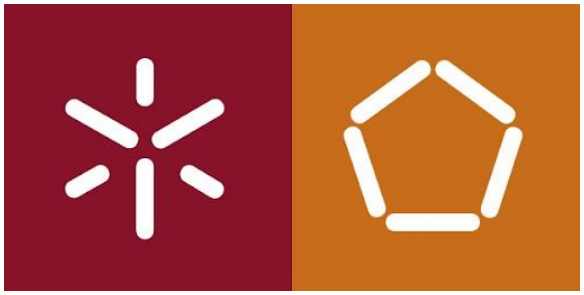


Universidade do Minho

Documento de Alterações

Ano letivo 2022/2023

janeiro 2023

Mestrado em Engenharia Informática

Unidade Curricular de Requisitos e Arquiteturas de Software

Ana Gonçalves pg50180

Bruno Pereira pg50271

João Freitas pg50462

Francisco Toldy pg50379

João Delgado pg50487

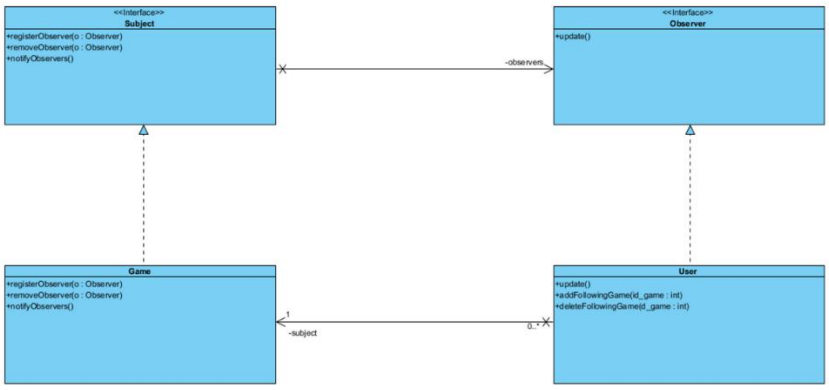
Alterações Efetuadas

Back-End

Padrões de Arquitetura

A segunda funcionalidade sugerida para incorporação no produto anteriormente desenvolvido, consistia em permitir que um apostador pudesse escolher seguir um jogo e as apostas relativas ao mesmo, ou seja, sempre que houvesse alterações nas odds, o utilizador receberia uma notificação caso estivesse interessado. Assim, era evidente o tipo de estrutura necessário para integrar a funcionalidade ao produto, e o padrão mais adequado para a implementar, o padrão Observer.

Assim, na perspetiva da aplicação, o jogo seria o subject, que estaria sujeito a alterações pontuais e o apostador seria o observer, que poderia escolher observar um certo jogo, sendo que qualquer alteração do mesmo iria desencadear uma ação de notificação para que os utilizadores que o seguem saibam que uma das suas odds foi alterada. Esta arquitetura poderá ser expressa através de um diagrama de classes, de modo seja mais percetível a solução desenvolvida.



Deste modo, estarão implementados no User métodos que permitirão aos utilizadores seguirem e deixarem de seguir um jogo, os métodos addFolowingGame() e deleteFollowingGame(). Estes desencadearão uma chamada aos métodos do Game que se pretende seguir, registerObserver() e removeObserver(), que fará o registo dos utilizadores que terá de notificar caso alguma das suas odds seja alterada.

Com o padrão implementado, sempre que qualquer odd do jogo, que um ou mais utilizadores escolheram seguir, for alterada, estes receberão uma notificação, que será desencadeada pelo notifyObservers(). Este método por sua vez, chamará os métodos update() dos Observers que tem registados, despoletando o envio de um email de notificação para cada um deles.

Code Refactoring

No que toca a *code refactoring* foram identificados dois tipos de *"code smells"*: *Strings hardcoded*, por exemplo nas respostas à *front-end*, e métodos muito extensos que poderiam ser divididos em múltiplos métodos, melhorando, assim, a legibilidade do mesmo.

Deste modo, procedeu-se à criação de uma classe que constrói as respostas para a *front-end* substituindo-se as *strings hardcoded* por métodos dessa classe. No que toca aos métodos extensos, analisou-se quais desses poderiam ser divididos, e onde era possível foram criados métodos auxiliares, permitindo a diminuição da complexidade de cada método.

Front-End

Nesta última fase, foi pedido a implementação de 2 novas funcionalidades:

- Apostas múltiplas com o máximo de 20 elementos
- Apostadores podem seguir jogos

Em termos de front-end, optou-se por apenas limitar o número de apostas no momento de efetivar a apostas. Isto é, quando o número de apostas numa única aposta múltipla ultrapassa o número 20, o apostador não vai conseguir prosseguir com a aposta e vai receber o erro desse mesmo limite.

Relativamente à outra funcionalidade, optou-se inicialmente por apenas adicionar um ícone em cada jogo que lhe é apresentado que permitia o jogador adicionar/remover o jogo dos seus favoritos. Este ícone consistia numa estrela que teria cor diferente consoante o evento estivesse a ser seguido ou não. Mais a frente a equipa decidiu adicionar uma página dedicada (na página de perfil) para que os apostadores consigam aceder facilmente a todos os jogos que seguem, podendo até os eliminar dos seus favoritos, se assim o desejarem. Para acomodar as alterações foi necessário criar mais páginas e componentes, no entanto, estes componentes não influenciaram a estrutura previamente construída, sendo simplesmente uma adição à mesma.

Em termos de limpeza de código, foram aplicadas algumas melhorias em termos de minimização de carga no servidor, redução da quantidade de dados enviado pela árvore de componentes, acréscimo de segurança de acesso e melhorias na facilidade de uso pelos utilizadores.

- **Minimização de carga no servidor:** ao contrário do que acontecia na versão entregue na segunda fase, já não existe a sequência constante de pedidos à front-end em todas páginas, existindo apenas nos momentos em que são estritamente necessário.
- **Redução de quantidade de dados enviados pela árvore de componentes:** utilizando o “useContext” foi possível eliminar 2 das variáveis que eram passadas como argumentos para uma grande parte dos componentes. Com a utilização do “useContext” a lista de argumentos diminui e está atualmente mais legível.
- **Acréscimo de segurança:** Ao contrário da quebra de segurança existente na fase 2, nesta fase 3 já não é possível acessos diretos a nenhuma das páginas da aplicação, sem passar inicialmente por uma fase de autentificação
- **Facilidade de uso pelos utilizadores:** Com a adição de cookies é possível agora guardar a informação relativamente ao utilizador, o que vai permitir fazer refresh das páginas sem ter de iniciar sessão novamente, como também aceder à sua conta sem iniciar sessão, se a sessão não foi terminada.