Universidade do Minho
Escola de Engenharia

Requisitos e Arquiteturas de Software

MEI - 1º Ano - 1º Semestre
Universidade do Minho

Trabalho Prático II
RASBet

Bárbara Ferreira Teixeira PG47038
Carlos Miguel Luzia de Carvalho PG47092
João Pedro da Santa Guedes PG47329
Paulo Silva Sousa PG47556
Rui Emanuel Gomes Vieira PG47635


PG47308


PG47092


PG47329


PG47556


PG47635

19 de novembro de 2021

Resumo

Neste segunda fase do trabalho da Unidade Curricular de Requisitos e Arquiteturas de Software foi-nos proposto desenvolver uma solução para a arquitetura do nosso produto criado na primeira fase.

Este produto tem como designação RASBet e consiste num aplicação *web based* de apostas online, estas apostas debruçar-se-ão em três desportos sendo estes futebol, ténis e fórmula1.

A solução do produto tem presentes as soluções arquitectónicas que encontramos numa perspectiva de fazer cumprir os requisitos enunciados na primeira fase e a lista de funcionalidades que mais tarde serão implementadas usando esta arquitectura. Para auxiliar a compreensão destas funcionalidades foram usados diversos diagramas UML que expõem os diferentes aspetos da solução criada de forma precisa e simples.

Juntando esta solução com a implementação da mesma iremos convergir para um produto de software bem estruturado e de excelente qualidade.

Índice

Lista de Figuras

Lista de Tabelas

I Introdução e Objetivos

Nesta segunda etapa do nosso projeto, vamos abordar a arquitectura do mesmo, numa fase anterior à sua efetiva construção, assim, para esta fase, tratando-se a fase referente à arquitectura do projeto, é necessário considerar não só os requisitos funcionais e não funcionais, explicitados na fase anterior, como também ter em conta os nossos objetivos e motivações subjacentes ao projeto, assim como todos os recursos necessários fundamentais ao desenvolvimento da arquitectura, é também ainda necessário considerar os nossos stakeholders e as suas expectativas.

1 Visão Geral dos Requisitos

Como podemos ver na fase anterior apresentamos as restrições às quais o nosso projeto tem de responder, e assim definimos os requisitos funcionais e não funcionais que este tinha de cumprir (este documento pode ser encontrado na página blackboard do nosso grupo, disponível em Anexos).

Numa prespetiva de criar uma aplicação de apostas que sobressaia em relação a todas as outras já existentes, temos como motivação criar um serviço de apostas cujo uso na perspetiva do utilizador seja fácil, dinâmico, justo, seguro e user friendly e, dessa forma, poder mesmo até gerar todo um novo interesse nos desportos que a aplicação suporta, por parte do público. Dentro deste Sistema existirão então 3 tipos de utilizadores, sendo estes os apostadores (utilizadores registados), utilizadores (visitantes do sistema) e ainda os administradores do mesmo.

Como já referido na fase anterior, chegamos a encontrar diversas restrições a ser impostas neste projeto assim como os variados requisitos que o sistema tem de cumprir sendo estes:

1. Registo de Utilizador
2. Autenticação no Sistema
3. Edição de Perfil do Utilizador
4. Consulta de desportos
5. Consulta de Eventos
6. Realizar Apostas
7. Consulta de Saldo
8. Carregamento de saldo
9. Levantamento de saldo
10. Consulta de histórico de saldo

Tendo dado uma vista geral nos requisitos funcionais desenvolvidos na fase anterior numa perspectiva de auxiliar a compreensão, vamos de seguida demonstrar o desenvolvimento de um dos requisitos anteriormente apresentados tendo em vista já a estruturação arquitetónica do sistema.

1.1 Use Case: Consultar Evento

Ator	Utilizador	
Descrição	O utilizador consulta um evento	
Pré-Condição	True	
Pós-Condição	O utilizador consulta o evento pretendido	
Fluxo Normal	Input do Ator	Resposta do Sistema
	1. Indica que pretende consultar um evento	
		2. Processa o pedido e apresenta as informações sobre o evento ao utilizador

Tabela 1: Use Case Consultar Eventos

1.2 Diagrama de Atividade: Consultar Evento

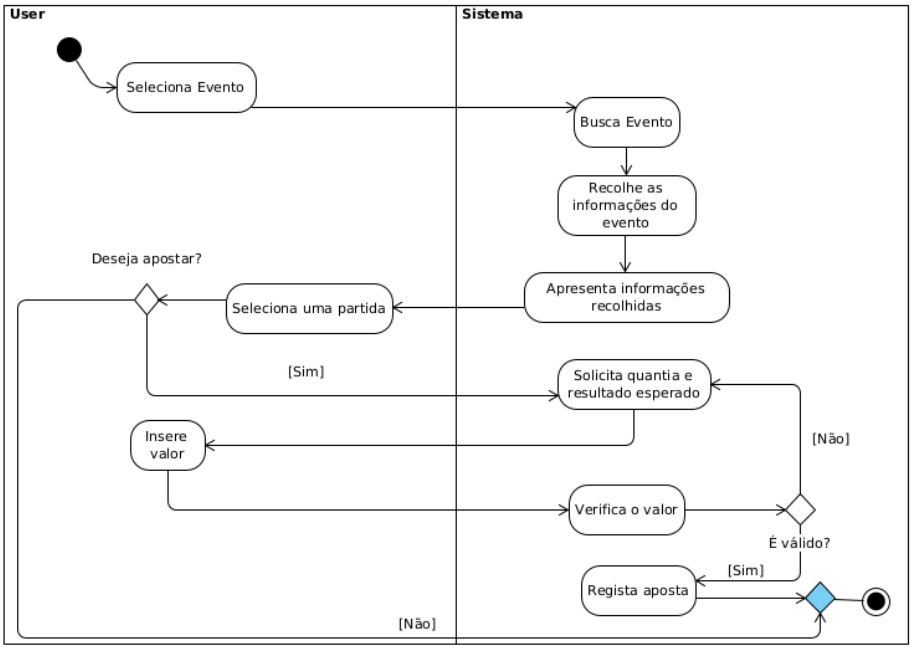

Figura 1: Diagrama de Atividade Consultar Evento

2 Objetivos de Qualidade

Para além dos requisitos funcionais, é também muito importante manter bem explicito que requisitos de Qualidade tem o sistema de cumprir, estes mais uma vez também já enunciados na fase anterior do projeto, vão ser também eles fatores de decisão no que toca a arquitectura do sistema.

Assim no que toca a arquitetura, vamos nos focar ao longo desta fase em alguns destes requisitos de qualidade como a **Segurança**, a **Performance e Eficiência**, a **Manutenção** e **Operacionalidade**

2.1 Segurança

A segurança é todo o requisito que vem proteger os dados da aplicação de acidentes, acessos malignos como destruição, alteração e corrupção de dados do sistema.

2.2 Performance e Eficiência

Uma vez que se trata de um sistema de apostas mexendo com transferências e movimentos bancários reais, e dados pessoais dos utilizadores é de grande importância que estes dados se mantenham livres de acidentes ou de atentados mal intencionados, evitando assim a corrupção dos dados, destruição ou roubo dos mesmos.

2.3 Manutenção

Tratando-se de uma aplicação dinâmica e evolutiva, o sistema deve estar preparado para quando necessário sofrer alterações às suas funções e aplicações numa perspectiva de o dinamizar.

2.4 Operacionalidade

O Sistema deve ser interactivo, de fácil compreensão e atractivo para todos os utilizadores com idade suficiente para utilizar a aplicação.

3 StakeHolders

Para aprovação da arquitectura é necessário cumprir todos os requisitos e expectativas dos nossos Stakeholders e associados ao projecto assim identificamos alguns deles e as suas respectivas expectativas, estando esta informação na tabela a baixo.

Função	**Relevância para Aprovação**	**Expectativa**
Líder do Projeto	Alta	Visão geral do risco técnico, interfaces externas
Patrocinador do Projeto ou Comprador do produto (Oasis)	Alta	Garantia que os requisitos de qualidade consigam ser alcançados e que seja apresentado um sistema funcional com os requisitos pedidos implementados e documentação bem estruturada e explicada.
Desenvolvedores	Nenhuma	—
Utilizadores ou Apostadores	Média	Garantia que os requisitos de Operacionalidade e Segurança vão estar de acordo com os requisitos levantados
Analistas Desportivos	Média	Garantia que todos os dados exibidos no Sistema vão estar de acordo com a realidade
Analistas de Marketing	Nenhuma	—

Tabela 2: StakeHolders

II Restrições

As restrições do projeto são qualquer tipo de requisito que restrinja os arquitetos de software na sua liberdade de design e tomada de decisões referentes à implementação ou processo de desenvolvimento do projeto, estas restrições podem se dividir em vários parâmetros, nomeadamente **Restrições Técnicas**, **Restrições Organizacionais** e **Restrições Politicas**.

1 Restrições Técnicas

As restrições tecnicas podem ser oriantações dos compradores do produto, em relação a operações, seleção de tecnologia, frameworks ou arquiteturas de referência.

Tendo em conta estes aspetos temos então como restrições técnicas, o uso da framework Flask, framework esta disponivél para a linguagem Python que utiliza uma arquitetura predefinida MVT(Model View Template), uma arquitetura similar ao MVC, que em comparação é um modelo mais facilmente escalável, facilitando assim o possível crescimento da aplicação e assim também possibilitar a criação de uma interface gráfica. O uso desta linguagem e framework traz inevitavelmente restrições provenientes da própria linguagem.

Relativamente à imparcialidade, a aplicação terá de ser suportada nos vários diferentes sistemas operativos, e tratando-se de uma aplicação web, nos vários browsers.

2 Restrições Organizacionais

Todas as restrições relativas a datas de entrega e orçamentos são referentes às restrições organizacionais, assim dentro deste parâmetro a maior restrição que existe para os arquitectos do sistema é a data de entrega final do projeto ser em janeiro, restando assim menos de 1 mês para este ser dado como concluído, esta data ainda poderá ser negociada com o nosso comprador.

3 Convenções

Relativamente a convenções a nossa arquitetura está então documentada seguindo o template arc42 e a linguagem da nossa plataforma será o Português de Portugal.

III Contexto e Área de Trabalho

1 Contexto Negócio

De seguida iremos mostrar todos os parceiros de comunicação mostrando o que delimita o nosso sistema de todos os nossos em relação aos mesmos.

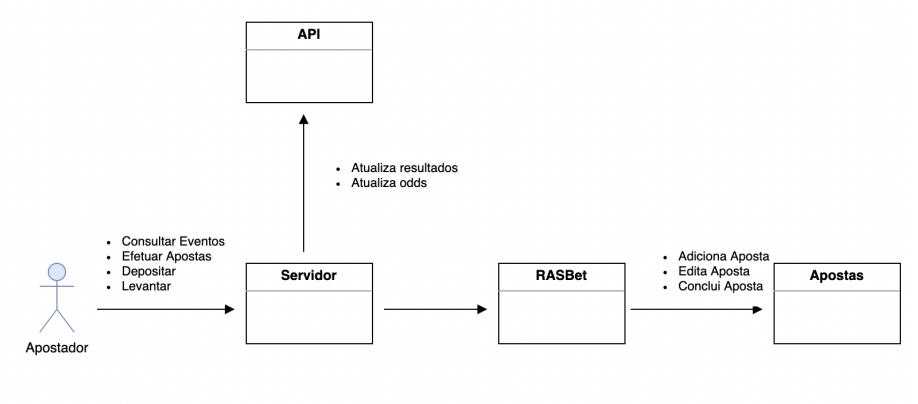

Figura 2: Diagrama de contexto do Sistema

Apostador	Utilizador que utiliza as funcionalidades do Sistema.
API	Providencia os valores atuais dos resultados dos eventos.
Apostas	Agrega todas as apostas ativas e inativas do Sistema.

Tabela 3:

2 Contexto Técnico

Assim, iremos representar também as interfaces técnicas, ligando o nosso sistema ao seu ambiente.

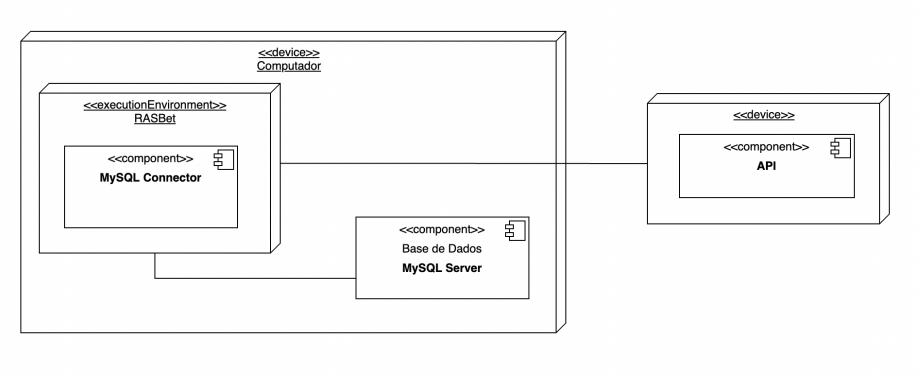

Figura 3: Diagrama de instalação

Dentro do dispositivo que corre a plataforma, ao executar a mesma irá ser estabelecida uma conexão com a base de dados da RASBet e também com a API fornecida, onde são retirados os resultados dos Eventos. A API não se encontra no mesmo dispositivo onde a plataforma se encontrará a correr.

IV Estratégia Solucionada

A tabela seguinte irá, de forma sucinta, resumir algumas decisões fundamentais e estratégias de solução, que moldam a arquitectura do nosso sistema. Posteriormente irá ser explicado de uma forma mais detalhada.

Goal/Requirement	Architectural Approach	Details
Segurança	Encriptar passwords e informações confidenciais e efetuar backups periódicos.	IV 1
Performance	Tratar imagens como um caso especial, armazenando as imagens no sistema de ficheiros em vez de na base de dados.	IV 2
Manutenção	Organizar o source code de forma a ser fácil de adicionar / remover features no sentido de dinamizar e evoluir a aplicação.	IV 3
Operacionalidade	Navegação feita com base em uso de botões, com títulos fáceis de interpretar.	IV 4

Tabela 4: Solution Strategy

1 Segurança

Para a parte de Segurança, iremos encriptar todas as passwords que são inseridas, de modo a garantir a privacidade de todos os utilizadores.

Além disso, de modo a manter a confidencialidade dos dados do nosso utilizador, iremos utilizar uma autoridade de certificação chamada *Let´s Encrypt*, que nos irá permitir encriptar todos os dados deste. Assim, em caso de fuga de dados estes não poderão ser visualizados, pois não poderão ser desencriptados.

Para finalizar, de forma a manter a integridade dos dados de toda a aplicação, iremos realizar backup de todos estes para uma segunda base de dados. Estes backups irão decorrer periodicamente e automaticamente.

2 Performance

Sendo que a performance tem uma grande importância para a escalabilidade de toda e qualquer aplicação, iremos reduzir ao máximo a carga da base de dados, minimizando as query's feitas sobre o mysql server, e guardando também imagens num filepath, invés de guardarmos na base de dados.

3 Manutenção

Para que a nossa aplicação seja facilmente atualizada, iremos focar-nos na organização do código, de forma a se tornar mais fácil a implementação de novas funcionalidades, ou até mesmo a remoção.

4 Operacionalidade

Sendo que a primeira interação com os utilizadores é a que tem maior impacte, iremos, com o auxílio do bootstrap, utilizar maioritariamente botões para a navegação dentro da aplicação, de forma a tornar-se mais fácil e interativa a interação do utilizador com a aplicação.

V Building Block View

O *Building Block View* apresentado proporciona uma divisão das várias camadas lógicas do projeto de forma abstrata sem revelar detalhes da implementação, permitindo uma melhor compreensão das diferentes partes do projeto por parte dos stakeholders.

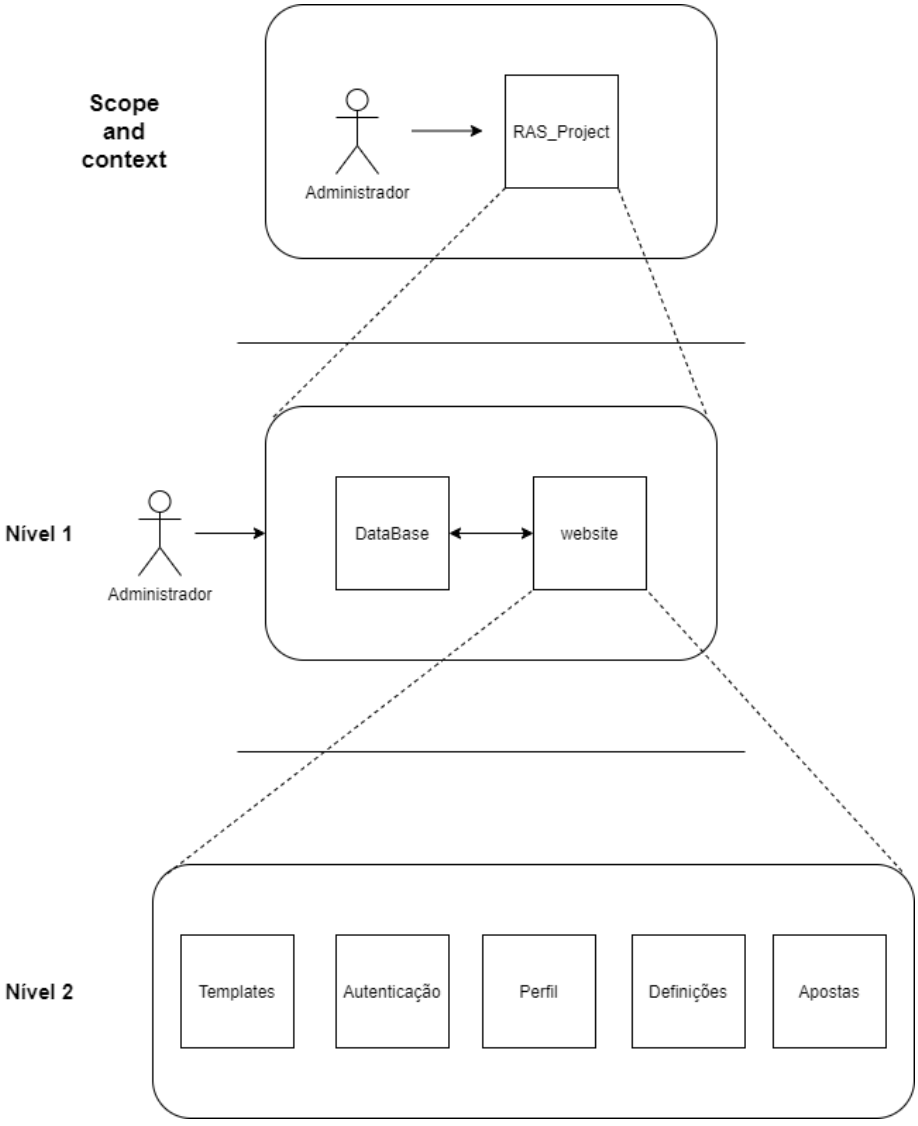

Figura 4: *Building Block View*

Tendo em consideração o *Building Block View* vemos que possuímos 2 níveis de hierarquia que achamos relevantes para expor a estrutura do projeto de forma abstrata sem perder a perceção da forma de implementação da aplicação.

No primeiro nível podemos observar as duas grandes partes da nossa aplicação em que cada uma possuí um propósito bem definido e distinto.

Nome	**Responsabilidade**
DataBase	Bloco responsável por criar e tratar de todos os assuntos relacionados com a base de dados do sistema
website	Bloco responsável por todos os aspetos e funcionamentos da aplicação web

Tabela 5: Nível 1 do *Building Block View*

O segundo nível é uma especificação do bloco de nível 1 que diz respeito ao website, neste bloco são apresentados os vários subsistemas que achamos imprescindíveis para o bom funcionamento da aplicação. De forma semelhante ao nível 1, estes serão descritos na tabela apresentada de seguida.

Nome	**Responsabilidade**
Templates	Bloco responsável pelo aspeto das diferentes páginas da aplicação e que serão expostas ao utilizador
Autenticação	Bloco responsável por permitir a um utilizador registar-se e/ou entrar no sistema para, assim, ser capaz de utilizar todas as funcionalidades do sistema
Perfil	Bloco responsável por permitir a um utilizador editar e/ou consultar os seus dados pessoais bem como o seu saldo ou até o seu histórico
Definições	Bloco responsável por tornar o sistema mais pessoal para cada utilizador, permitindo a cada um manipular as diferentes opções de forma a tornar a aplicação o mais apelativa e mais interativa possível para si
Apostas	Bloco responsável por todo o sistema de apostas, desde a exposição das mesmas como também permitir a um utilizador proceder com o grande propósito deste sistema que é a realização de apostas desportivas

Tabela 6: Nível 2 do *Building Block View*

VI Runtime View

Neste tópico, pretendemos demonstrar como os nossos blocos de construção cooperam com os utilizadores e sistemas vizinhos. Para melhor representar estes cenários, decidimos voltar a utilizar diagramas de sequência, criados no VisualParadigm.

Nesta fase, estes diagramas foram criados e realizados de forma mais pormenorizada, tendo em consideração, de forma mais concreta em comparação com a primeira fase, aquilo que ocorrerá nos vários cenários representados e da forma como o nosso sistema responderá aos vários pedidos do utilizador.

Foram realizados apenas alguns diagramas porque sentimos que estes cenários são os mais significativos para representar o nosso sistema.

1 Registar

Para representar o procedimento que ocorre cada vez que um utilizador se pretende registar, construímos o diagrama de sequência representado.

Neste diagrama podemos reparar no pormenor que nos indica que o autor deste cenário é o *Utilizador*, pois só um *Utilizador* já registado é considerado um *Apostador*.

Após selecionar que pretende efetuar um registo, o *Utilizador* irá receber uma página dedicada apenas ao registo que lhe irá solicitar que preencha os campos necessários para concluir o registo. Se estes campos não estiverem bem preenchidos, irá ser solicitado que os volte a preencher. Neste momento, o *Utilizador* idealmente corrige os campos mal preenchidos e efetua o registo na aplicação, no entanto, se desejar, poderá também cancelar o registo e voltar à página principal.

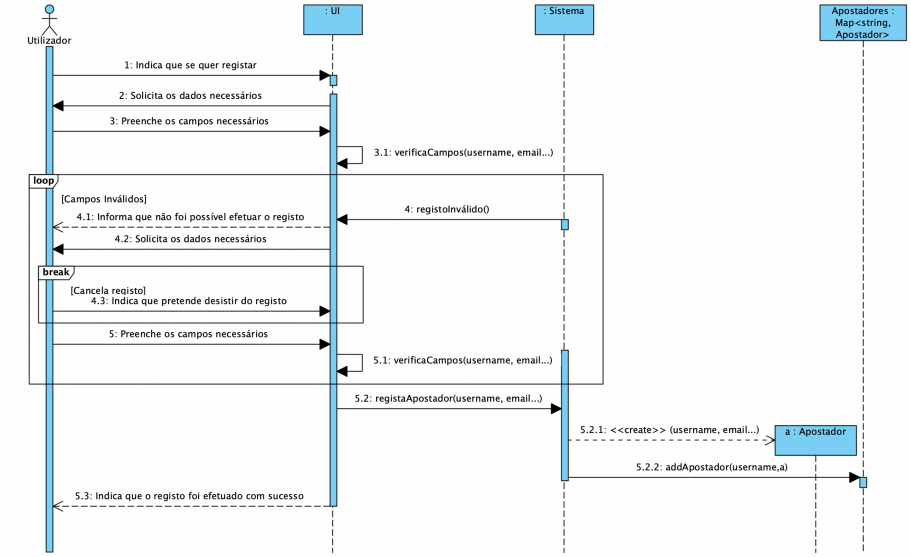

Figura 5: Diagrama de Sequência - Registar

2 Iniciar Sessão

Neste diagrama está representada a autenticação de um *Apostador* na nossa Aplicação.

Posteriormente ao pedido de autenticação efetuado pelo *Apostador*, o sistema solicita-lhe os campos necessários para que os possa preencher. A autenticação apenas será bem-sucedida se os valores inseridos existirem na nossa base de dados.

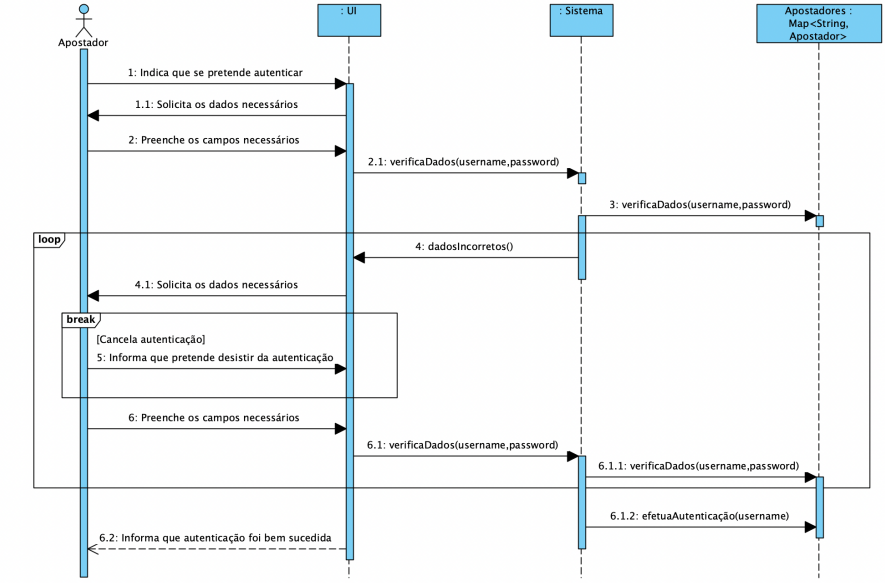

Figura 6: Diagrama de sequência - Iniciar Sessão

3 Apostar

Considerámos que o momento de efetuar uma aposta é um dos principais cenários da nossa aplicação e, portanto, não podia deixar de ser feito um diagrama de sequência a representar precisamente este cenário.

Após o *Apostador* selecionar um evento no qual pretende apostar e indicar ao sistema esse desejo, será solicitada a quantia que pretende apostar e o resultado esperado. Depois de esta quantia ser inserida, o sistema irá averiguar se essa quantia é válida e voltará a solicitá-la em caso negativo. Quando esta quantia for aceite, o sistema efetuará a aposta e irá atualizar o saldo do *Apostador*.

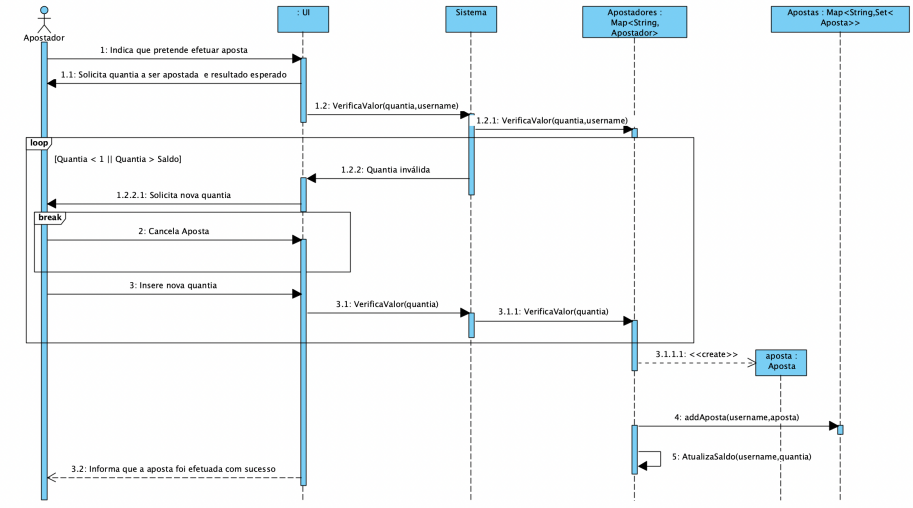

Figura 7: Diagrama de Sequência - Apostar

4 Carregar Saldo

Para que o *Apostador* possa apostar continuamente, o sistema terá que possibilitar ao *Apostador* carregar saldo para a aplicação. Depois de indicar o pretendido, o sistema irá solicitar a quantia que pretende carregar e irá avaliar esta quantia para verificar se tudo se encontra correto. No caso de a transferência não ser aceite por saldo insuficiente, o *Apostador* será devidamente informado do ocorrido e será lhe solicitado uma nova quantia. Quando o sistema validar esta quantia, este irá efetuar o carregamento e irá atualizar o saldo do *Apostador*.

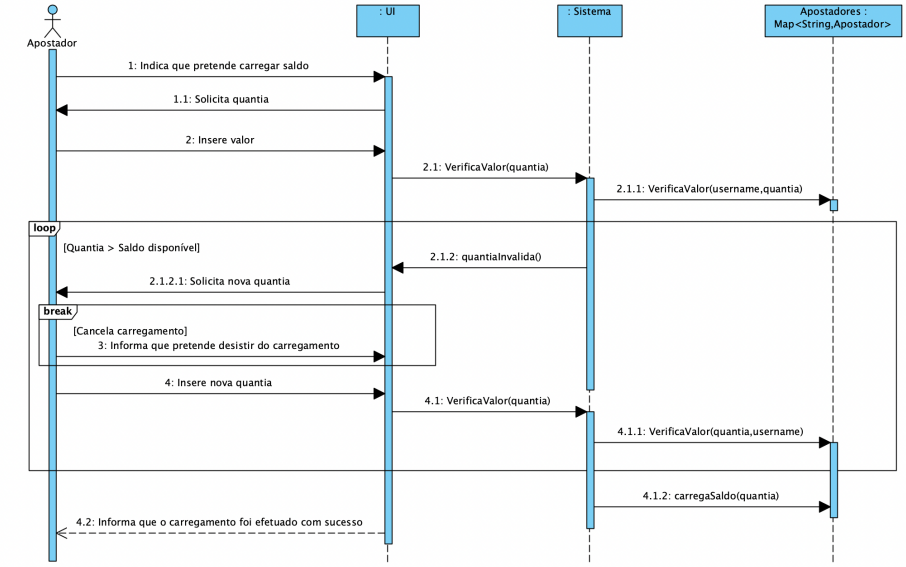

Figura 8: Diagrama de Sequência - Carregar Saldo

5 Levantar Saldo

Tal como faz sentido ser possível carregar saldo para a aplicação, também considerámos essencial que o *Apostador* possa levantar qualquer quantia válida de saldo da sua conta. O procedimento é idêntico ao do carregamento, sendo que lhe é solicitado uma quantia para levantamento, que irá ser validada pelo sistema que irá garantir que esta quantia não é maior do que o seu saldo. Se tudo estiver válido, o *Apostador* irá poder levantar esta quantia para a sua conta principal, e o seu saldo será atualizado.

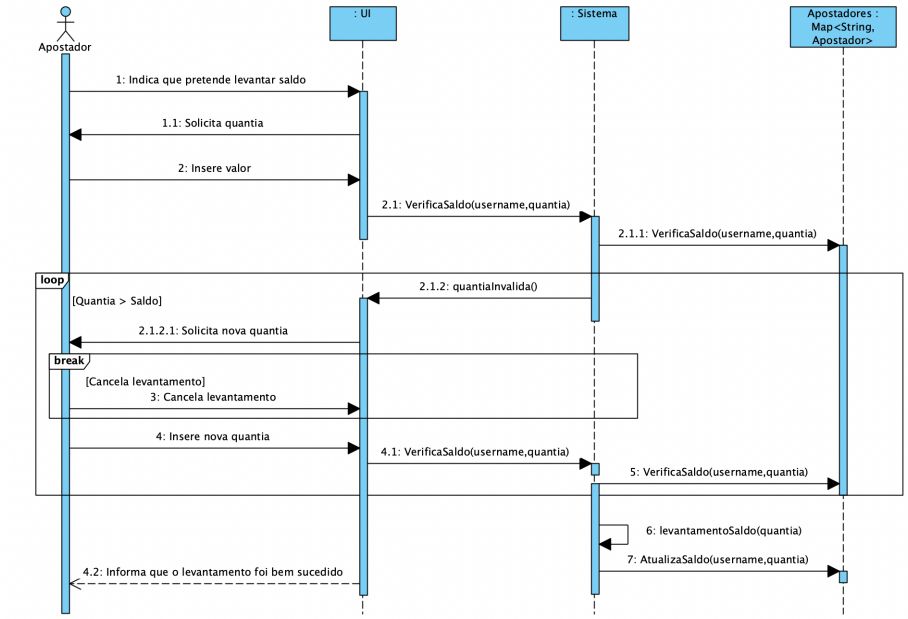

Figura 9: Diagrama de Sequência - Levantar Saldo

6 Consultar Desporto

Decidimos representar também este cenário, por ser um cenário possível a qualquer *Utilizador*, esteja este registado ou não. Logo, se um *Utilizador* pedir a consulta de um determinado desporto, o sistema irá reunir as informações deste desporto e irá apresentá-las ao *Utilizador*.

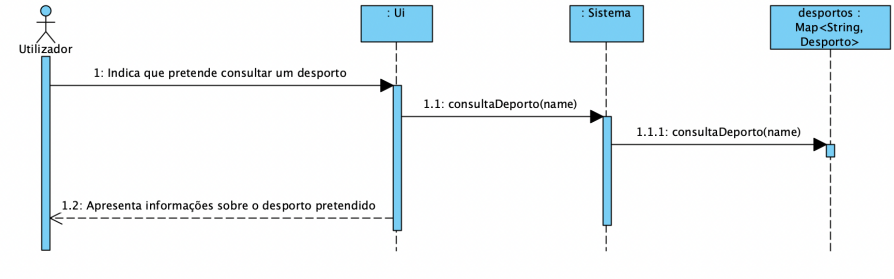

Figura 10: Diagrama de Sequência - Consultar Desporto

VII Deployment View

No decorrer deste projeto foi sempre mencionada a forma como o nosso trabalho irá responder a certas ações e a forma como pretendemos realizar certos requisitos e objetivos, no entanto, software não funciona sem hardware, por isso, é importante que conheçamos bem a nossa infraestrutura técnica, pois esta irá influenciar bastante o nosso sistema.

1 Descrição geral da infraestrutura técnica do Sistema

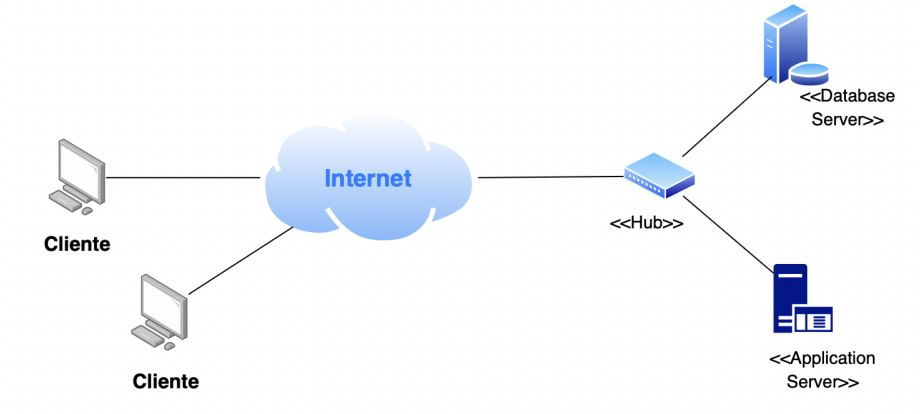

Figura 11: Descrição simples da infra-estrutura técnica

Neste gráfico, representámos, de forma simplificada, a infraestrutura em que o nosso sistema corre.

Começa pelo momento em que um ou vários clientes acede à internet e se conecta ao nosso website. Este website é, no entanto, suportado pela sua base de dados onde a maioria dos dados estarão guardados, e o servidor aplicacional que disponibiliza um ambiente para a instalação e execução de nossa aplicação.

VIII Conceitos Transversais

1 Conceitos de Domínio

Nos conceitos de domínio é essencial o desenvolvimento de um modelo que exprima de uma forma clara o nosso sistema, para isto escolhemos o modelo de domínio que possuí os elementos do sistema e as relações entre eles com uma linguagem universal permitindo, assim, apresentar o sistema da aplicação com uma compreensão simples.

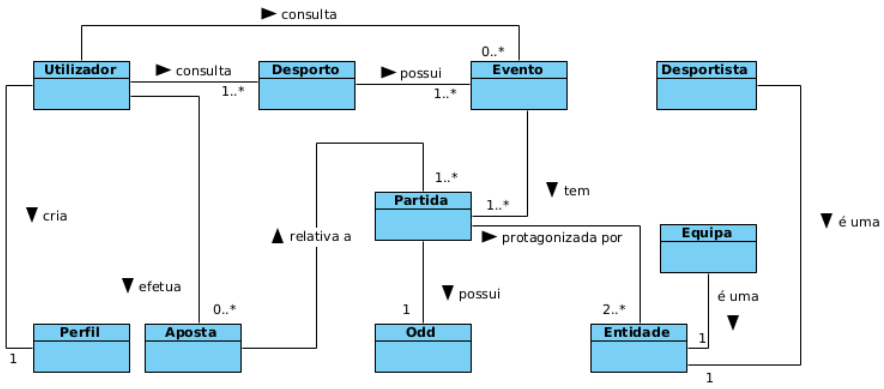

Figura 12: Modelo de Domínio

2 Conceitos da Experiência do Utilizador

A experiência do utilizador na nossa aplicação será muito influenciada pela interface da mesma. Esta será simples e de fácil interação para que, assim, qualquer utilizador, seja ele uma pessoa com muito ou pouco conhecimento sobre a manipulação de aplicações web, capaz de usufruir do sistema no seu potencial máximo.

3 Conceitos de Segurança e Proteção

Para garantir a segurança e a proteção do nosso sistema iremos encriptar as *passwords* dos utilizadores, para além deste fator crucial para impedir invasões das contas dos utilizadores, também temos em atenção a encriptação dos dados para, deste modo, se houver uma fuga estes não serem visíveis. Ainda achamos importante a criação de um backup para um base de dados adicional.

4 Conceitos *Under the Hood*

Será imprescindível o controlo das idades dos utilizadores, isto porque é um sistema que envolve o aposta de dinheiro que só são permitidas a pessoas com 18 ou mais anos. Para confirmar tal facto é essencial os dados do Cartão de Cidadão, com isto ainda conseguimos garantir que nenhum utilizador possua várias contas.

5 Conceitos de Desenvolvimento

O nosso sistema irá ser desenvolvido em Python com o apoio da sua *framework* Flask para desenvolvimento web e será testado exaustivamente para o produto ficar o melhor possível.

6 Conceitos Operacionais

Existe um requisito fundamental para um bom funcionamento da aplicação sendo este a escabilidade. É com a capacidade do nosso sistema ser escalável que nos é permitido aumentar o número de utilizadores com o aumento da procura na nossa aplicação.

IX Decisões Arquitetónicas

De modo a estruturar a nossa aplicação, tivemos de tomar algumas decisões arquitetónicas.

1 Utilização de templates html

Com o objetivo de tornar a interface da nossa aplicação modificável optamos por utilizar um template *html* que será comum a todas as *view*.

Assim, temos um ficheiro *base.html* que contém o código para o *header*, a *navbar* e as notificações da nossa aplicação, que impede que este código seja repetido em todas as views e, na eventualidade de ter de ser alterado, apenas será neste ficheiro.

2 Flask Blueprints

De modo a estruturarmos a nossa aplicação decidimos utilizar *Flask Blueprints*. Com a utilização destes, conseguimos dividir o código em vários componentes (*templates*, por exemplo, onde serão guardados os ficheiros *html*) tornando este mais organizado e mais escalável.

Estes componentes irão seguir uma estrutura funcional, ou seja, cada componente terá o código correspondente a uma função (como por exemplo, o componente associado às views).

3 Armazenamento de dados

De modo a termos sempre acesso aos dados da nossa aplicação iremos ter duas bases de dados, uma principal e uma secundária.

Na base de dados principal serão inseridos todos os dados relativos a Utilizadores, Eventos, Apostas, etc.

A base de dados secundária irá realizar backups periódicos da principal e, caso essa falhe por alguma razão, os dados guardados nesta base de dados secundária serão repostos na outra.

Assim, garantimos a integridade dos dados da nossa aplicação.

X Qualidade

Os requisitos de qualidade são talvez dos parâmetros mais importantes de descrever em toda a aplicação, uma vez que tem um impacto muito grande na arquitectura da mesma, assim indo de acordo com a estratégia solucionada encontramos alguns requisitos deste nível muito importantes a cumprir.

Na figura abaixo, tempos a explicitação destes requisitos de qualidade e alguns exemplos e impactos que estes vão influenciar na arquitetura da aplicação.

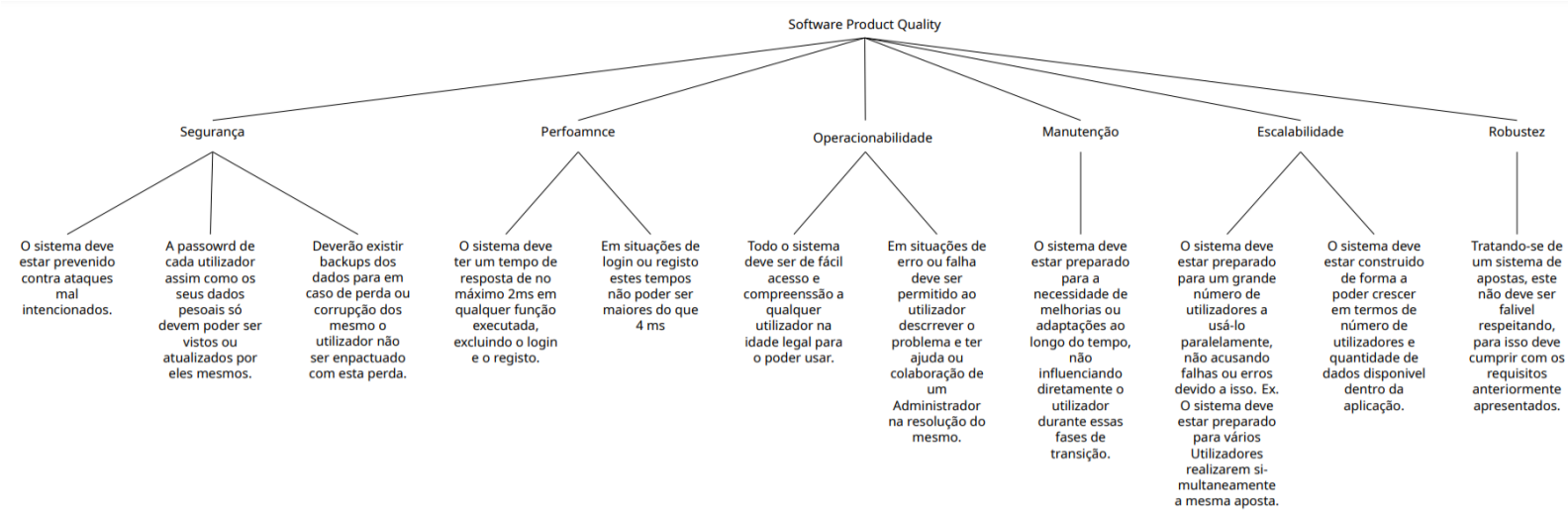

Figura 13: Quality-Tree

XI Riscos e dívida técnica

Embora tendo uma arquitetura previamente estudada e pensada para o melhor desenvolvimento possível existe sempre, apesar de tudo, determinados problemas ou falhas que poderão acontecer, seja devido à interface gráfica, falta de tempo de desenvolvimento ou mesmo de testes.

1 Segurança

Tratando-se de um site de apostas, envolvendo o dinheiro dos utilizadores e os respectivos dados pessoais destes, a segurança é uma das nossas prioridades, porém não é fácil garantir a 100% este requisito, uma vez que muita coisa pode correr mal, desde falha humana por parte dos desenvolvedores, a mesmo terceiros mal intencionados. Assim, é sem dúvida um risco na arquitetura e algo a que a equipa tem de prestar muita atenção.

2 Legal

Do ponto de vista legal e tratando-se, como já referido, de um site de apostas , este está sujeito a vários termos e legislações impostas pelo Estado que têm necessariamente de cumprir, desde, como já enunciado em cima relativamente a segurança, toda a protecção de dados referente ao utilizador como mesmo a todas as legislações impostas de modo a que seja uma casa de apostas reconhecida e fidedigna, assim este é obviamente um problema ao qual a equipa vai ter de prestar muita atenção tanto no desenvolvimento, como na manutenção da plataforma a considerar.

3 Interface

A interface desta aplicação representa também um risco, uma vez que pode ser o epicentro de muitos problemas, sendo que esta acaba por abranger todos os outros tópicos aqui apresentados, podendo ter um impacto significativo em aspectos como a segurança e a robustez do produto. Assim, tem de ser encarado como um aspeto muito relevante para que da melhor forma consiga fazer cumprir todos os requisitos de qualidade impostos à arquitectura idealizada.

4 Sincronização

Uma vez que todo o nosso sistema e o seu consequente uso está associado à sincronização ao minuto de todos os eventos que lhe são intrínsecos, a sincronização do conteúdo apresentado na aplicação é ou pode ser um problema, estando o sistema comprometido a fornecer a hipótese de apostas em variados jogos e eventos diários. Assim há a necessidade de procurar evitar que isto seja um problema procurando mecanismos que o contrariem, porém obstante isso é sem dúvida um risco e um aspeto a ter em consideração.

5 Testes

Tendo em conta a curta margem de manobra que a equipa de desenvolvimento terá para facultar a versão final, a testagem do produto será um problema, sendo este um fator que pode influenciar em grande parte o sucesso da aplicação, uma vez que a testagem vem prevenir ou identificar falhas em todos os aspetos enunciados acima, sendo então um dos maiores riscos que podemos identificar caso o tempo seja um fator decisivo no decorrer do desenvolvimento da aplicação.

XII Glossário

O glossário apresentado posteriormente contém a explicação terminologias sobre o mundo das apostas desportivas.

- **Evento** ⇒ Jogos ou corridas nos quais serão exercias as apostas
- **Aposta** ⇒ O utilizador ganha a aposta caso a mesma acerte nos resultados de jogos ou corridas
- **Template** ⇒ Um template é utilizado para fazer com que a *web based app* tenha o aspeto e as funções que ela precisa para funcionar num ambiente *web*
- ***View*** ⇒ Página da aplicação apresentada ao utilizador
- ***Header*** ⇒ Efeitos estáticos que são visíveis no topo da aplicação
- ***Navbar*** ⇒ Painel de controlo entre as diferentes secções da aplicação
- **RASBet** ⇒ Produto em desenvolvimento

XIII Conclusão

Esta segunda fase do projeto abordou, numa perspectiva mais atenta e cuidadosa, a arquitectura que o sistema deve implementar, sendo assim, talvez em parceria com a primeira fase, uma fase importantíssima no desenvolvimento do projeto.

Inicialmente, começámos por abordar as restrições que este projeto tem a nível aquitetónico, e aprofundar o nosso conhecimento relativo aos requisitos de qualidade que a arquitectura deve seguir, de seguida desenvolvemos as variadas respostas da aplicação a nível arquitectónico quando confrontada com as situações reais que poderão acontecer estando esta em uso, explorando assim esses requisitos de qualidade num contexto real, e procurando enquadrá-los em determinadas situações.

No que diz respeito a esta fase do projeto, consideramos, enquanto grupo, ter alcançado uma solução possível, exequível e capaz de corresponder ao problema que nos foi proposto pelos docentes.

Relativamente à próxima fase, tratando-se esta de uma fase já de desenvolvimento, enquanto grupo consideramos ter já uma boa ideologia e base do caminho a seguir, considerando assim que todo este trabalho de planeamento vai por ventura dar os seus frutos na fase de desenvolvimento.