

# Arquitetura de Software

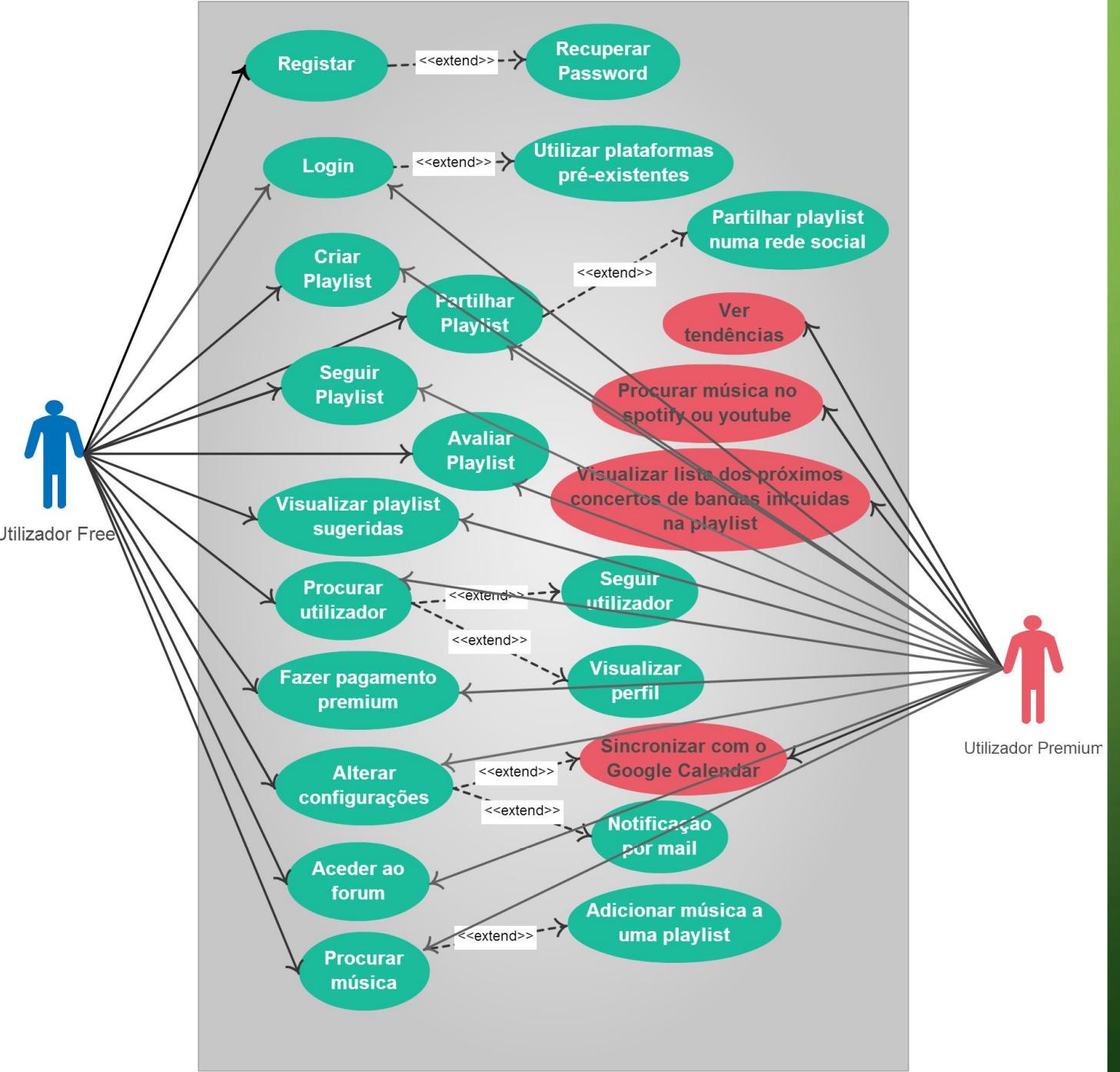
Bruno Madureira, Fábio Moura e Nuno Miranda

# Introdução

Este trabalho tem como objectivo...

# Drivers Arquitecturais Requisitos Funcionais

Sistema



# Descrição da Arquitetura

A arquitetura proposta baseia-se num típico modelo MVC (Model-view-controller) mas com algumas alterações para conseguirem suportar com as necessidades impostas pelas restrições e atributos de qualidade. O sistema é composto por 3 tipos de entidades: **Load balancer**, **Web Server** e **Bases de dados**.

As nossas alterações ao modelo MVC foram a inclusão de um load balancer que irá permitir ter vários servidores a fazer alojamento do serviço mantendo o mesmo domínio. Isto irá permitir também uma fácil e rápida adição de novos servidores sem ter de alterar grandes quantidades de código.

# Descrição da Arquitetura

## Restrições arquiteturais de negócio

### 1) Ambientes suportados

Página web apenas inclui suporte para desktop, excluindo suporte para todas as plataformas móveis, visto que o número de utilizadores do nosso público-alvo é bastante superior a nível de computador e o custo de migrar a plataforma para mobile não justifica o investimento.

# Descrição da Arquitetura

## Restrições arquiteturais de negócio

### 2) Limitações para utilizadores free

Nos utilizadores free será limitado o número de playlists (para 10 playlists) e o número de músicas por playlist (30 músicas). Nos utilizadores premium estes parâmetros serão ilimitados. Desta forma será promovida a adesão ao modelo premium.

# Descrição da Arquitetura

## Restrições arquiteturais de negócio

### 3) Publicidade na página web

Nos utilizadores free irá ser mostrada publicidade para assegurar a rentabilidade do negócio.

# Descrição da Arquitetura

## Restrições arquiteturais de negócio

### 4) Escalabilidade

A arquitetura deve garantir escalabilidade, isto é, deverá ter particionamento de servidores e dados.

# Descrição da Arquitetura

## Restrições arquiteturais de negócio

### 5) Dados de armazenamento de músicas

As músicas não iram estar guardadas na nossa base de dados pois isso iriam implicar contratos e custos extra. Além disso não está no intuito do nosso negócio ser um fornecedor de música mas apenas um serviço de partilha de playlists e discussão musical. No entanto utilizaremos a API do spotify e do youtube para facilitar o acesso das músicas que o utilizador queira ouvir (nesses serviços). Os dados que iram ser guardados sobre as músicas é apenas o nome das mesmas assim como informação adicional (ano de lançamento, autor, álbum, informação de vendas, etc.).

# Descrição da Arquitetura

## Restrições arquiteturais técnicas

### 1) Construção da página web

Página web será otimizada apenas para os browsers Firefox (a partir da versão 38.0.1) e Google Chrome (a partir da versão 43.0).

# Descrição da Arquitetura

## Restrições arquiteturais técnicas

### 2) Autenticação com contas externas

O Sistema de autenticação a partir das APIs das redes sociais será feita apenas através do twitter, Google+ e facebook.

# Descrição da Arquitetura

## Restrições arquiteturais técnicas

### 3) Base de dados de músicas

O Sistema vai usar API do Spotify e do Youtube para pesquisa de músicas, guardando em base dados apenas o nome das mesmas. Se a música não se encontrar em ambos os serviços o próprio utilizador poderá inseri-la no sistema, enriquecendo assim a base de dados atual.

# Descrição da Arquitetura

## Restrições arquiteturais técnicas

### 4) Ouvir música

O sistema vai usar a API do Spotify e do Youtube para reproduzir a música.

# Descrição da Arquitetura

## Restrições arquiteturais técnicas

### 5) Pagamentos

Os pagamentos serão efetuados apenas por paypal.

# Descrição da Arquitetura

## Restrições arquiteturais técnicas

### 6) Partilha de playlist

A partilha de playlists será assegurada através das redes sociais Facebook, twitter e google+ apenas.

# Descrição da Arquitetura

## Atributos de Qualidade - Sistema facilmente modificável

TABELA XXV. SISTEMA FACILMENTE MODIFICÁVEL

Fonte de estímulo	Ambiente de negócio
Estímulo	Mudanças no negócio ditam mudanças arquiteturais
Ambiente	Desenvolvimento e produção
Componente estimulado	Sistema
Resposta	Custo de mudança arquitetural
Medida de resposta	Custo total das mudanças anuais deverá ser no máximo 30% da receita líquida anual.

# Descrição da Arquitetura

## Atributos de Qualidade - Disponibilidade do Sistema

TABELA XXVIII. DISPONIBILIDADE DO SISTEMA 1

Fonte de estímulo	Utilizador
Estímulo	Utilizador tenta aceder ao sistema
Ambiente	Produção
Componente estimulado	Sistema
Resposta	Percentagem de tempo em que o sistema está ativo durante o ano
Medida de resposta	Certificar que a percentagem de disponibilidade é igual ou superior a 99,9%

TABELA XXIX. DISPONIBILIDADE DO SISTEMA 2

Fonte de estímulo	Utilizador maligno
Estímulo	Utilizador tenta atacar o sistema usando SQL injection
Ambiente	Produção
Componente estimulado	Sistema
Resposta	Número de ataques efetuados com sucesso
Medida de resposta	Número de ataques graves efetuados com sucesso deverá ser zero e o número de ataques pouco graves efetuados com sucesso deverá ser inferior a 5 por ano.

TABELA XXX. DISPONIBILIDADE DO SISTEMA 3

Fonte de estímulo	Máquina onde está alojado o servidor ou a base de dados
Estímulo	Máquina falha
Ambiente	Produção
Componente estimulado	Sistema
Resposta	Tempo que demorará ao serviço passar a ser fornecido por outra máquina
Medida de resposta	O tempo deverá ser inferior a 5 segundos (mudança deve ser automática)

TABELA XXXI. DISPONIBILIDADE DO SISTEMA 4

Fonte de estímulo	Máquina onde está alojado o servidor ou a base de dados
Estímulo	Máquina falha
Ambiente	Produção
Componente estimulado	Sistema
Resposta	Tempo que a máquina demorará a ser substituída ou reparada
Medida de resposta	A máquina deve ser substituída idealmente em menos de uma hora, no entanto é tolerável que o tempo de reparação seja de até 24 horas.

# Descrição da Arquitetura

## Atributos de Qualidade - Performance

TABELA XXXII. PERFORMANCE 1

<b>Fonte de estímulo</b>	<i>Browser</i>
<b>Estímulo</b>	<i>Browser</i> recebe um pedido e envia nova página ao utilizador
<b>Ambiente</b>	Produção
<b>Componente estimulado</b>	Sistema
<b>Resposta</b>	Tempo de resposta
<b>Medida de resposta</b>	Sistema deve enviar nova página em menos de 1 segundo

TABELA XXXIII. PERFORMANCE 2

<b>Fonte de estímulo</b>	Utilizador
<b>Estímulo</b>	Utilizador faz pesquisa na base de dados de músicas
<b>Ambiente</b>	Produção
<b>Componente estimulado</b>	Sistema
<b>Resposta</b>	Tempo de resposta
<b>Medida de resposta</b>	Resultados devem ser exibidos em menos de 1.5 segundos

# Descrição da Arquitetura

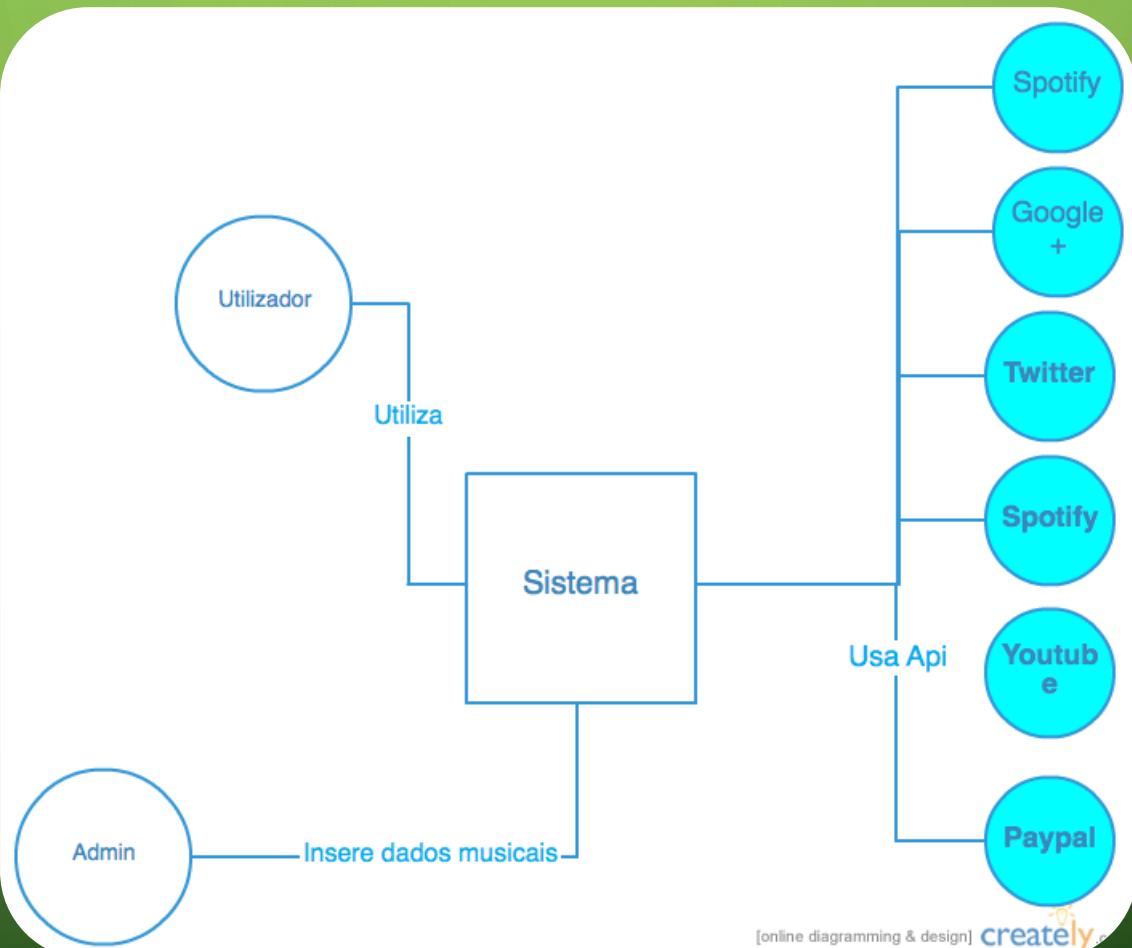
## Atributos de Qualidade - Sistema escalável

TABELA XXXIV. SISTEMA ESCALÁVEL

Fonte de estímulo	Sistema
Estímulo	Sistema recebe um aumento exponencial de pedidos
Ambiente	Produção
Componente estimulado	Sistema
Resposta	Tempo de resposta a cada um dos pedidos
Medida de resposta	Sistema deve responder em menos de um segundo

# Descrição da Arquitetura

## Vistas - Digrama de contexto



[online diagramming & design] **creately**

# Descrição da Arquitetura

## Vistas - Digrama de contexto

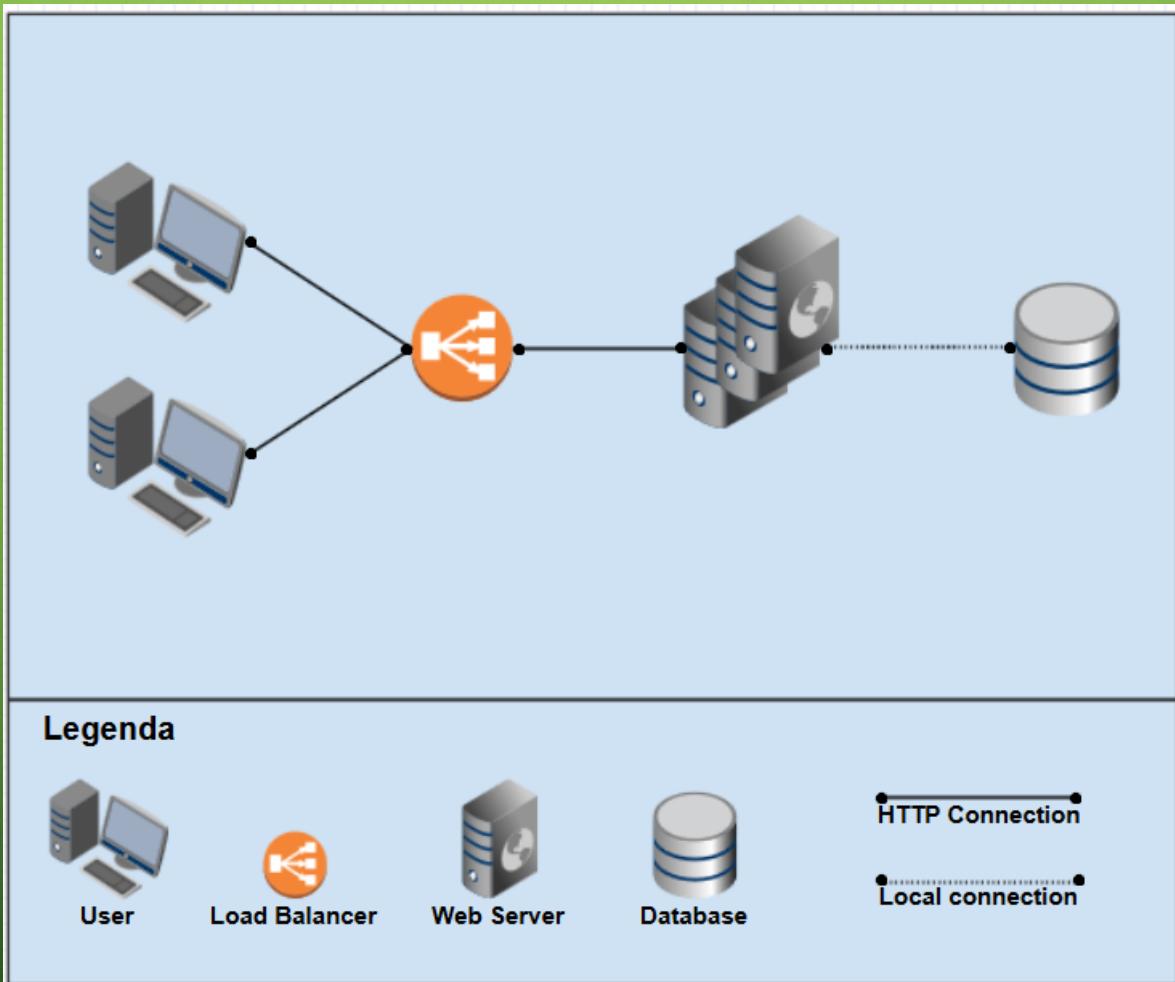
Este diagrama mostra o contexto do nosso sistema no negócio. O sistema terá ligação a várias entidades externas através das suas Apis. Será feita a comunicação também com um grupo de APIs externas (através dos servidores web):

- **Facebook, Google+ e Twitter:** será utilizado para fazer autenticação no sistema e partilhar playlists na plataforma;
- **Spotify e youtube:** será utilizado para aceder às músicas que o utilizador pretenda (que esteja numa playlist no sistema);
- **Paypal:** será utilizado para realizar pagamentos.

É responsabilidade do administrador adicionar informação de músicas.

# Descrição da Arquitetura

## Vistas - Digrama físico



# Descrição da Arquitetura

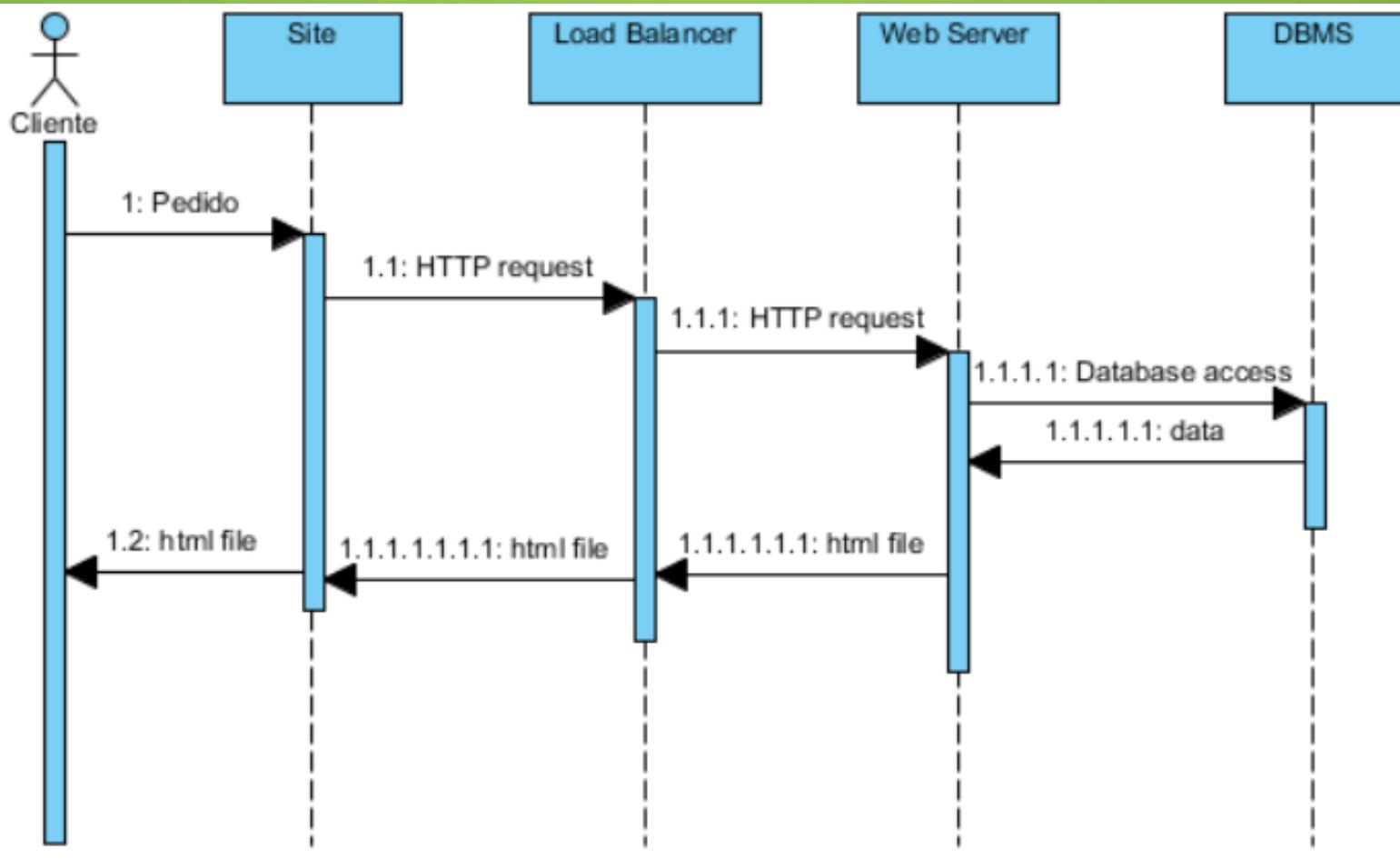
## Vistas - Digrama físico

Como foi referido anteriormente, vamos então ter um Load Balancer que irá encaminhar os pedidos que estão a chegar para o servidor web que estiver com menor carga, isto é, a processar menos pedidos. É de notar que o próprio load balancer irá estar replicado para o caso deste falhar.

Depois o servidor web vai comunicar com a base de dados (caso seja necessário) e posteriormente comunica com o load balancer para devolver o pedido. De notar que não existe qualquer comunicação entre os servidores web e têm de existir pelo menos dois servidores web para garantir o mínimo de tolerância a falhas.

# Descrição da Arquitetura

## Vistas - Digrama de sequência



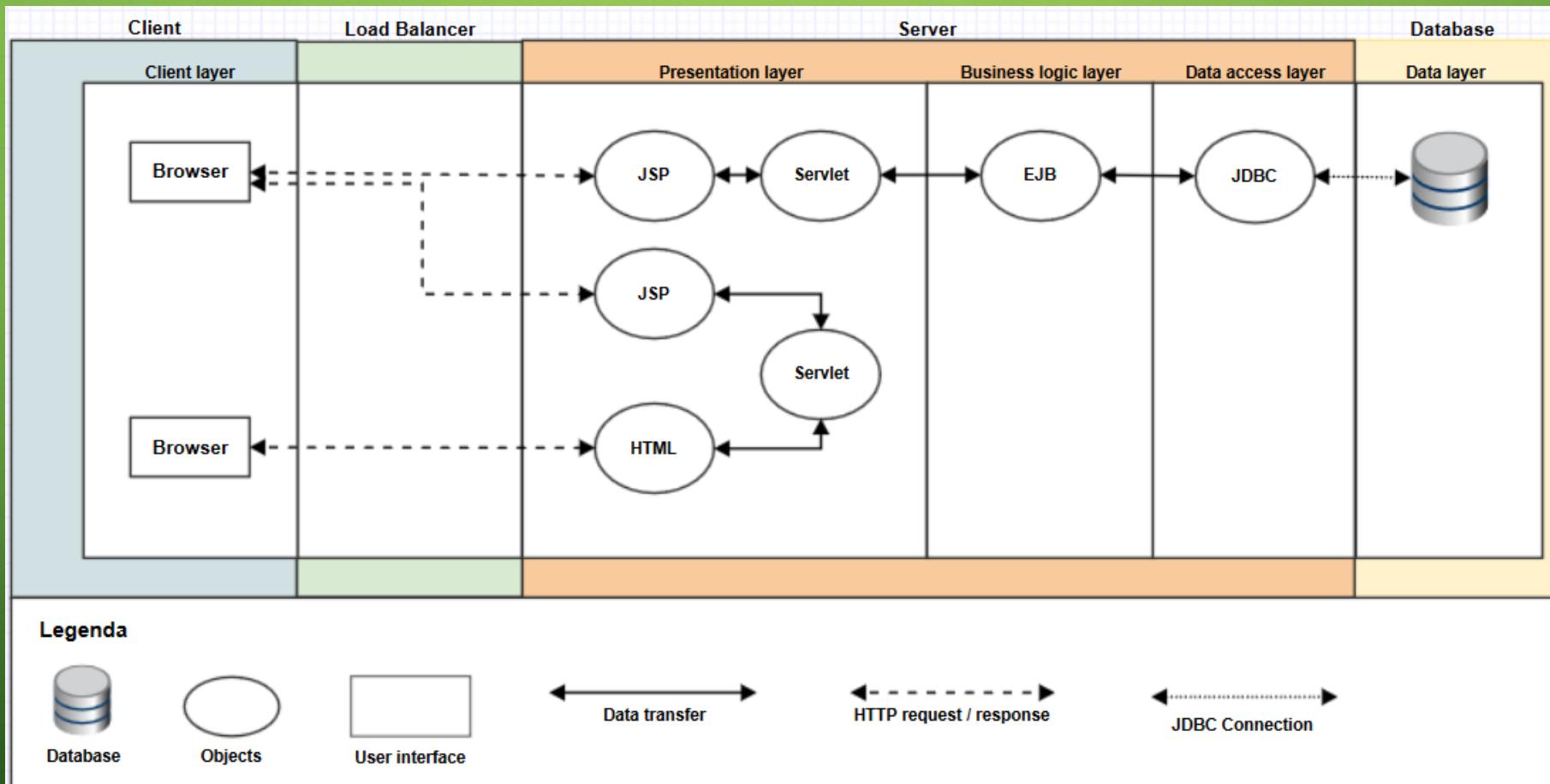
# Descrição da Arquitetura

## Vistas - Digrama de sequência

Este diagrama mostra o fluxo de informação no sistema. Para isso utilizados um cenário genérico em que um utilizador faz um pedido à aplicação. O pedido é transmitido ao load balancer que irá redirecionar o pedido ao servidor com menor carga. Caso seja necessário usar informação da base de dados o servidor web ligar-se-á a esta trocando informação. De seguida o servidor web gera uma resposta ao pedido do utilizador no formato de um documento html, este documento é enviado ao load balancer que o reencaminhará ao utilizador que fez o pedido.

# Descrição da Arquitetura

## Vistas - Digrama de desenvolvimento



# Descrição da Arquitetura

## Vistas - Digrama de desenvolvimento

Este diagrama mostra a perspetiva do programador. Nesta vista supomos que o programa será implementado em java, não sendo isto obrigatório nem prescrito pela arquitetura. Podemos ver as 5 camadas principais, a camada do load balancer , a presentation layer, a business logic, a data access layer e da data layer. É possível observar o tipo de conexão entre os vários tipos de objetos.

# Decisões Chave

## Decisões arquiteturais

### 1) Bases de dados

- Num estado inicial do projeto estávamos a pensar em utilizar duas bases de dados;
- Será usada uma base de dados SQL clássica garantindo assim consistência e atomicidade, não podendo garantir particionamento. A tolerância a falhas será garantida através de diversos discos ligados através do sistema de RAID-10.

# Decisões Chave

## Decisões arquiteturais

### 2) Segurança

- As passwords não deverão ser gravadas em plain-text;
- Deverá ser usado um método baseado em chaves simétricas para cifrar o pedido de login e registo;
- O sistema deverá estar protegido contra ataques de SQL injection.
- A segurança do sistema de pagamento pelo paypal está garantida pela API do mesmo;
  - A disponibilidade e a segurança são muitas vezes atributos de qualidade rivais. No entanto, no nosso caso os procedimentos de segurança implementados não irão comprometer a disponibilidade do sistema.

# Decisões Chave

## Decisões arquiteturais

### 3) Disponibilidade do sistema

Para garantir este atributo de qualidade, iremos ter diversos servidores web (como irá ser explicado em detalhe na próxima secção) de preferência em diferentes localidades geográficas para evitar que situações como cortes de eletricidade ou mesmo catástrofes naturais).



# Decisões Chave

## Decisões arquiteturais

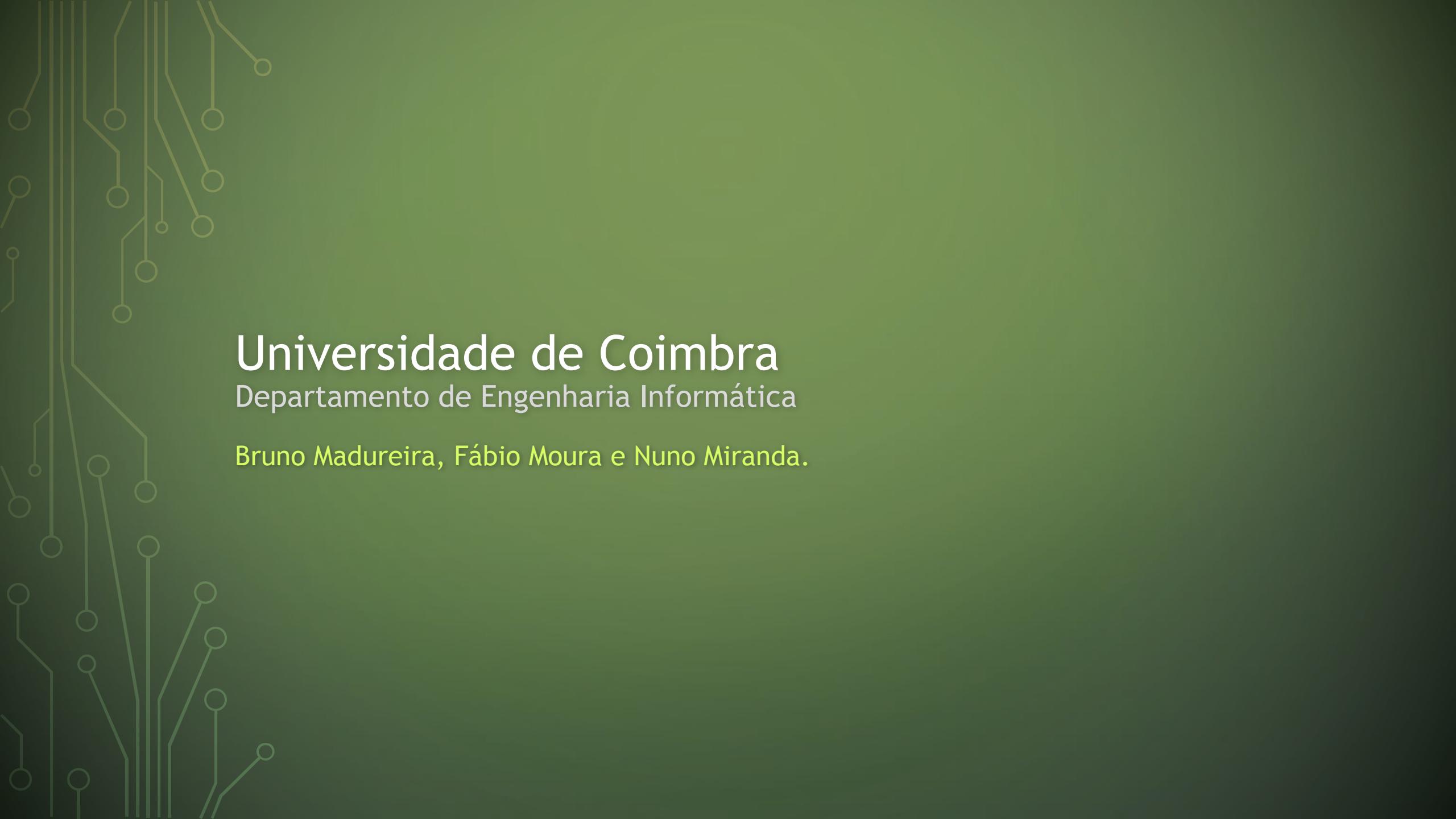
### 4) Interoperabilidade

O tempo de resposta a pedido de login através de contas externas estará sempre sujeito à disponibilidade dos servidores dessas mesmas aplicações.

# Decisões Chave

Análise dos atributos de qualidade

- 1) Sistema facilmente modificável e modular
- 2) Segurança
- 3) Disponibilidade do sistema
- 4) Sistema escalável e performance



# Universidade de Coimbra

Departamento de Engenharia Informática

Bruno Madureira, Fábio Moura e Nuno Miranda.