codeflex – aplicação web de programação competitivA

miguel antónio ferrão Brito

RELATÓRIO PARA A OBTENÇÃO DO GRAU DE LICENCIADO

EM ENGENHARIA INFORMÁTICA

Mês/Ano (conclusão do estágio/projeto)

Ficha de Identificação

Aluno: Miguel António Ferrão Brito

Nº 1011695

Licenciatura: Engenharia Informática

Estabelecimento de Ensino:

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

Orientador de Estágio:

Nome: Professor Celestino Gonçalves

Grau académico: Mestre

Agradecimentos

Resumo

Abstract

# Índice

# Siglário

# Introdução

## Motivação

Nos dias que correm existe uma grande procura no que diz respeito à área das TI, sendo a programação um dos principais focos do mercado. Nesta procura e escolha de candidatos, muitas empresas recorrem a problemas tipo para avaliar o pensamento lógico dos seus candidatos, a forma como lidam com o problema e o implementam e o seu conhecimento das mais variadas tecnologias.

Neste contexto, as aplicações *web* de programação competitiva têm uma grande importância como um método de preparação e estudo de algoritmos e estruturas de dados, e como tal, existe a necessidade de uma aplicação que possa reunir problemas tipo para facilitar e organizar a aprendizagem e a prática aos utilizadores, simulando um ambiente de pressão pela realização de torneios.

A nível pessoal, o interesse surgiu pela participação em alguns torneios no passado e pela consideração deste desafio como interessante para aplicar e solidificar os conhecimentos adquiridos ao longo da licenciatura e explorar novas tecnologias.

## Objetivos

Este projeto tem como objetivo a implementação de uma aplicação *web* que permitirá a prática de programação competitiva. Esta prática consiste na criação de soluções para um determinado problema, proposto por outros utilizadores, recorrendo a código que é posteriormente compilado e executado relativamente a *inputs* predefinidos. Caso o código submetido pelo utilizador gere os *outputs* pretendidos considera-se a resposta como válida. Tendo em vista o proposto, a plataforma deverá cumprir os seguintes objetivos:

1. Plataforma online
2. Sistema de autenticação de utilizadores.
3. Desafios para prática sem limitações de tempo organizados por categorias.
4. Torneios limitados por tempo.
5. Conquistas obtidas pelo utilizador de acordo com a sua performance nos desafios.
6. Classificação de utilizadores por ELO.
7. Editor de texto embutido.
8. Compilação e avaliação das soluções submetidas.
9. Criação de torneios públicos ou privados por parte do utilizador

## Estrutura do Documento

# Estado da Arte

A realização do estudo e de uma análise crítica das restantes plataformas da área é um passo essencial para identificar e procurar colmatar as falhas existentes no modelo de aplicações atual.

Inicialmente foram realizadas pesquisas acerca das principais aplicações atuais e das suas principais características das quais resultou a Tabela 1.

Esta pesquisa foi baseada nos seguintes parâmetros:

1. Características inovadoras
2. Semelhança com os objetivos apresentados
3. Conhecimento e experiência anterior de algumas das plataformas

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plataforma  Características | Codeflex | TopCoder | LeetCode | HackerEarth | HackerRank | CodeChef |
| Torneios | Sim | Sim | Sim | Sim | Sim | Sim |
| Prática de desafios | Sim | Sim | Sim | Sim | Sim | Sim |
| Simulação de Torneios | Não | Não | Sim | Não | Não | Não |
| Conquistas | Sim | Sim | Não | Não | Sim | Sim |
| Classificação | Sim | Sim | Sim | Sim | Sim | Sim |
| Procura de Emprego | Não | Sim | Não | Sim | Sim | Não |
| Networking/Discussão | Não | Sim | Sim | Sim | Sim | Sim |
| Mock interviews | Não | Não | Sim | Não | Não | Não |
| Torneios Privados | Sim | Não | Não | Não | Não | Não |

Tabela 1- Comparação de funcionalidades por plataformas

## Aplicações existentes

Das aplicações apresentadas anteriormente foram escolhidas a *LeetCode* e *HackerRank* para uma análise mais aprofundada e crítica comparativamente à solução que se pretende desenvolver tendo em vista as suas semelhanças e completude.

### LeetCode

A plataforma *LeetCode* [1]é uma das plataformas mais populares da atualidade e conta em especial com uma enorme variedade de problemas de preparação para entrevistas de emprego na área da programação. Existem categorias de problemas redirecionados para entrevistas a grandes empresas da área como por exemplo Google, Facebook, Amazon, Microsoft, Apple entre outras. Oferece também sessões de *mock interviews* para utilizadores com subscrição *premium* na tentativa de simular o ambiente de pressão de uma entrevista.

Para além das categorias de preparação focadas em empresas possuí também uma variedade de tópicos para aprendizagem genérica de Algoritmos e Estruturas de Dados. Até à atualidade a plataforma disponibiliza um total de 816 problemas para resolução. [2]

Na Figura 1 podem ser observadas algumas das categorias anteriormente referidas.



Figura 1 - Categorias da plataforma LeetCode

Apesar da plataforma conter todo um foco para preparação e aprendizagem não deixa aquém na parte competitiva. São realizados torneios semanais regularmente e cada um destes torneios conta com um conjunto de 4 questões de dificuldade progressiva com uma duração total de 90 minutos.

Outro dos pontos de inovação desta plataforma é a possibilidade de um utilizador poder simular a realização de um torneio que já decorreu.

A plataforma contém um editor de texto *online*, proporcionado pela *framework Codemirror* que pretende simular a existência de um *IDE*. Esta *framework* processa também *syntax* de acordo com a linguagem escolhida o que torna o editor mais apelativo e o código mais legível, possuí ainda algumas características básicas de *code completion*.

Existe também a possibilidade de escolher o modo do editor de texto: *Vim, Emacs* ou Normal e escolher entre uma variedade de temas disponibilizados.

A plataforma disponibiliza compiladores para as seguintes linguagens: C++, Java, Python, Python 3, C, C#, Javascript, Ruby, Swift, Go, Scala e Kotlin. Não disponibiliza no entanto qualquer informação sobre se utiliza um serviço de *APIs* para a compilação e avaliação dos problemas como por exemplo o *Sphere Engine* ou se contém uma *API* proprietária.

No geral, esta aplicação oferece uma interface intuitiva e fácil de utilizar com uma enorme variedade e qualidade de problemas para qualquer utilizador que tenha a curiosidade e/ou necessidade de estudar as áreas propostas.

### HackerRank

A plataforma *HackerRank* [3] é até á data a plataforma com mais utilizadores registados na sua *leaderboard* (inserir citação para anexo). Comparativamente com o *LeetCode* esta plataforma contém algum foco na preparação para entrevistas e na procura de trabalho, sendo usada a nível empresarial para realização de entrevistas técnicas.



Figura 2 - Landing page da plataforma HackerRank

A oferta de problemas tipo estende-se a várias categorias como por exemplo Algoritmos, Estruturas de Dados, Inteligência Artificial, Base de Dados entre outras.

No que diz respeito a nível competitivo a oferta é bem mais variada pela oferta de problemas de duração semanal, *Week of Code*, que se realizam mensalmente, pelas competições de duração de 60 minutos, *Hour of Code*, também realizadas mensalmente, para além de todas as outras competições e *Hackathons* promovidos por empresas.

A plataforma contém um editor de texto *online*, proporcionado também pela *framework Codemirror* [4] não variando muito a sua customização relativamente á plataforma anterior.

A lista de compiladores disponíveis por esta plataforma é bastante extensiva, oferecendo BASH, C, Clojure, C++, C++14, C#, D, Erlang, F#, Go, Groovy, Haskell, Java 7, Java 8, Javascript, LOLCode, Lua, Objective-C, OCaml, Pascal, Perl, PHP, Python 2, Python 3, R, Racket, Ruby, Rust, Scala, Swift e VB.Net. Estima-se que a plataforma recorra a um serviço de compilação e avaliação externo e não a software proprietário dada a variedade de compiladores disponibilizados.

Esta plataforma possibilita a criação e gestão de torneios por parte dos utilizadores algo que não é possível na maioria das plataformas estudadas, no entanto estes torneios são sempre públicos e estão disponíveis para todos os utilizadores.

No geral esta plataforma obteve a sua popularidade pela facilidade com que um utilizador novato se pode introduzir no mundo da programação competitiva, quer pelo seguimento de problemas aconselhados ou pela regularidade de torneios mais ‘amigáveis’.

## Análise crítica das soluções existentes

Ambas as soluções apresentadas são excelentes soluções que abrangem na sua maioria os objetivos propostos para a criação da nova solução.

A plataforma *LeetCode* tem como principais pontos de destaque o seu foco para preparação de entrevistas e aprendizagem, no entanto não possibilita ao utilizador a criação de torneios e a diversidade de torneios é inferior ao *HackerRank*.

O *HackerRank* é uma solução bastante sólida e é a plataforma que mais se assemelha aos objetivos pretendidos, no entanto a funcionalidade de criação de torneios não permite que sejam classificados como privados.

A criação de torneios privados é um dos principais problemas que a nova solução pretende colmatar. Fornecerá uma maior facilidade ao utilizador por exemplo em situações de realização de *Hackathons*/competições *onsite* e também poderá ser utilizado como um método de avaliação e/ou preparação de alunos para disciplinas envolvendo programação.

Tendo em conta que um dos desafios da plataforma passa pela compilação e avaliação do código submetido pelos utilizadores, será feita uma breve análise do principal serviço da área. O serviço *Sphere Engine* [5]é utilizado por várias aplicações (*ex.: ideone, codechef*)*,* para realizar a compilação e execução das submissões de utilizadores. É um serviço pago com foco em prestação de serviços a nível empresarial. A oferta de compiladores é extremamente extensiva abrangendo 66 linguagens [6].

As principais características que o serviço fornece são:

1. Compilação de submissões
2. Execução de casos de teste relativos a uma submissão
3. Avaliação parcial das submissões
4. Limitações de execução para tempo de CPU e memória
5. Análise da complexidade dos algoritmos

Espera-se que a solução apresentada seja capaz de realizar os 4 primeiros pontos listados anteriormente.

https://www.quora.com/What-is-the-most-popular-platform-in-competitive-programming

# Metodologia

Uma metodologia de *software* é um conjunto de atividadesque tem um papel fundamental na estruturação, planeamento e controlo do processo de desenvolvimento de *software*. É o seguimento de uma metodologia que permite que o desenvolvimento seja realizado de forma focada para com as exigências do cliente e com um maior grau de organização e qualidade.

A metodologia Ágil surgiu com o objetivo de trazer mais flexibilidade para o processo de planeamento e desenvolvimento de *software* relativamente às metodologias clássicas (ex.: *Waterfall*), nas quais, os seus processos rígidos e pouco iterativos dificultam os ambientes que necessitam de atualização constante, quer por parte da introdução de novas tecnologias, ou por exigências do cliente.

Sendo assim, a metodologia Ágil caracteriza-se, segundo o seu manifesto [7] por priorizar:

* Indivíduos e interações sobre processos e ferramentas – Existe a participação de todos os intervenientes do processo (designers, gestores de projeto e produto, programadores, utilizadores, etc.) a cada iteração.
* *Software* funcional sobre documentação – A implementação dos requisitos e a sua validação por testes é preferível à extensiva realização de documentação
* Colaboração entre clientes sobre negociação de contratos.
* Responder e adaptar às mudanças sobre seguir a rigidez do plano – As mudanças que se consideram como importantes para a evolução do projeto devem ser aplicadas mesmo que não façam parte do plano atual.

Dentro do espetro da metodologia Ágil optou-se pela utilização do modelo Scrum.

## **Metodologia de desenvolvimento: Scrum**

O Scrum é uma *framework* baseada em metodologia Ágil que se destaca pela grande iteratividade e possibilidade da realização de um processo incremental. É um modelo eficaz quando ainda não está claro o que se pretende desenvolver na totalidade do projeto. O funcionamento deste modelo pode ser observado pela Figura 3 e descreve-se como o seguinte:

1. Os requisitos obtidos pelo estudo do mercado e a interação com o cliente são armazenados (*Product Backlog*) de forma ordenada de acordo com a sua prioridade.
2. São selecionados *n* requisitos do *Product Backlog* para o planeamento de um Sprint (período de tempo no qual a equipa se propõe a realizar o que for definido) de acordo com a sua prioridade e métricas (ex.: tempo de implementação) obtidas por experiência passada. Os Sprints duram tipicamente entre 1 a 4 semanas. Nesta fase determina-se essencialmente quem faz o quê.
3. Ao longo do Sprint são realizadas reuniões diárias de curta duração e informais, envolvendo toda a equipa onde cada interveniente deve procurar responder às seguintes questões:
   1. “O que fiz desde a última reunião?”
   2. “O que pretendo ter feito até à próxima reunião?”
   3. “Que dificuldades estou a ter?”

Estas questões pretendem transmitir à equipa o estado do trabalho atual, planear trabalho futuro e criar um espírito de entreajuda para situações de dificuldade.

1. No fim de cada Sprint é feita uma revisão para avaliar quais os objetivos cumpridos. Por norma, a equipa demonstra as funcionalidades implementadas. É também importante nesta fase realizar uma retrospetiva tendo em vista analisar o que pode ser melhorado para Sprints futuros.
2. Cada Sprint contará como um incremento no avanço do projeto.

O processo descrito realiza-se várias vezes até que o cliente se encontre satisfeito com o resultado.



Figura 3 - Diagrama de funcionamento do Scrum, Fonte [7]

A escolha do Scrum para a utilização como metodologia de planeamento e desenvolvimento resultou das incógnitas inicias associadas ao rumo de alguns módulos da aplicação e da pouca experiência inicial de algumas das tecnologias. A realização de reuniões semanais com o orientador é também um facto que suporta o uso deste modelo visto que simula as revisões e o curto período de tempo de um Sprint. Nestas reuniões pretende-se apresentar e avaliar o que foi feito e planear o que será feito na próxima semana.

(FALAR DO TRELLO)

# Funcionalidades

A aplicação apresentada terá que possibilitar a interação com 2 tipos de utilizadores. O ‘Programador’ que será o principal utilizador a interagir com o sistema e o ‘Gestor de Conteúdos’ que será o utilizador com mais permissões e um maior leque de funcionalidades.

O Programador deverá poder realizar as seguintes ações:

* Consultar categorias de problemas práticos
* Consultar lista de problemas associados a uma categoria
* Submeter solução(código) para o problema
* Consultar classificação global
* Consultar classificação de torneio
* Consultar submissões para determinado problema
* Consultar torneios
* Participar em torneios
* Adicionar e gerir torneio
* Adicionar e gerir problemas de um torneio
* Adicionar e gerir casos de teste de um problema

O Gestor de Conteúdos deverá poder realizar as seguintes ações:

* Todas as funcionalidades do utilizador programador.
* Gerir categorias
* Gerir problemas
* Gerir casos de teste
* Gerir propostas de torneios

# Bibliografia

|  |  |
| --- | --- |
| [1] | “LeetCode Landing Page,” [Online]. Available: https://leetcode.com/. [Acedido em 12 Junho 2018]. |
| [2] | “Leet Code Problem Set,” [Online]. Available: https://leetcode.com/problemset/all/. [Acedido em 13 Junho 2018]. |
| [3] | “HackerRank Landing Page,” [Online]. Available: https://www.hackerrank.com/. [Acedido em 12 Junho 2018]. |
| [4] | “Codemirror Landing Page,” [Online]. Available: https://codemirror.net/. [Acedido em 12 Junho 2018]. |
| [5] | “Sphere Engine Landing Page,” [Online]. Available: https://sphere-engine.com/. [Acedido em 13 Junho 2018]. |
| [6] | “Sphere Engine compiler support,” [Online]. Available: https://sphere-engine.com/demo/1-online-compiler. [Acedido em 13 Junho 2018]. |