesma devido a isso. A linguagem utilizada é Python onde é das linguagens mais utilizadas atualmente e que está em constante crescimento, onde a facilidade de aprendizagem também é um ponto positivo devido a aumentar assim a continuidade da mesma.

### As 5 linguagens de programação mais usadas em 2021

- 1 – Python. Python tem como diferencial a facilidade no aprendizado. ( ...
- 2 – Java. Java é uma das **linguagens** mais utilizadas no mundo. ( ...
- 3 – C++ (e C) C++ e C são duas **linguagens** geralmente aprendidas nas faculdades de tecnologia. ( ...
- 4 – JavaScript. ...
- 5 – Go.

Fig. 10 – Linguagens de programação mais usadas em 2021.

Utilizamos estas tecnologias porque precisamos de uma DataBase onde sejam guardados os dados das entidades e eventos necessários para o funcionamento das provas de Wakeboard. Iríamos usar uma API (rest) para fazer a ligação dos pontos dados, em direto, dos júris e dar upload para um ficheiro .json onde serão retirados os dados para a DataBase, o mesmo vai ser feito para as tabelas das entidades fazendo depois o upload para as nossas tabelas dos dados todos.

Os componentes que requerem mais esforço é a parte de geração de Heats e as consequentes partes técnicas envolvidas referidas na **Identificação do Problema**. Como referido esta framework foi trabalhada em Programação Web.

Ao longo deste projeto preocupamo-nos em cumprir todos os pontos e pedidos propostos pelo cliente fazendo, para cada ponto, testes com o intuito de verificar se estão operacionais e corretos ou não, e do agrado do cliente, todos os testes descritos no capítulo 6 foram feitos, quer pelo orientar quer pelo cliente, e estão operacionais. Sendo que os mais trabalhosos e com maior complexidade foram: encontrar o vencedor de uma competição e criar uma competição.

Ao desenvolver estes dois pontos, tivemos de usar grande parte do que aprendemos em Programação web (PW) e conhecimento adquirido ao longo do

curso em várias cadeiras que nos levam a pensar em soluções práticas e eficazes, o que foi um desafio, visto que a maior parte do que fizemos foi novo, devido ao nível de complexidade do projeto. Na parte mais complexa, geração de competições, usamos a lógica de filtragem seja para obter as heats de cada competição seja para obter os atletas que faziam parte de cada heat. Para cada ronda (Qualfs, LCQ, Semis, Quarters e Final) filtramos por N° de atletas, evento, categoria e ronda da competição para obter as heats das rondas específicas e as suas informações como, quantos atletas passavam para uma próxima fase e para onde é que os mesmos iam ( Semis, Quarters, Final) e quantos atletas passavam apenas para a próxima ronda apenas, por exemplo de Qualfs para LCQ.

14 Riders	1 Heat of 4	1 Heat of 5	1 Heat 5	1 Heat of 6
<b>SF - NS</b>	1 Heat of 5		1 Heat 5	
	1 Heat of 5			
	(Top 3 to Semi)	(LCQ Winner to Semi)	(Top 3 to Final)	
	<b>Total 14 riders</b>	<b>Total 5 riders</b>	<b>Total 10 riders</b>	<b>Total 6 riders</b>

Fig. 11 – Competição com 14 atletas.

Neste exemplo observamos que dos iniciais 14 atletas, 9 passam para os Semis, e os restantes para a LCQ e só depois o vencedor da ronda passa para as Semis, nas Semis de 10 atletas passam 6 para a final tendo apenas 1 vencedor.

```
for ht in Heat.objects.filter(Event=event_obj).filter(Category=category_obj).filter(Round=4).filter(
    AthletesNumb=heat.AthletesNumb):
    listaHeats.append(ht)
```

Fig. 12 – Filtragem de heats na ronda “Semis”

Na parte dos atletas que faziam parte de heats, utilizámos filtragem por Heat e por n° de atletas, isto ordenado pelo parâmetro “score” que traduzia a prestação do atleta em pontos dados por júris. Inicialmente a ideia era ser desenvolvida juntamente com outro TFC, referido anteriormente, onde recebíamos os dados através de uma API mas devido ao facto de trabalho desenvolvido por parte do mesmo não foi possível continuar.

```
for atl in sorted(HeatAthlete.objects.filter(Heat=heat).filter(AthletesNumb=heat.AthletesNumb),
    key=lambda p: (p.score()), reverse=True):
```

Fig. 13– Filtragem de HeatAthletes consoante a Heat



Por vezes tivemos de ordenar os atletas, por ranking points por outras scores, pois se não usássemos estas ferramentas nunca iríamos conseguir chegar ao output do Snake System, nem à distribuição dos atletas pelos heats da forma correta.

Por fim, quando a competição terminasse, é dado o feedback do vencedor da mesma.



Event Heats Final for Open Men

H1

Heat Details

serra is the Winner

Placement	Name	Ranking Points	Score	Set Scores
1	serra	2222	0.10	🏆
2	Katcup	34	0.09	🏆
3	tomé	451	0.08	🏆

Fig. 14 – Mensagem de vencedor

## 5 Benchmarking

Uma comparação à solução que existe é bastante importante, temos aqui vários exemplos de features que existem e como estamos a pensar alterar as mesmas.

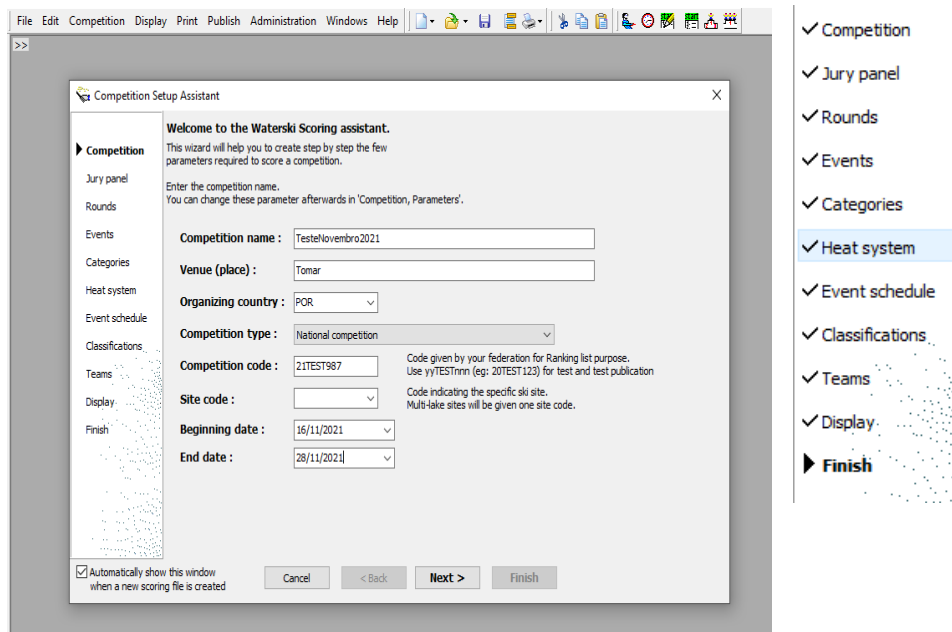


Fig.15 – Criação de um evento no software (No software já existente).

Onde contém muitos dados que são pouco relevantes para o assunto e são necessários bastantes cliques e coisas muito manuais para dar setup a uma competição.

**A nossa solução** é exigir só o necessário para a criação da mesma.

Fig.16 – Criação de um evento no software (Na nossa solução).

A forma de se adicionar atletas à competição (depois da competição estar criada) é pouco confortável, tem que ser 1 a 1 tendo que preencher antes os campos à direita para seleccionar o atleta pretendido, por exemplo:

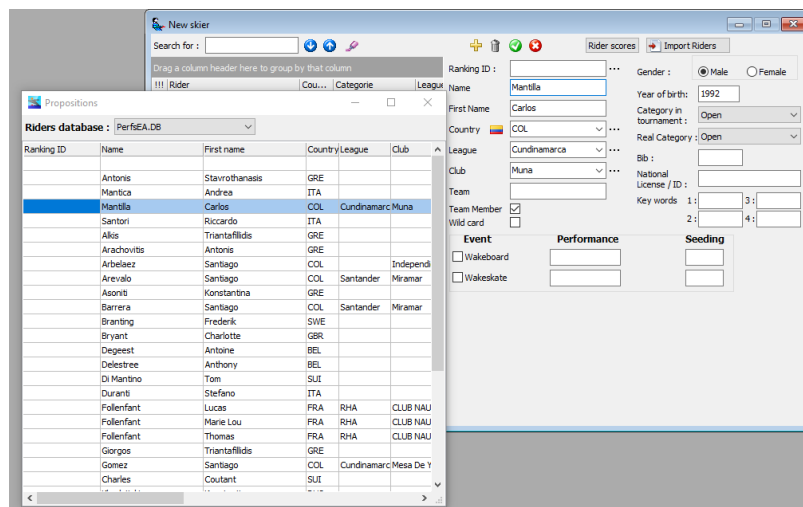


Fig.17 – Adição de atletas a uma competição (No software já existente).

Para uma lista com estes atletas já foi pouco eficiente então para competições de larga escala, por exemplo com 200 atletas, ainda é menos.

!!!	Rider	Cou...	Categorie	League
	Mantilla Carlos	COL	Open Men	Cundin
⚠	Aaboe Jon Simen	NOR	Open Men	
⚠	Allaman Vonny	FRA	Open Men	IDF
⚠	Arachovitis Antonis	GRE	Open Men	
⚠	Arango Pedro	COL	Open Men	Cundin
⚠	Arbelaez Santiago	COL	Open Men	
⚠	Arias Mateo	COL	Open Men	Cundin
⚠	Avenel Arthur	FRA	Open Men	

Fig. 18 – Lista de atletas inscritos (No software já existente).

**A nossa solução** acaba por ser uma inserção mais prática, mais eficiente e confortável. Isto retirávamos da DB dos atletas, inserindo nas tabelas do nosso software e seleccionávamos o atleta apenas tendo que preencher os campos em baixo referidos.

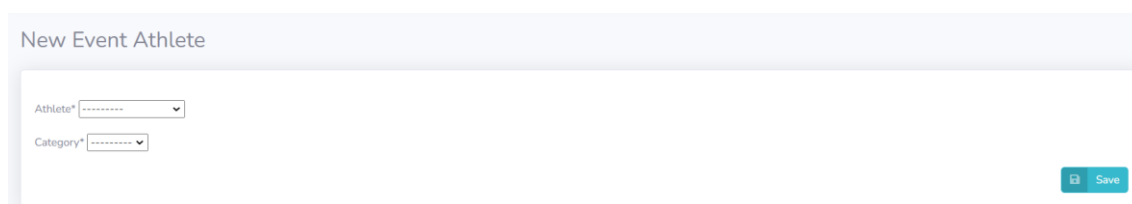


Fig. 19 – Adição de atletas a um evento (Na nossa solução).

Um aspeto importante é a visualização/geração das heats/baterias que é feita desta forma:

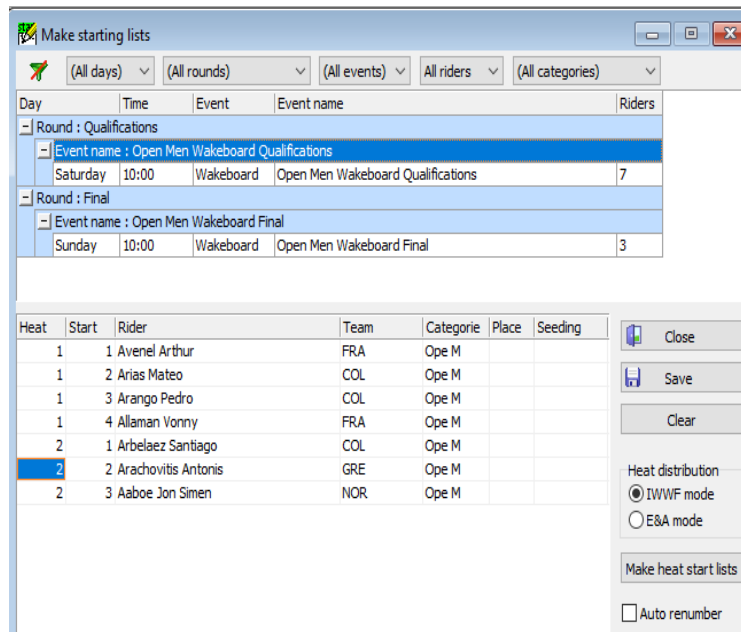


Fig. 20 – Geração de baterias e visualização das mesmas (No software já existente).

Pouco user-friendly e bastante “antiga”.

Outro aspeto é que a **nossa solução** é de uso fácil e toda a gente deve aprender em 10mins e tem uma leitura mais fácil, esta alternativa é bastante complicada e difícil de aprender.

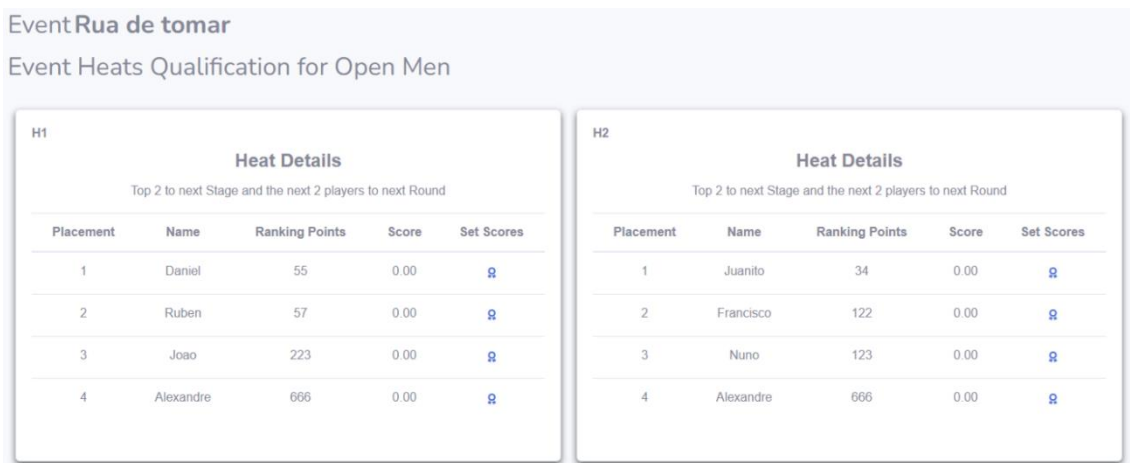


Fig. 21 – Geração de baterias e visualização das mesmas (Na nossa solução).

A adição de pontuação ao atleta também é pouco eficiente, tem que ser desta forma, pouco automática e moderna:

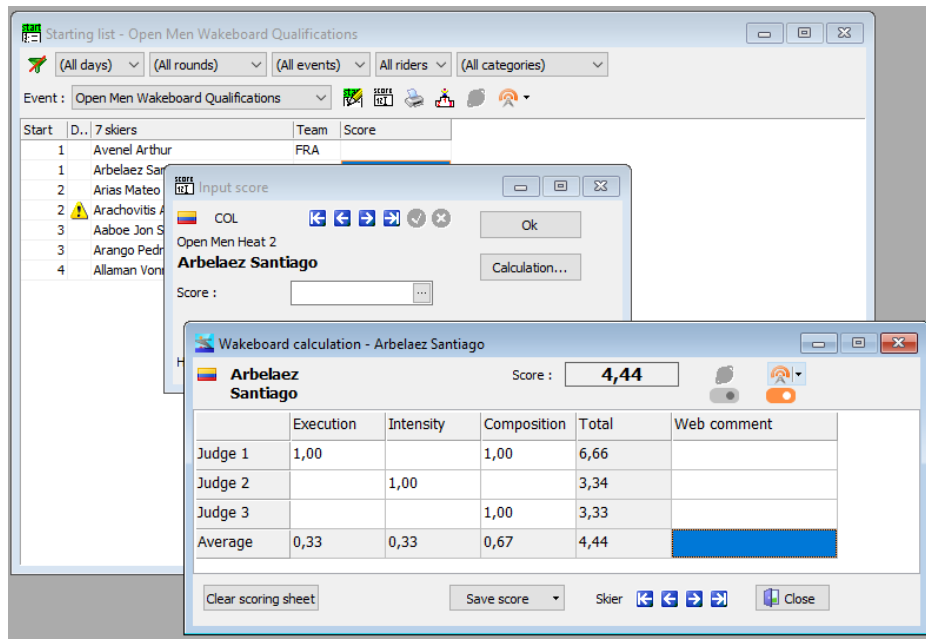


Fig. 22 – Pontuação a ser atribuída (No software já existente).

A **nossa solução** tem como ambição ir buscar os pontos a uma outra plataforma (API) que está a ser desenvolvida por outro TFC do orientador Pedro Serra, onde acontece uma atualização automática dos mesmos pontos de cada atleta.

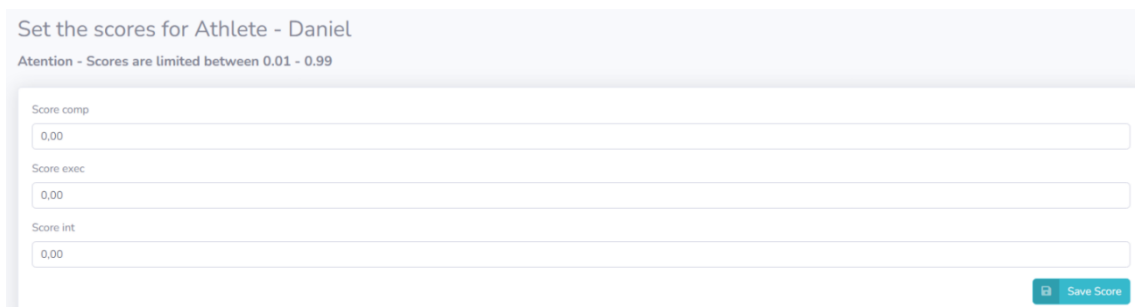


Fig. 23 – Pontuação a ser atribuída (Na nossa solução).

## **6 Método de Planeamento**

Ao longo do desenvolvimento deste projeto, abordamos uma metodologia Agile utilizamos o Gantt onde nós, e através de reuniões com o nosso coordenador, discutimos e planeamos o mesmo para esse semestre, com o Gantt feito começámos a desenvolver os tópicos. Em certos relatórios não conseguimos concluir certos tópicos mostrados no Gantt devido à escassez de tempo, e da complexidade do projeto, mas logicamente conseguindo fazer posteriormente esses mesmos tópicos. Neste momento conseguimos por em prática todos os tópicos, assim como fizemos testes relativamente à criação de uma competição e descobrir o vencedor dessa mesma.

## 7 Resultados

Este projeto tinha a finalidade de desenvolver uma aplicação mais sofisticada e simples da já existente, onde fosse possível criar: atletas, eventos e competições para os atletas de Wakeboard. Visualizando os requisitos feitos na secção 2, podemos verificar que a maior parte foram cumpridos e integrados no nosso projeto.

O centro deste projeto está na criação de competições, heats, adicionar scores aos atletas dessa competição e no final verificar quem foi o vencedor.

Vamos deduzir que criamos uma competição com 5 atletas de uma determinada categoria como na seguinte figura:


Category	Number of Athletes	HeatSystem	Generate Heat
Boys	0	No Event	Not enough players
Girls	0	No Event	Not enough players
Junior Men	0	No Event	Not enough players
Junior Women	0	No Event	Not enough players
Master Men	0	No Event	Not enough players
Master Women	0	No Event	Not enough players
Veteran Men	0	No Event	Not enough players
Veteran Women	0	No Event	Not enough players
Open Men	5	Qualf-Finals	
Open Women	0	No Event	Not enough players

Fig. 24 – Atletas de uma determinada competição.

Caso o número de atletas seja superior a 5, é permitido criar uma competição caso seja menor é mostrado uma mensagem de que não há atletas suficientes (“Not enough players”). Ao clicar no botão somos redirecionados para uma página onde estão todos os atletas dessa competição, utilizando o exemplo dos 5 atletas verificamos que os atletas estão distribuídos de forma correta, onde o que tem mais “Ranking Points” será o último.



Event Tomar  
Event Heats Qualification for Open Men

H1

**Heat Details**  
Top 5 to next Stage

Placement	Name	Ranking Points	Score	Set Scores
1	Daniel	55	0.00	⚙
2	Ruben	57	0.00	⚙
3	Francisco	122	0.00	⚙
4	Nuno	123	0.00	⚙
5	Joao	223	0.00	⚙

Fig. 25 – Competição de 5 atletas

Introduzimos os scores de cada atleta, com os scores todos introduzidos é permitido passar para a próxima ronda, neste momento os atletas estão ordenados pela média dos scores de cada atleta, como este Heat só tem 5 atletas do Qualifying passamos diretamente para a final.



Event Tomar  
Event Heats Qualification for Open Men

[Continue to Final](#)

H1

**Heat Details**  
Top 5 to next Stage

Placement	Name	Ranking Points	Score	Set Scores
1	Nuno	123	0.51	⚙
2	Joao	223	0.17	⚙
3	Francisco	122	0.11	⚙
4	Ruben	57	0.07	⚙
5	Daniel	55	0.03	⚙

Fig. 26 – Competição com os atletas ordenados por scores



Ao irmos para a final todos os atletas começam com os scores a 0.00 (Fig. 27), e no final dos scores estarem preenchidos para cada atleta é mostrado uma mensagem de quem foi o vencedor (Fig. 28).

Event **Tomar**  
Event Heats Final for Open Men

H1

**Heat Details**  
**Daniel is Winning**

Placement	Name	Ranking Points	Score	Set Scores
1	Daniel	55	0.00	<a href="#">🏆</a>
2	Ruben	57	0.00	<a href="#">🏆</a>
3	Francisco	122	0.00	<a href="#">🏆</a>
4	Nuno	123	0.00	<a href="#">🏆</a>
5	Joao	223	0.00	<a href="#">🏆</a>

Fig. 27 – Atletas sem scores.

Event **Tomar**  
Event Heats Final for Open Men

H1

**Heat Details**  
**Nuno is the Winner**

Placement	Name	Ranking Points	Score	Set Scores
1	Nuno	123	0.16	<a href="#">🏆</a>
2	Joao	223	0.16	<a href="#">🏆</a>
3	Francisco	122	0.13	<a href="#">🏆</a>
4	Ruben	57	0.08	<a href="#">🏆</a>
5	Daniel	55	0.05	<a href="#">🏆</a>

Fig. 28 – Atletas com os scores, e mensagem de quem ganhou

Para demonstrar o nosso algoritmo a funcionar (Snake System), vamos demonstrar uma competição com 7 atletas.

Com 7 atletas serão criadas 2 heats uma com 4 atletas e outra com 3 atletas. Onde podemos verificar que os atletas estão dispostos em Snake System (tabela 1)

Event **Rua de tomar**  
Event Heats Qualification for Open Men

H1 Heat Details				
Top 2 to next Stage and the next 2 players to next Round				
Placement	Name	Ranking Points	Score	Set Scores
1	Juanito	34	0.00	🏆
2	Daniel	55	0.00	🏆
3	Nuno	123	0.00	🏆
4	Joao	223	0.00	🏆

H2 Heat Details				
Top 2 to next Stage and the next 1 players to next Round				
Placement	Name	Ranking Points	Score	Set Scores
1	Ruben	57	0.00	🏆
2	Francisco	122	0.00	🏆
3	Alexandre	666	0.00	🏆

Fig. 29 – Competição com 7 atletas

## 7.1 Plano de testes e validação

Plano de testes a realizar por parte, e com o cliente, numa data mais tarde marcada com o devido conhecimento do mesmo.

### 1. T01

Criar um atleta e verificar se esse atleta fica guardado na base de dados

- Abrir a página de Athletes
- Carregar no botão de Add Athlete
- Criar o atleta, preenchendo todas as informações necessárias
- Guardar os dados
- Verificar na lista dos Atletas se o atleta criado anteriormente, se encontra lá.

### 2. T02

Criar um Evento

- Abrir a página dos Events
- Carregar no botão de Add Event
- Criar o Evento, preenchendo com os dados necessários para a sua criação
- Guardar os dados
- Verificar na lista dos Eventos se o evento criado anteriormente, se encontra lá.

### 3. T03

Criar um Oficial

- Abrir a página dos Officials
- Carregar no botão de Add Official
- Criar o Oficial, preenchendo com os dados necessários para a sua criação
- Guardar os dados
- Verificar na lista de oficiais se o evento criado anteriormente, se encontra lá.

### 4. T04

Adicionar um atleta a um Evento/Competição

- Verificar se o atleta está criado
- Verificar se o evento está criado
- Ir para a página dos Events
- Ir aos detalhes do Evento que deseja adicionar o atleta
- Dentro do Evento ir às informações dos atletas (Athletes), e carregar no botão add Event Athlete
- Escolher o atleta (Athlete), que criou inicialmente, escolhendo a categoria e o placement
- Guardar o atleta
- Verificar se o atleta escolhido, está na lista de atletas do Evento

5. T05

Editar um atleta

- Carregar na página dos athletes
- Carregar no botão editar (botão amarelo à direita)
- Editar a informação que pretende alterar desse atleta
- Guardar essas alterações
- Verificar se as alterações foram guardadas

6. T06

Eliminar um atleta

- Carregar na página dos athletes
- Selecionar o atleta que pretende eliminar
- Carregar no botão eliminar (botão vermelho)
- Verificar se o atleta foi eliminado

7. T07

Editar um Evento

- Carregar na página dos Events
- Carregar no botão editar (botão amarelo à direita)
- Editar a informação que pretende alterar desse Evento
- Guardar essas alterações
- Verificar se as alterações foram guardadas

8. T08

Eliminar um Evento

- Carregar na página dos Events
- Selecionar o Evento que pretende eliminar
- Carregar no botão eliminar (botão vermelho)
- Verificar se o Evento foi eliminado

9. T09

Visualizar a lista de atletas de uma categoria

- Carregar na página dos Events
- Dentro do Evento ir às informações das Categorias (Categories)
- Verificar se a categoria escolhida tem mais que um atleta
- Carregar no botão com uma seta, para visualizar mais detalhes dessa categoria
- Visualizar a lista de atletas dessa categoria

10. T10

Criação de uma competição

- Carregar na página dos Events
- Dentro do Evento ir às informações dos Heats
- Verificar se a Categoria pretendida está disponível para ter competição associada, sendo a verificação ter mais que três atletas
- Carregar no botão com uma seta, visualizar gerar Heat da categoria pretendida
- Visualizar a competição com os atletas dessa categoria

11. T11

Encontrar o Vencedor de uma competição

- Carregar na página dos Events
- Dentro do Evento ir às informações dos Heats
- Verificar se a Categoria pretendida está disponível para ter competição associada, sendo a verificação ter mais que três atletas
- Carregar no botão com uma seta, visualizar gerar Heat da categoria pretendida
- Visualizar a Heat com os atletas dessa categoria
- Adicionar os Scores a todos os atletas
- Dependendo do número de atletas nessa Heat, passar pelas várias rondas até chegar à final
- Na final podemos verificar a informação de quem está a ganhar
- Quando introduzimos os scores a todos os atletas, é mostrado uma mensagem de quem ganhou a competição

Fizemos alguns Test Cases, de alguns requisitos, onde o nosso coordenador (Pedro Serra), e o nosso cliente (Nuno Eça), pediram para fazer testes aos requisitos no nosso projeto, passando a todos os propostos.

Aqui estão os testes feitos aos requisitos do Test Case:

	Orientador	Cliente
T01	✓	✓
T02	✓	✓
T03	✓	✓
T04	✓	✓
T05	✓	✓
T06	✓	✓
T07	✓	✓
T08	✓	✓
T09	✓	✓
T10	✓	✓
T11	✓	✓

## **8 Conclusão e trabalhos futuros**

Concluimos que apesar de difícil e muito complexo este trabalho só nos trouxe qualidades como, mais aptidão para desenvolver problemas complexos, mais autonomia, mais criatividade para soluções, mais maturidade a trabalhar em equipa com prazos e deadlines envolvidas e em comunicar com a equipa. Com o desenvolvimento deste trabalho, encontramos nos mais aptos para a realização de trabalhos que requerem mais complexidade, com mais facilidade em arranjar soluções criativas que resultem e a trabalhar em equipa.

## **Bibliografia**

[DEISI21] DEISI, Regulamento de Trabalho Final de Curso, Set. 2021.

[ULHT21] Universidade Lusófona de Humanidades e Tecnologia, [www.ulusofona.pt](http://www.ulusofona.pt),  
acedido em Out. 2021.

[Linguagens mais usadas em 2021](#)

[Competições da IWWF](#)

[Website da IWWF](#)

[Regulamento da competição de Wakeboard, produzido pela IWWF](#)