

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

JEFERSON AFONSO DO PATROCÍNIO
Orientador: Prof. Dr. Reinaldo Silva Fortes

**OPCODERS JUDGE: UMA VERSÃO ONLINE PARA O CORRETOR
AUTOMÁTICO DE EXERCÍCIOS DE PROGRAMAÇÃO DO PROJETO
OPCODERS**

Ouro Preto, MG
2023

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

JEFERSON AFONSO DO PATROCÍNIO

**OPCODERS JUDGE: UMA VERSÃO ONLINE PARA O CORRETOR AUTOMÁTICO
DE EXERCÍCIOS DE PROGRAMAÇÃO DO PROJETO OPCODERS**

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Reinaldo Silva Fortes

Ouro Preto, MG
2023



FOLHA DE APROVAÇÃO

Jeferson Afonso do Patrocínio

Uma versão online para o corretor automático de exercícios de programação do projeto opCoders

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 27 de Março de 2023.

Membros da banca

Reinaldo Silva Fortes (Orientador) - Doutor - Universidade Federal de Ouro Preto
Valéria de Carvalho Santos (Examinadora) - Doutora - Universidade Federal de Ouro Preto
Marcelo Luiz Silva (Examinador) - Mestre - Universidade Federal de Ouro Preto

Reinaldo Silva Fortes, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 27/03/2023.



Documento assinado eletronicamente por **Reinaldo Silva Fortes, PROFESSOR DE MAGISTERIO SUPERIOR**, em 28/03/2023, às 10:13, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0493053** e o código CRC **6D5D2809**.

Agradecimentos

Agradeço, primeiramente, a Deus por me dar força e sabedoria para concluir essa etapa tão importante em minha vida! A minha querida mãe Pilar, pela luta para me educar e ensinar os princípios que me tornaram o homem que hoje sou. À minha amada esposa Jaqueline que tanto me apoiou nessa etapa de minha vida e também pela compreensão da minha ausência durante os momentos de estudos. À minha doce filha Júlia que me deu um grande propósito para concluir os meus estudos. E a todos os meus familiares que depositaram sua confiança em mim o que tanto me incentivou.

Agradeço também ao professor Reinaldo pela oportunidade e confiança depositada em mim nesse projeto. Suas orientações e ensinamentos foram fundamentais para possibilitar esse passo tão importante. Ao Núcleo de Tecnologia da Informação pela flexibilidade que viabilizou a realização de minha graduação, ao Departamento de Computação e à Universidade Federal de Ouro Preto pelo ensino público, gratuito e de qualidade.

Resumo

Este projeto é a continuação de um projeto anterior desenvolvido pela graduanda de 2019 Palloma Stéphanne Silva Brito e tem como objetivo avaliar e aprimorar o módulo web do corretor automático de exercícios de programação de computadores existente atualmente na UFOP, tornando-o assim, capaz de além de auxiliar no processo de aprendizagem de programação, fornecendo a correção da atividade submetida pelo aluno em tempo reduzido, auxiliar o processo de gestão e correção de atividades pelos professores, que possuem turmas com um número consideravelmente grande de alunos. Pretende-se concluir e aprimorar a interface com o usuário, disponibilizando mais recursos para os alunos e criando o módulo administrativo para os professores. A interface web já existente atendia precariamente a ambos os usuários, mas era limitada pelo fato de sua não finalização. Havia recursos para os alunos, mas frequentemente oferecia pouco no módulo administrativo; assim, os professores precisavam configurar as tarefas diretamente no banco de dados, acessando o servidor via *ssh* para registro de atividades e relatórios de notas, além de atualizações diretas no banco de dados. Metodologicamente, configurou-se um ambiente de desenvolvimento, a partir do repositório do projeto no GitHub. Foram usadas as seguintes tecnologias: PHP, MySQL e Bootstrap. Através desse projeto, propõe-se aumentar a motivação de alunos e professores nas disciplinas de programação tão essenciais nas engenharias e em outros campos do conhecimento.

Palavras-chave: Corretor automático de código. Programação em cursos de engenharia. Juízes online. Desenvolvimento web.

Abstract

This project is the continuation of a previous one developed by the undergraduate student Palloma Stéphanne Silva Brito, class of 2019. It aims to evaluate and better the web module automatic corrector currently applied at UFOP (Federal University of Ouro Preto), making it able to help managing and the activities correction by the professors with lots of students in their classes, besides helping the learning process of programming, providing the activities correction submitted by the student in a short time. It is intended to conclude and improve the user interface, providing more resources for students and making the administrative module for the professors. The current web interface precariously served both users, but it was limited by its not conclusion. There were resources for the students, but not enough in the administrative module. That is why the professors had to set up the tasks directly on the database, accessing the server via ssh for activities log and grade reports. In terms of methodology, a development environment was set up from the project repository cited on GitHub. The following technologies were used: PHP, MySql and Bootstrap. This project proposes to motivate more the students and professors in the programming subjects that are essential for engineering courses and other areas of knowledge.

Keywords: Code automatic corrector, Programming at Engineering Courses, Online Judges, Web development.

Lista de Ilustrações

Figura 3.1 – Versão final do mapa do site <i>opCoders Judge</i>	10
Figura 3.2 – Versão preliminar do mapa do site <i>opCoders Judge</i>	11
Figura 3.3 – Caso geral de uso	12
Figura 3.4 – Cadastrando um novo semestre	12
Figura 3.5 – Editando um semestre	13
Figura 3.6 – Cadastrando uma nova disciplina	13
Figura 3.7 – Editando uma disciplina	13
Figura 3.8 – Cadastrando uma nova turma	14
Figura 3.9 – Editando uma turma	14
Figura 3.10–Tela de gerenciamento de estratégias de correção	15
Figura 3.11–Tela de gerenciamento de usuários	15
Figura 3.12–Versão preliminar da tela do banco de questões	16
Figura 3.13–Versão final da tela do banco de questões	16
Figura 3.14–Versão preliminar da tela do banco de questões	17
Figura 3.15–Versão final da tela do banco de questões	17
Figura 3.16–tela de gestão de alunos e exportação de notas	18
Figura 3.17–tela com a listagem das entregas feitas pelo aluno	18
Figura 3.18–Página minhas tarefas	19
Figura 3.19–Página de detalhes da tarefa	19
Figura 3.20–Versão preliminar da página de entrega de questão	19
Figura 3.21–Versão final da página de entrega de questão	20
Figura 3.22–Página de visualização correção e nota	20
Figura 3.23–Página perfil do usuário	20
Figura 3.24–Diagrama do enquadramento das classes do projeto no padrão MVC	21
Figura 3.25–Versão preliminar do diagrama ER - Entidade relacionamento do banco de dados	22
Figura 3.26–Versão final do diagrama ER - Entidade relacionamento do banco de dados	23

Sumário

1	Introdução	1
1.1	Justificativa	2
1.2	Objetivos	2
1.3	Estrutura da Monografia	2
2	Revisão Bibliográfica	4
2.1	O uso de ferramentas computacionais para o ensino de programação para alunos de engenharia	4
2.2	Corretores automáticos de código	4
2.3	Tecnologias usadas no projeto	5
2.3.1	PHP	5
2.3.2	MySQL	5
2.3.3	Bootstrap	5
2.3.4	Padrão de projeto	5
2.3.5	Engenharia de requisitos	6
2.3.5.1	Requisitos funcionais e não funcionais	7
2.3.6	Scrum	7
2.4	Trabalhos relacionados	7
2.4.1	JOnline: proposta preliminar de um juiz online didático para o ensino de programação	7
2.4.2	Um Ambiente Virtual com <i>Feedback</i> Personalizado para Apoio a Disciplinas de Programação	8
3	Metodologia e Desenvolvimento	9
3.1	Visão Geral	10
3.2	Casos de Uso	11
3.2.1	Funções disponíveis para o administrador do sistema	11
3.2.1.1	Configurações	11
3.2.1.1.1	Semestres	12
3.2.1.1.2	Disciplinas	13
3.2.1.1.3	Turmas	14
3.2.1.1.4	Estratégia de correção	15
3.2.1.2	Gestão de usuários	15
3.2.2	Funções disponíveis para o professor	16
3.2.2.1	Banco de questões	16
3.2.2.2	Banco de Tarefas	16
3.2.2.3	Gestão de Alunos	17
3.2.3	Funções disponíveis para o aluno	18

3.3	Enquadramento da abordagem MVC no projeto	20
3.3.1	Camada controle	21
3.3.2	Camada modelo	21
3.4	O Banco de dados	22
4	Considerações Finais	24
4.1	Conclusão	24
4.2	Trabalhos Futuros	24
	Referências	26

1 Introdução

Em variados cursos de graduação, da área de ciências exatas, a disciplina de Algoritmos e Programação faz parte da grade curricular obrigatória, com o objetivo de desenvolver o raciocínio lógico de que o aluno vai necessitar durante o curso. Estas disciplinas são consideradas de grande dificuldade para a maior parte dos alunos, muitas vezes devido ao alto grau de abstração que deve ser aplicado. Durante essa fase, enquanto os alunos desenvolvem suas primeiras linhas de código, é importante que eles recebam um *feedback* a respeito do código escrito por eles estar correto o mais rápido possível. Para isso foi proposto o corretor de exercícios de programação *opCoders Judge*, no qual aluno pode submeter sua atividade e receber uma resposta em tempo reduzido e de forma contínua ao longo do semestre letivo.

O corretor citado foi desenvolvido em duas etapas, a primeira ocorrendo em 2019 por Brito (2019), graduanda naquele ano no curso Ciências da Computação na Universidade Federal de Ouro Preto (UFOP). Na primeira etapa, foi implementada uma versão offline para a correção das tarefas dos alunos. A gestão das tarefas era feita pelo Moodle, os alunos tinham acesso ao enunciado em arquivo de formato PDF e entregavam suas soluções em arquivos de código-fonte que, posteriormente eram baixados em máquina local do professor e corrigidos automaticamente. A resposta aos alunos ocorria pela geração e envio de um arquivo, também em formato PDF, contendo o resultado da avaliação.

A segunda etapa constitui-se da implementação de uma versão Web, onde o aluno poderia visualizar os enunciados em formato de página Web, fazer a entrega do código-fonte e visualizar o resultado também em formato HTML. Outra vantagem é que a correção passou a ser online, em poucos segundos após o envio da resposta o aluno já possuiria acesso ao resultado da correção automática. Esta etapa foi iniciada pelo aluno da graduação, Guilherme Augusto Rodrigues Melo, orientado pelo professor Reinaldo Silva Fortes e com a colaboração do professor Rafael Antônio Marques Gomes. Porém, o aluno Guilherme não concluiu a primeira etapa em seu trabalho de monografia com uma versão completa, contendo apenas algumas funcionalidades básicas. Sendo assim, nesse presente estudo de monografia pretende-se concluir a segunda etapa com uma versão Web completa.

A implementação obtida no presente estudo já está em produção, com as funcionalidades necessárias já implementadas, possibilitando desde a importação dos usuários, passando pela criação de atividades pelo docente e entregas de soluções pelo discente, até o relatório de entregas contendo as notas dos alunos¹.

A adesão ao padrão de projeto Model-View-Controller (MVC) e a abordagem para a solução dos problemas apresentados facilitaram a extensão das funcionalidades do *opCoders*

¹ <https://bcc701.decom.ufop.br/>

Jugde, pois independentemente da linguagem de programação usada, o fato de separar as camadas de modelo de dados, controle e visualização torna mais fácil e rápido a manutenção e ampliação do software, devido a grande possibilidade de reaproveitamento de código que essa abordagem provê (GAMMA, 2009).

Nas Subseções a seguir, apresenta-se a justificativa, os objetivos e a estrutura da monografia.

1.1 Justificativa

Disciplinas de programação exigem a compreensão da abstração do problema proposto na atividade para que o aluno desenvolva o respectivo código para solucioná-lo. Além da semântica é necessário também que a lógica esteja correta para a obtenção do resultado esperado para determinada entrada dada. Receber esse *feedback* em tempo reduzido é importante para que o aluno esteja engajado com a disciplina, além de acelerar o processo de aprendizagem. Quanto ao professor, é importante uma ferramenta que o auxilie na gestão de atividades e turmas melhorando a metodologia didática utilizada no ensino.

Além disso, o uso de novas ferramentas nesse campo pode reduzir a evasão e desestímulo, o que justifica essa segunda etapa do projeto.

1.2 Objetivos

O objetivo desse estudo é avaliar e aprimorar o módulo web do corretor automático existente atualmente na UFOP, tornando-o, capaz de auxiliar no processo de aprendizagem de programação, através da correção das atividades submetidas pelos alunos e do auxílio no processo de gestão e correção de atividades pelos professores. São objetivos específicos:

- Realizar ajustes e correções nas páginas existentes;
- Fornecer uma interface web funcional e de fácil utilização para que alunos possam submeter soluções de problemas e visualizar o resultado de correção;
- Fornecer aos professores uma interface web funcional e prática onde seja possível editar de forma fácil as questões, realizar agendamento das questões e visualizar os resultados das entregas dos alunos.

1.3 Estrutura da Monografia

No Capítulo 1, apresenta-se de forma geral, o que o contém neste projeto, incluindo a justificativa. No Capítulo 2, apresenta-se a revisão bibliográfica com os principais autores que

foram analisados no decorrer do projeto No Capítulo 3, apresentam-se os processos metodológicos para o desenvolvimento do projeto. E, por fim, no Capítulo 4, foram trazidas as principais conclusões obtidas na aplicação do projeto, bem como trabalhos futuros.

2 Revisão Bibliográfica

Através de portais de referências, como Google Acadêmico, portal de periódicos CAPES e Scielo, buscaram-se autores vinculados aos temas analisados nesse projeto. As palavras chave usadas para a busca foram: corretor automático de código, programação Web, linguagem de programação e padrão de projetos de software. O recorte temporal foi de 2015 a 2022, admitindo-se autores mais antigos caso fosse necessário.

Para fins de organização, esta revisão foi dividida em tópicos relacionados a corretores específicos, e tecnologias usadas no projeto.

2.1 O uso de ferramentas computacionais para o ensino de programação para alunos de engenharia

Ferramentas computacionais são frequentes em diversos cursos de graduação e pós-graduação em diversas áreas do conhecimento. Segundo PALHARES, SANTOS e VASCONCELOS (2017), muitas vezes o uso das ferramentas computacionais pode promover o efeito contrário nos estudantes, visto que nem sempre eles se adaptam ou conhecem os softwares. Todavia essas ferramentas possuem diversos benefícios tanto para os alunos quanto para os professores.

Diante da evasão que ocorre normalmente em cursos em que a programação é essencial, Nazário e Souza (2010, p. 43), afirmam que a contribuição dos corretores pode ser muito útil, uma vez que proporciona ao estudante comodidade, facilidade e agilidade, o que torna o aluno mais motivado com o aprendizado. Segundo os autores “A disciplina de programação possui um dos maiores índices de reprovação nas instituições brasileiras” e “o processo de aprendizagem nas disciplinas de programação está diretamente ligado à motivação do aluno”. Ocorre também, diante da situação de os alunos reprovados na disciplina promoverem a superlotação de salas de aula.

O uso das ferramentas computacionais, portanto, é considerado recurso didático para que aluno e professor se sintam motivados em temas tão áridos nas engenharias.

2.2 Corretores automáticos de código

Os corretores automáticos de código tem como função a validação da consistência de um algoritmo através de diversos casos de teste (NAZÁRIO; SOUZA, 2010). Esses corretores são também conhecidos como Juízes Online. Segundo Júnior et al. (2020, p. 13), os JO - Juízes Online são “sistemas de correção automático de códigos que vem sendo bastante utilizados em treinamentos empresariais, na área de educação e para áreas afins”. Segundo o autor, instituições

de ensino têm usado essas plataformas para turmas de programação. Nos JOs torna-se disponível um acervo de problemas de variados tópicos e níveis de dificuldade que podem ser resolvidos pelo ambiente de desenvolvimento integrado. Esses corretores ou juízes contribuem para a satisfação do usuário em resolver questões de programação.

Segundo Brito (2019), corretores automáticos de código, ou Juízes-online, são ferramentas acessíveis na Internet que permitem automatizar o processo de correção de exercícios de programação. Esses ambientes possuem a descrição de exercícios com entradas e saídas já pré-definidas e abordam os mais variados temas relacionados a programação.

2.3 Tecnologias usadas no projeto

Nesse tópico serão relacionadas as tecnologias utilizadas nesse projeto.

2.3.1 PHP

O PHP é uma linguagem de programação que nasceu para possibilitar o pré-processamento de páginas HTML. Dessa forma o PHP consegue alterar o conteúdo de uma página ou até mesmo criar uma página nova a cada requisição feita pelo cliente. Além disso o PHP permite capturar dados em formulários criadas em páginas HTML (BENTO, 2021). Também o PHP é uma linguagem de código aberto e pode ser usado de forma gratuita e toda sua documentação está disponível no site do projeto.

2.3.2 MySQL

O Mysql é um SGDB (Sistema de Gerenciamento de Banco de Dados) relacional que utiliza a linguagem SQL como interface. Nele as informações são guardadas em estruturas chamadas tabelas, onde cada coluna representa um atributo da mesma (BENTO, 2021).

2.3.3 Bootstrap

O Bootstrap é um framework *open-source* para desenvolvimento front-end. Através das tecnologias HTML, CSS e Javascript, permite a desenvolvedores a construção de aplicativos e websites robustos e de forma responsiva, considerando em primeiro lugar a plataforma dos dispositivos móveis, além de possuir uma grande quantidade de componentes personalizados (BOOTSTRAP, 2022).

2.3.4 Padrão de projeto

Segundo Gamma (2009, p. 20), “um padrão de projeto nomeia, abstrai e identifica os aspectos-chave de uma estrutura de um projeto comum para torná-la útil na criação de um projeto

com objetos reutilizáveis”. Em cada padrão é descrito um problema e são apresentadas uma ou mais alternativas para a solução, mas sempre focando o mesmo objetivo.

Em geral, um padrão tem quatro elementos essenciais: (i) o nome do padrão (ii) o problema (iii) a solução e (iv) as consequências. O nome do padrão é a referência que se usará para a descrição do problema do projeto, e suas soluções em poucas palavras em geral, de uma a três. O problema é a situação em que se quer aplicar o padrão. A solução, como o próprio nome afirma, descreve elementos do padrão, suas responsabilidades e colaborações. As consequências são os resultados e as análises a partir das vantagens e desvantagens do padrão.

O padrão de projeto adotado nesse estudo é o MVC, que tem como objetivo isolar ao máximo a camada de apresentação da camada de lógica de negócio, isso provê grande possibilidade de reaproveitamento de código.

Segundo [Gamma \(2009\)](#), a abordagem MVC é composta por três tipos de objetos: O **Modelo**, a **Visão** e o **Controlador**. O Modelo é o objeto de aplicação, ou seja é a parte que é responsável pela modelagem dos dados e a parte lógica dos processos. A Visão é responsável pela apresentação na tela das interfaces para os usuários. O Controlador define a forma como a interface do usuário reage às suas entradas. É uma intermediação entre o modelo e a visão. Antes do MVC os projetos de interface tendiam a agrupar esses objetos, desfavorecendo a reutilização de código.

2.3.5 Engenharia de requisitos

Baseado em [Pressman e Maxim \(2021, p. 103\)](#), a engenharia de requisitos é um elo entre o projeto e a construção do software, nela são descritos as necessidades do negócio, os cenários de usuários, esboçados as funções e recursos e também identificadas as restrições do projeto. A Engenharia de Requisitos possui as seguintes etapas:

- **Concepção** - etapa em que é estabelecido um entendimento do problema, das pessoas que querem uma solução e da natureza da solução pretendida;
- **Levantamento** - etapa em que questiona os envolvidos sobre quais os objetivos do sistema;
- **Elaboração** - etapa onde são refinados e criados os cenários de usuários obtidos durante a etapa de levantamento;
- **Negociação** - etapa onde são feitos acordos dos requisitos levantados pelos clientes, priorizando-se a ordem de implementação dos requisitos;
- **Especificação** - geralmente as especificações são documentos por escrito, um conjunto de gráficos, modelos matemáticos, conjunto de cenários de uso, protótipos ou combinações desses fatores;

- **Validação** - nessa etapa é examinada a especificação para garantir que os requisitos declarados tenham sido atendidos.

2.3.5.1 Requisitos funcionais e não funcionais

Segundo Pressman e Maxim (2021), um requisito não funcional pode ser definido como um atributo de qualidade, segurança, desempenho e afins. Também incluem-se requisitos lógicos e de dados. Por sua vez, os requisitos funcionais são aqueles que representam a finalidade do sistema.

2.3.6 Scrum

Segundo SUTHERLAND e SCHWABER (2013), o Scrum é um *framework* estrutural em que é possível empregar vários processos ou técnicas na prática de gerenciamento e desenvolvimento de produtos. O *framework* Scrum consiste nos times associados a papéis, eventos, artefatos e regras, todos esses componentes têm um propósito para o sucesso de uso do Scrum.

O item mais importante do *Scrum* é a *Sprint*, que sua definição é um período de tempo fixo e curto, no qual a equipe trabalha para executar uma quantidade definida de tarefas, essas tarefas são definidas durante a *Sprint Planning* que é basicamente uma reunião onde é feito todo o planejamento do que será realizado naquela *Sprint*, e o resultado da execução das mesmas é apresentado na *Sprint Review*. Os detalhes da aplicação do Scrum no desenvolvimento desse estudo são apresentados no Capítulo 3.

2.4 Trabalhos relacionados

Nesta seção, faz-se a descrição de dois trabalhos relacionados ao tema deste projeto e uma comparação de suas propostas.

2.4.1 JOnline: proposta preliminar de um juiz online didático para o ensino de programação

Este artigo, de autoria de Santos e Ribeiro (2012) apresentado no XXII SBIE - Simpósio Brasileiro de Informática na Educação, 2011, registra a importância do aprendizado de programação para a formação de profissionais na área de computação. No cenário acadêmico, é possível o desenvolvimento de ambientes online para a facilitação do aprendizado. Os autores apresentam juizes online como essenciais para avaliação de códigos fonte por usuários diversos. Registram como principais funcionalidades a apresentação de dicas em língua portuguesa para correção de possíveis erros de compilação do código fonte, apresentação de casos de testes que geram os resultados considerados errados a organização da atividades por nível de dificuldade e votação do nível de dificuldade pelos usuários. Salienta-se a funcionalidade da programação colaborativa.

2.4.2 Um Ambiente Virtual com *Feedback* Personalizado para Apoio a Disciplinas de Programação

Alves e Jaques (2014) apresentam um sistema nomeado como *feeper*, acrônimo das palavras “*feedback* personalizado”. O *feeper* é uma ferramenta online com o propósito de apoiar o aprendizado de programação, seja em atividades em classe ou extraclasse, onde os exercícios de programação são disponibilizados em ordem de dificuldade crescente de forma gradual. Nesse projeto, o aluno utiliza um editor de código fonte integrado no sistema e, ao concluir, pode submeter o código para verificação. Caso ocorra um erro de compilação, este é capturado e exibido ao aluno e, quando ocorre um erro de execução, uma mensagem personalizada feita pelo professor é exibida.

3 Metodologia e Desenvolvimento

No trabalho apresentado por Brito (2019) foi implementado um corretor automático de atividades de programação para os alunos de engenharia da UFOP. Como resultado foi implementado um software que automatizava a correção das atividades dos alunos de forma *offline*, os resultados sendo enviados para o aluno posteriormente em formato PDF. Como trabalho futuro, a autora sugeriu a disponibilização do corretor, agora na forma *on-line*, para que o aluno pudesse receber o resultado das correções mais rapidamente, agilizando seus estudos. A implementação do corretor *on-line* foi iniciada pelo aluno Guilherme Augusto Rodrigues Melo, com a contribuição do professor Rafael Antônio Marques Gomes e sob a orientação do professor Reinaldo Silva Fortes. Como continuidade ao trabalho anteriormente desenvolvido, no estudo apresentado por este trabalho foram implementadas as funcionalidades fundamentais a fim que de fato o corretor funcione de forma intuitiva sem a necessidade de utilização de ferramentas auxiliares, ou seja, tudo está disponível através da interface *web* e dos próprios recursos do corretor.

Neste estudo foi seguida uma metodologia exploratória de cunho bibliográfico, seguida de uma pesquisa experimental, já que o código implementado foi testado para analisar se o resultado obtido estava em conformidade como esperado. A pesquisa exploratória fornece ao pesquisador o aprimoramento das ideias para melhor entendimento do problema a que ele busca investigar, e já a pesquisa experimental especifica o objeto de estudo colaborando com a construção de possíveis variáveis que controlam os resultados obtidos na aplicação do estudo (GIL et al., 2002).

Na implementação foram aplicados os princípios do Scrum. No caso em questão, o orientador Reinaldo desempenhou o papel de *Product Owner* (PO), que é responsável por definir as necessidades do cliente, gerenciar o *backlog* do produto e priorizar as tarefas a serem realizadas pela equipe de desenvolvimento. Já o orientando Jeferson atuou como equipe de desenvolvimento, responsável por transformar as demandas do PO em incrementos de software funcionais, prontos para serem entregues ao cliente.

A lista de demandas que foram elencadas compuseram o *backlog* do produto, que são todas as funcionalidades e melhorias a serem implementadas. O *backlog* é gerenciado pelo PO, definindo as prioridades do que deve ser implementado.

O Scrum utiliza eventos, como a *Sprint Review*, e a *Sprint Planning*. Esses eventos ocorreram nas reuniões semanais de orientação, onde a equipe de desenvolvimento apresentava os incrementos desenvolvidos com o intuito de receber as validações e *feedback* pelo PO. Em seguida era realizado o *Sprint Planning*, onde eram definidas as tarefas a serem realizadas na próxima *Sprint*.

Nas subseções a seguir são apresentados os resultados finais do desenvolvimento.

3.1 Visão Geral

O opCoders Judge é uma ferramenta que acrescenta uma camada adicional ao corretor automático de programação, propiciando um ambiente onde o professor pode elaborar as tarefas para suas turmas em base de questões existentes no banco de questões, bem como elaborar novas questões para os alunos. Também é possível realizar um planejamento com as datas nas quais as tarefas devem ser entregues. Para os alunos o opCoders Judge auxilia nos estudos, podendo visualizar as tarefas as quais devem submeter solução nas datas definidas e também conseguem visualizar em tempo reduzido o nível de acerto da sua solução da atividade.

Para termos uma visão geral sobre o *opCoders Judge* a Figura 3.1 apresenta o mapa completo do site, indicando quais páginas são acessíveis partindo de sua origem. As páginas do primeiro nível sempre podem ser acessadas pelo menu principal. Em título de comparação, na Figura 3.2 é apresentada a versão preliminar do mapa do site do *opCoders Judge*. Nas seções a seguir apresenta-se os casos de uso e uma breve descrição de cada página.

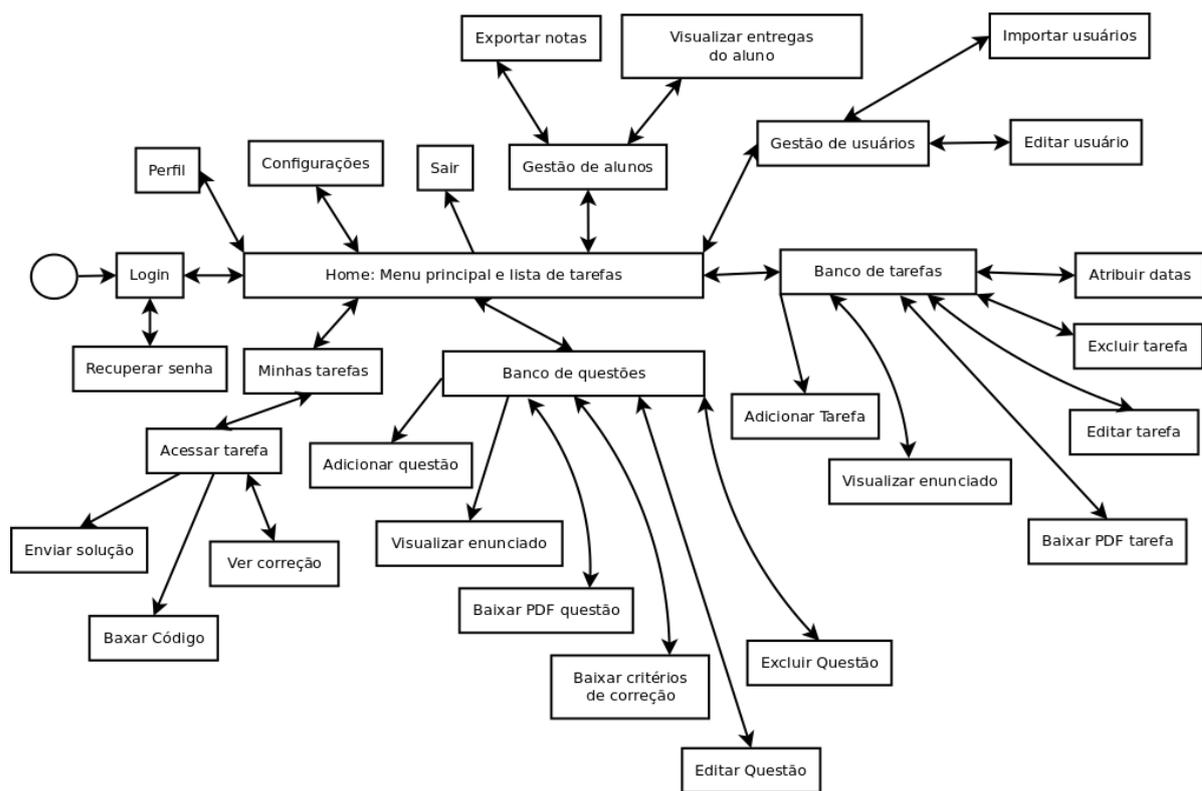


Figura 3.1 – Versão final do mapa do site *opCoders Judge*

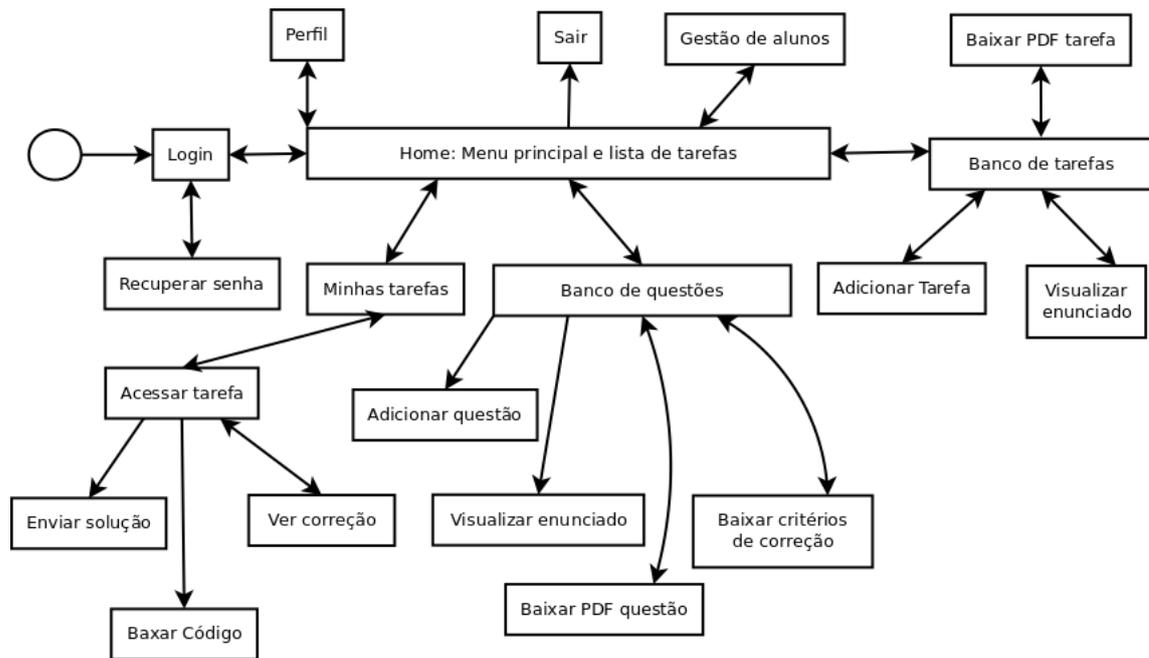


Figura 3.2 – Versão preliminar do mapa do site *opCoders Judge*

3.2 Casos de Uso

O *opCoders Judge* permite diferentes possibilidades de utilização, com base nos papéis existentes no ambiente de ensino: *Aluno*, *Professor*, *Estagiário Docente*, *Tutor*, *Monitor* e *Administrador do sistema*. As ferramentas fornecidas pelo *opCoders Judge* são disponibilizadas de acordo com as atividades que cada sujeito deve executar. Para o professor estão disponíveis os recursos de elaboração das tarefas e para os alunos recursos para realização das tarefas a eles atribuídas. Na Figura 3.3 é mostrado o diagrama do caso geral de uso do corretor. No caso, estagiário docente, tutor e monitor são considerados atores do tipo *Aluno*.

3.2.1 Funções disponíveis para o administrador do sistema

A administração do sistema é uma atribuição especial dada a um ou mais usuários. A seguir são listadas as funções disponíveis exclusivamente para esse papel.

3.2.1.1 Configurações

Essa função administrativa é usada para manutenção geral do sistema. Nela é possível cadastrar, editar e excluir itens fundamentais para o uso do corretor durante os períodos letivos: semestres, disciplinas, turmas e estratégias de correção.

Essa nova funcionalidade trouxe o benefício de não ser mais necessário acessar o banco de dados para realizar a criação e modificação de turmas, disciplinas, semestres e também estratégias de correção, tornando o sistema mais seguro e simples de ser configurado a cada semestre que se

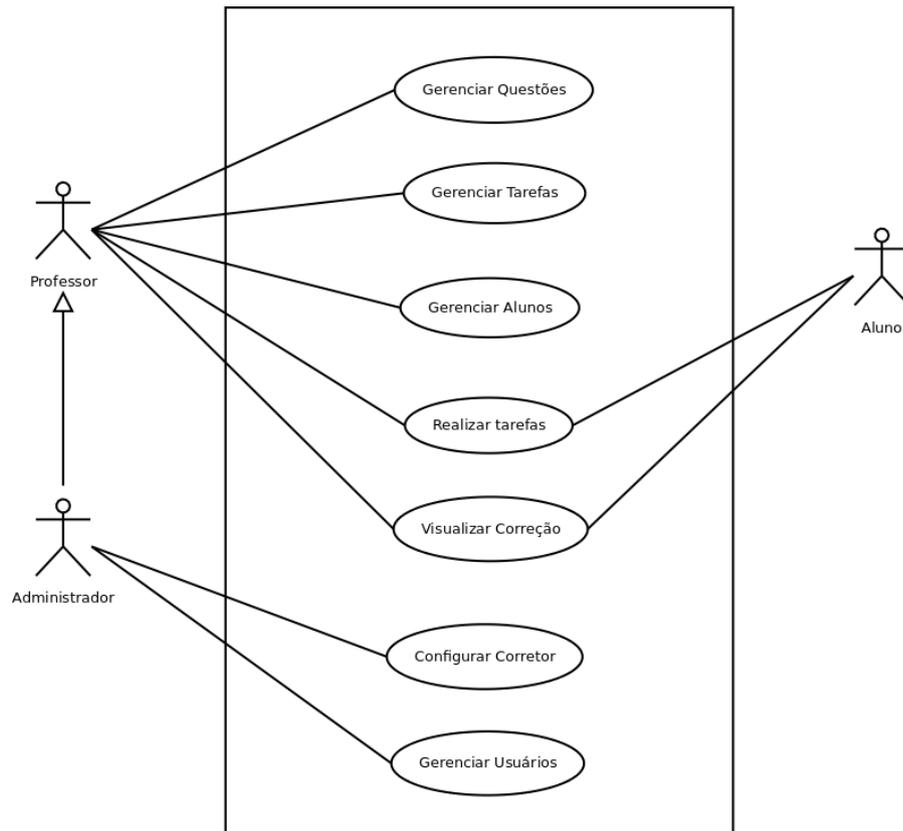


Figura 3.3 – Caso geral de uso

inicia. Assim, o executor dessa tarefa administrativa não necessita ter conhecimento técnico para cadastrar e alterar essas informações.

3.2.1.1.1 Semestres

Nessa configuração são listados os semestres cadastrados, além disso é possível cadastrar e editar um semestre, para que posteriormente sejam cadastradas turmas relacionadas a ele, por fim quando necessário é possível excluir um semestre. Na Figura 3.4 é mostrada a tela de criação de um novo semestre, e na Figura 3.5 a tela de edição do semestre.

Novo Semestre

Descrição (aaaa-p)

Ativo

SALVAR **CANCELAR**

Figura 3.4 – Cadastrando um novo semestre



Editar semestre

Descrição: 2023-1

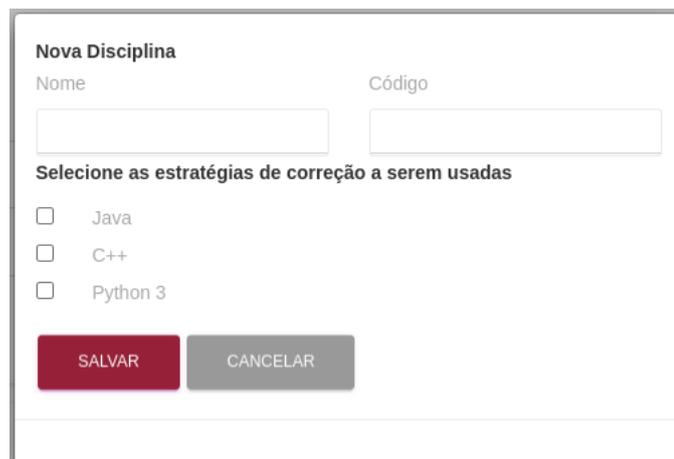
Ativo: Sim

SALVAR CANCELAR

Figura 3.5 – Editando um semestre

3.2.1.1.2 Disciplinas

Nessa configuração são listadas as disciplinas cadastradas na plataforma, também é possível cadastrar e editar uma disciplina, para que posteriormente seja possível cadastrar turmas relacionadas e ela. Quando necessário pode excluir uma disciplina selecionada. Na Figura 3.6 é mostrada a tela de criação de disciplina e na Figura 3.7 é mostrada a tela de edição de disciplina.



Nova Disciplina

Nome:

Código:

Selecione as estratégias de correção a serem usadas

- Java
- C++
- Python 3

SALVAR CANCELAR

Figura 3.6 – Cadastrando uma nova disciplina



Editar Disciplina

Descrição: Programação Orientada a Objetos

Código: BCC221

Selecione as estratégias de correção a serem usadas

- Python 3
- C++
- Java

SALVAR CANCELAR

Figura 3.7 – Editando uma disciplina

3.2.1.1.3 Turmas

Na configuração de turmas finalmente é cadastrada uma nova turma, nela são relacionados o semestre e a disciplina que devem ser cadastrados previamente, a edição também é possível, bem como a exclusão de uma turma caso seja necessário. Na Figura 3.8 é mostrada a tela de criação de turmas e na Figura 3.9 a tela de edição de uma turma.



Nova turma

Nome

Turma

Semestre

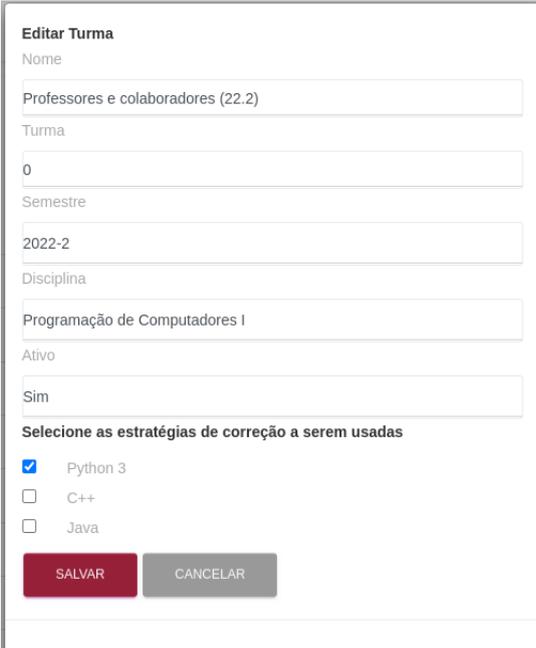
Disciplina

Ativo

Selecione as estratégias de correção a serem usadas

Java
 C++
 Python 3

Figura 3.8 – Cadastrando uma nova turma



Editar Turma

Nome

Turma

Semestre

Disciplina

Ativo

Selecione as estratégias de correção a serem usadas

Python 3
 C++
 Java

Figura 3.9 – Editando uma turma

3.2.1.1.4 Estratégia de correção

Na Figura 3.10 é mostrado o resultado de implementação do mecanismo que permite a leitura, criação, edição e remoção de registros de estratégias de correção. Nessa tela são cadastradas as as linguagens de programação que o corretor consegue lidar. É importante ressaltar que mecanismo de correção de uma determinada linguagem deve ser devidamente instalado no servidor.

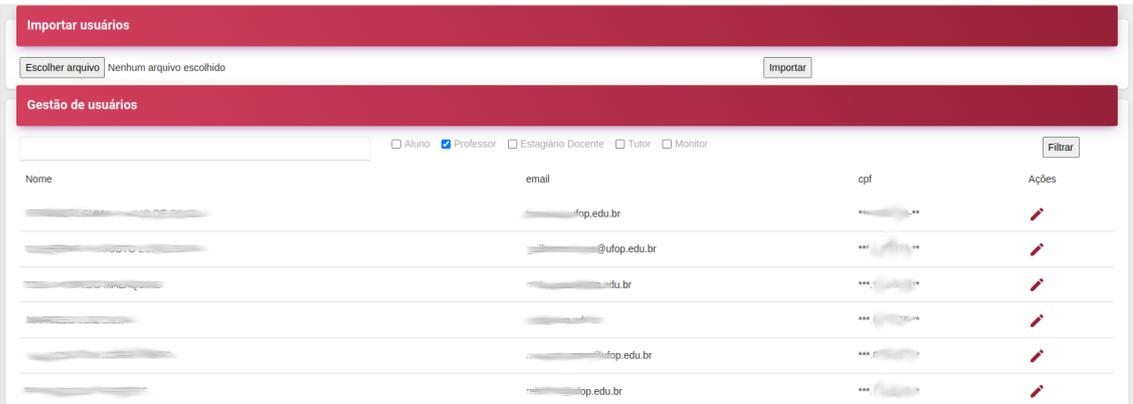


Descrição	Corretor	Formatos Aceitos	Ações
Java	corretorJava	java, zip	 
C++	corretorCplus	cpp, zip, c	 
Python 3	Corretor.py teste	py, zip, rar	 

Figura 3.10 – Tela de gerenciamento de estratégias de correção

3.2.1.2 Gestão de usuários

Função utilizada para provisionamento e edição dos usuários. O provisionamento é feito através da importação de um arquivo pré formatado contendo as informações do usuário, sendo elas *nome*, *email*, qual o *papel* daquele usuário além de uma *senha* que é gerada de forma aleatória. Uma vez importado, o *script* de provisionamento é executado e todas as informações são importadas para o banco de dados do corretor. Já a edição é possível realizar ajustes como mudar o papel do usuário e também defini-lo como administrador do sistema.



Nome	email	cpf	Ações
...	...@ufop.edu.br	***-**-****	
...	...@ufop.edu.br	***-**-****	
...	...@ufop.edu.br	***-**-****	
...	...@ufop.edu.br	***-**-****	
...	...@ufop.edu.br	***-**-****	

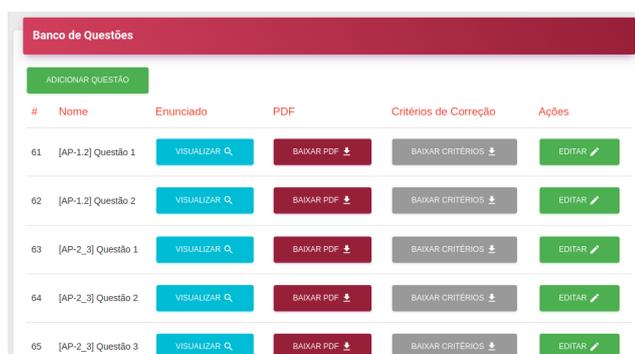
Figura 3.11 – Tela de gerenciamento de usuários

3.2.2 Funções disponíveis para o professor

O professor conta com ferramentas que possibilitam a criação e planejamento de tarefas e questões usando o corretor.

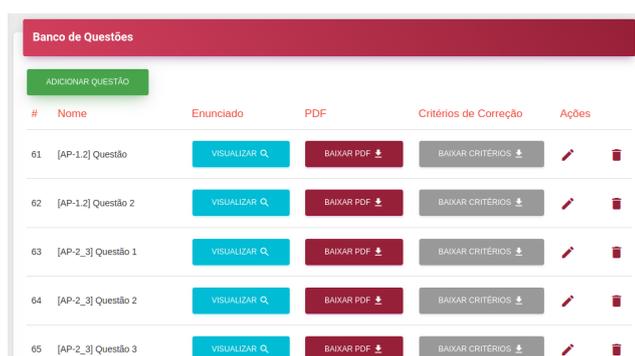
3.2.2.1 Banco de questões

No banco de questões são listadas as questões disponíveis para serem usadas no planejamento de tarefas, bem como criar novas questões e também editar e excluir as existentes. Na Figura 3.12 é mostrada a versão inicial do banco de questões, e na Figura 3.13 é mostrada a versão final do banco de questões, com as novas funcionalidades implementadas durante este estudo. Embora a interface da versão preliminar contivesse um botão de edição para cada questão, esta funcionalidade não estava implementada, ou seja, a versão preliminar permitia apenas inserir uma questão e visualizar uma listagem de questões existentes.



#	Nome	Enunciado	PDF	Critérios de Correção	Ações
61	[AP-1.2] Questão 1	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	EDITAR
62	[AP-1.2] Questão 2	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	EDITAR
63	[AP-2.3] Questão 1	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	EDITAR
64	[AP-2.3] Questão 2	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	EDITAR
65	[AP-2.3] Questão 3	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	EDITAR

Figura 3.12 – Versão preliminar da tela do banco de questões



#	Nome	Enunciado	PDF	Critérios de Correção	Ações
61	[AP-1.2] Questão	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	✎ 🗑️ ➕
62	[AP-1.2] Questão 2	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	✎ 🗑️ ➕
63	[AP-2.3] Questão 1	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	✎ 🗑️ ➕
64	[AP-2.3] Questão 2	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	✎ 🗑️ ➕
65	[AP-2.3] Questão 3	VISUALIZAR Q	BAIXAR PDF	BAIXAR CRITÉRIOS	✎ 🗑️ ➕

Figura 3.13 – Versão final da tela do banco de questões

3.2.2.2 Banco de Tarefas

No banco de tarefas são listadas todas as tarefas disponíveis. Para usá-las, basta atribuir uma data de agendamento as quais o aluno deve entregar. Também é possível cadastrar novas tarefas, editar e excluir as existentes. Na Figura 3.14 é exibida a versão preliminar do banco de

#	Nome	Enunciado Texto	Enunciado PDF	Ações
19	[AP-1.2] Testando o Corretor Automático	VISUALIZAR Q	BAIXAR PDF	EDITAR ATRIBUIR DATAS
20	[AP-2.3] Variáveis, Expressões, Entrada e Saída	VISUALIZAR Q	BAIXAR PDF	EDITAR ATRIBUIR DATAS
21	[AP-4.1] Estrutura de Decisão (1)	VISUALIZAR Q	BAIXAR PDF	EDITAR ATRIBUIR DATAS
22	[AP-4.2] Estrutura de Decisão (2)	VISUALIZAR Q	BAIXAR PDF	EDITAR ATRIBUIR DATAS
23	[AP-4.3] Estrutura de Decisão (3)	VISUALIZAR Q	BAIXAR PDF	EDITAR ATRIBUIR DATAS

Figura 3.14 – Versão preliminar da tela do banco de questões

tarefas e na Figura 3.15 é mostrada a versão final do banco de tarefas, com as novas funcionalidades disponíveis após a implementação deste estudo. Embora a interface da versão preliminar contivesse botões de edição e atribuição de datas para cada tarefa, estas funcionalidades não estavam implementadas, ou seja, a versão preliminar permitia apenas inserir uma tarefa e visualizar uma listagem de tarefas existentes.

#	Nome	Enunciado Texto	Enunciado PDF	Ações
19	[AP-1.2] Testando o Corretor Automático	VISUALIZAR Q	BAIXAR PDF	📅 ✎ 🗑️
20	[AP-2.3] Variáveis, Expressões, Entrada e Saída	VISUALIZAR Q	BAIXAR PDF	📅 ✎ 🗑️
21	[AP-4.1] Estrutura de Decisão (1)	VISUALIZAR Q	BAIXAR PDF	📅 ✎ 🗑️
22	[AP-4.2] Estrutura de Decisão (2)	VISUALIZAR Q	BAIXAR PDF	📅 ✎ 🗑️

Figura 3.15 – Versão final da tela do banco de questões

3.2.2.3 Gestão de Alunos

Na gestão de alunos, o professor tem acesso a todas as turmas que estão cadastradas para ele no corretor. Nessa função, é possível gerar relatórios de notas por semestre e por atividade específica, também, nessa funcionalidade o professor pode visualizar as entregas realizadas pelo aluno, estas são novidades que foram implementadas neste estudo.

Para gerar o relatório de tarefas entregues primeiro deve selecionar o semestre desejado, o último semestre ativo já é selecionado por padrão, em seguida escolhe para qual tarefa deseja que seja feito o relatório ou todas as disponíveis, que já vem pré-selecionado. Ao clicar em exportar notas, o relatório contendo as notas de todas as turmas por aluno, no formato *csv* (valores separados por vírgulas), é baixado. O resultado dessa implementação pode ser observado na Figura 3.16.

Para acessar as entregas realizadas por um aluno específico, basta clicar no botão de ação visualizar as entregas do aluno, ao clicar é redirecionado para a tela onde são listadas todas as

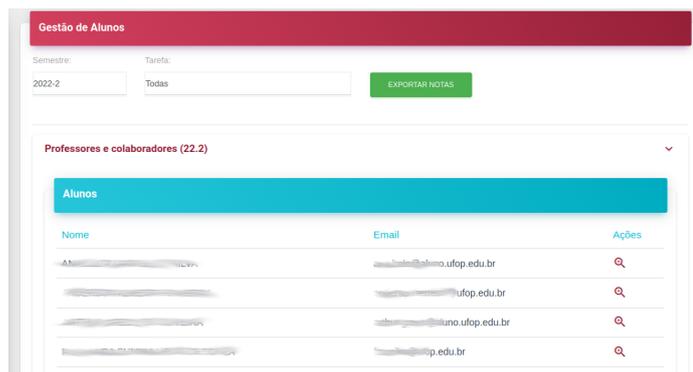


Figura 3.16 – tela de gestão de alunos e exportação de notas

entregas feitas pelo aluno, sendo possível visualizar a nota e baixar o código entregue por ele. Na Figura 3.17 é exibido o resultado da implementação desta funcionalidade.

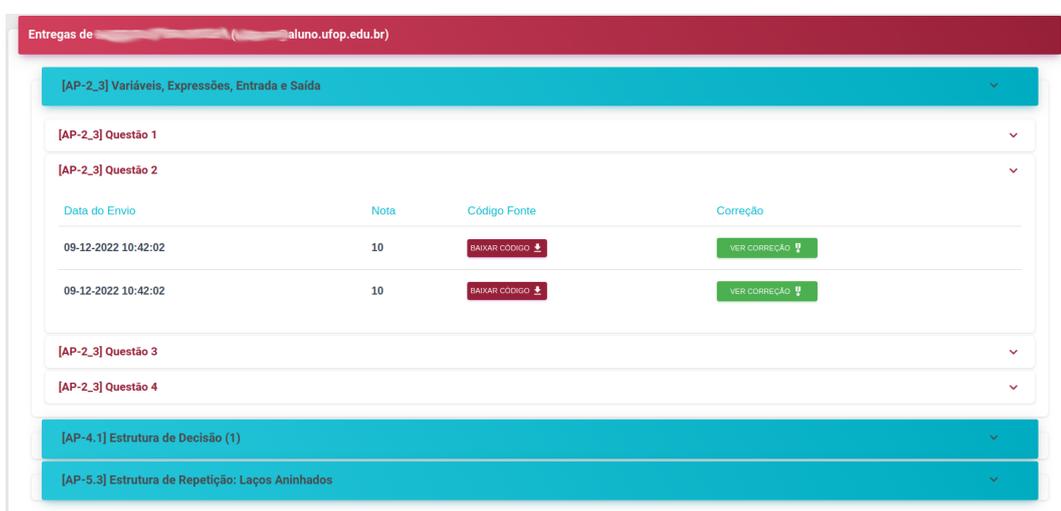


Figura 3.17 – tela com a listagem das entregas feitas pelo aluno

3.2.3 Funções disponíveis para o aluno

Para o aluno está disponível uma página inicial, onde ele pode visualizar as atividades abertas, entregues e perdidas. Esta é a primeira página acessada após o login. Como mostrado na Figura 3.18, o aluno pode visualizar de forma intuitiva o número de tarefas pendentes, finalizadas e perdidas. Ao clicar em uma das tarefas listadas é direcionado para a respectiva página que contém seus detalhes, como, a descrição da tarefa e cada questão que a compõe, a Figura 3.19 mostra o aspecto dessa página. Nela é possível acessar uma atividade específica para que seja possível submeter as soluções implementadas, visualizar o resultado da correção e baixar o código submetido anteriormente. Na Figura 3.21, é exibida a interface de entrega das tarefas e, na Figura 3.22, a de visualização do resultado de correção.

A página de entrega de questão recebeu uma importante atualização. Quando permitido é possível selecionar qual a linguagem de programação foi usada para resolver aquela questão,

Professores e colaboradores (22.1)



Tarefas Abertas
0



Tarefas Finalizadas
10



Tarefas Perdidas
0

Tarefas Abertas
Você possui 0 tarefas para entregar

Tarefa	Data Limite	Nota

Tarefas Finalizadas
Você concluiu 10 tarefas

Tarefa	Data Limite	Nota
[AP-5] Implementação de Funções	28-08-2022 23:59:59	10.00

Figura 3.18 – Página minhas tarefas

[AP-2_3] Variáveis, Expressões, Entrada e Saída

Descrição da Tarefa

Pré-requisitos:

- Estar familiarizado com o conteúdo das aulas anteriores e ter realizado as tarefas correspondentes.
- Ler os capítulos 2 e 3 do livro texto da disciplina.
- Assistir às aulas teóricas correspondentes.
- Aplicativo Thonny instalado.

Resumo:

- Variável é um mecanismo de armazenamento de dados na memória.
- A atribuição de valores é feita com o comando:

```
[nome_da_variavel] = valor
```

OBS.: Não copie e cole o código, digite no Thonny. Copiar/colar pode trazer erros por conta da codificação dos caracteres do texto.

Questão 1

Questão 2

Figura 3.19 – Página de detalhes da tarefa

assim, o corretor correto é selecionado para a correção. Na Figura 3.20 é mostrado a versão anterior dessa tela, e na Figura 3.21 é mostrado a versão atual.

Enviar nova solução (código fonte):

Nenhum arquivo escolhido

ENVIAR ARQUIVO ↕

Entregas

Data do Envio	Nota	Código Fonte	Correção
13-09-2022 15:23:21	-	BAIXAR CÓDIGO ↕	Pendente
16-08-2022 16:55:09	10	BAIXAR CÓDIGO ↕	VER CORREÇÃO ↕

Figura 3.20 – Versão preliminar da página de entrega de questão

A página perfil, exibida na Figura 3.23, traz informações básicas do usuário e também possibilita realizar a troca de senha, ela está disponível para todos os usuários do sistema.

Data do Envio	Nota	Código Fonte	Correção
25-11-2022 17:08:38	10	BAIXAR CÓDIGO	VER CORREÇÃO

Figura 3.21 – Versão final da página de entrega de questão

Questão 1 -- Correção da entrega: 16-08-2022 16:55:09

Execuções do programa 0.0

Nota final: 0.0

Figura 3.22 – Página de visualização correção e nota

Dados pessoais:

Nome

Email

CPF

Alterar senha:

Senha atual:

Nova senha:

Repetir nova senha:

ALTERAR SENHA

Figura 3.23 – Página perfil do usuário

Como parte dos objetivos desse estudo, foram realizadas correções pontuais para o aprimoramento da interface do corretor automático *opCoders Judge*, dentre eles estão a visualização de itens em negrito na formatação dos enunciados que estavam sendo ignorados pela página, a extensão do arquivo que sempre continha um *underline* no final quando se baixava o código fonte anteriormente submetido e também travamentos que ocorriam ao expirar a sessão o que obrigava fechar completamente o navegador. Essas correções contribuíram para uma melhor experiência de uso do corretor online.

3.3 Enquadramento da abordagem MVC no projeto

Na seção anterior teve o foco de mostrar a interface com o usuário, todas tendo seu código na camada de visualização (*View*). Mas ainda existem as camadas de modelo (*Model*) e de controle (*Control*) que completam o padrão MVC, a seguir vamos detalhar os componentes dessa camada.

3.3.1 Camada controle

A camada controle é responsável em receber as requisições e encaminha-las para as ações, sejam elas leitura ou escrita da camada de modelo ou uma simplesmente carregar uma visão. No *opCoders Judge* a camada controle possui três classes: (i) *IndexController* responsável em carregar a página inicial quando o usuário ainda não está autenticado, (ii) *AuthController* responsável em realizar as funções relacionadas a autenticação, e (iii) *AppController* que gerencia todas as demais funções do corretor.

3.3.2 Camada modelo

A camada modelo é responsável pela gerência de dados e as lógicas de negócio da aplicação. No *opCoders Judge* essa camada possui 7 classes que são identificadas de forma intuitiva de acordo com a funcionalidade que elas atendem, sendo *BancoDeQuestoes* para o gerenciamento das questões, *BancoDeTarefas* e *Tarefas* para o gerenciamento das tarefas, *Entrega* para o gerenciamento das entregas realizadas, *Correção* para o gerenciamento das correções, *GestaoDeAlunos* para o gerenciamento dos alunos e turmas e, finalmente, *Correção* com as funções responsáveis para o gerenciamento das estratégias de correção. Todas essas classes possuem funções para acesso de criação, alteração e exclusão de dados, e, por esse motivo é a única camada que tem acesso direto ao banco de dados.

A Figura 3.24 contém o diagrama em alto nível mostrando como as classes que compõem este projeto estão agrupadas em suas respectivas camadas.

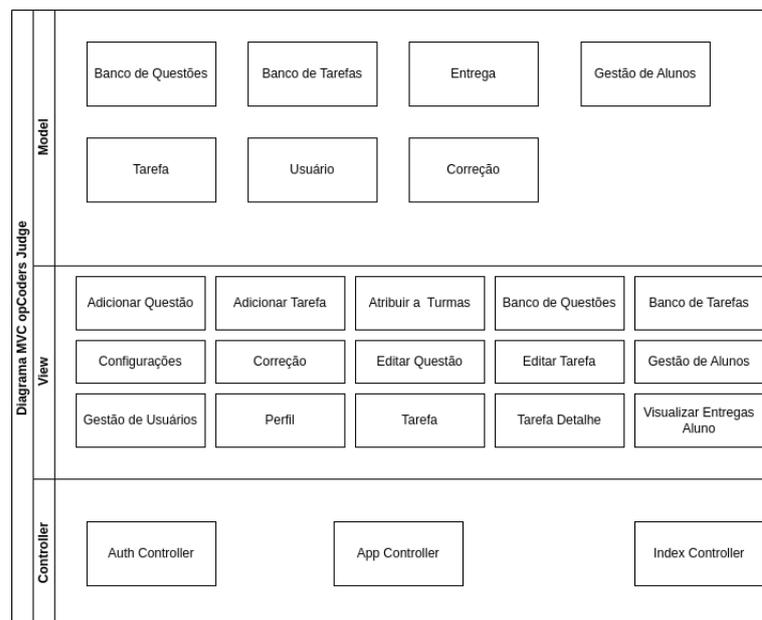


Figura 3.24 – Diagrama do enquadramento das classes do projeto no padrão MVC

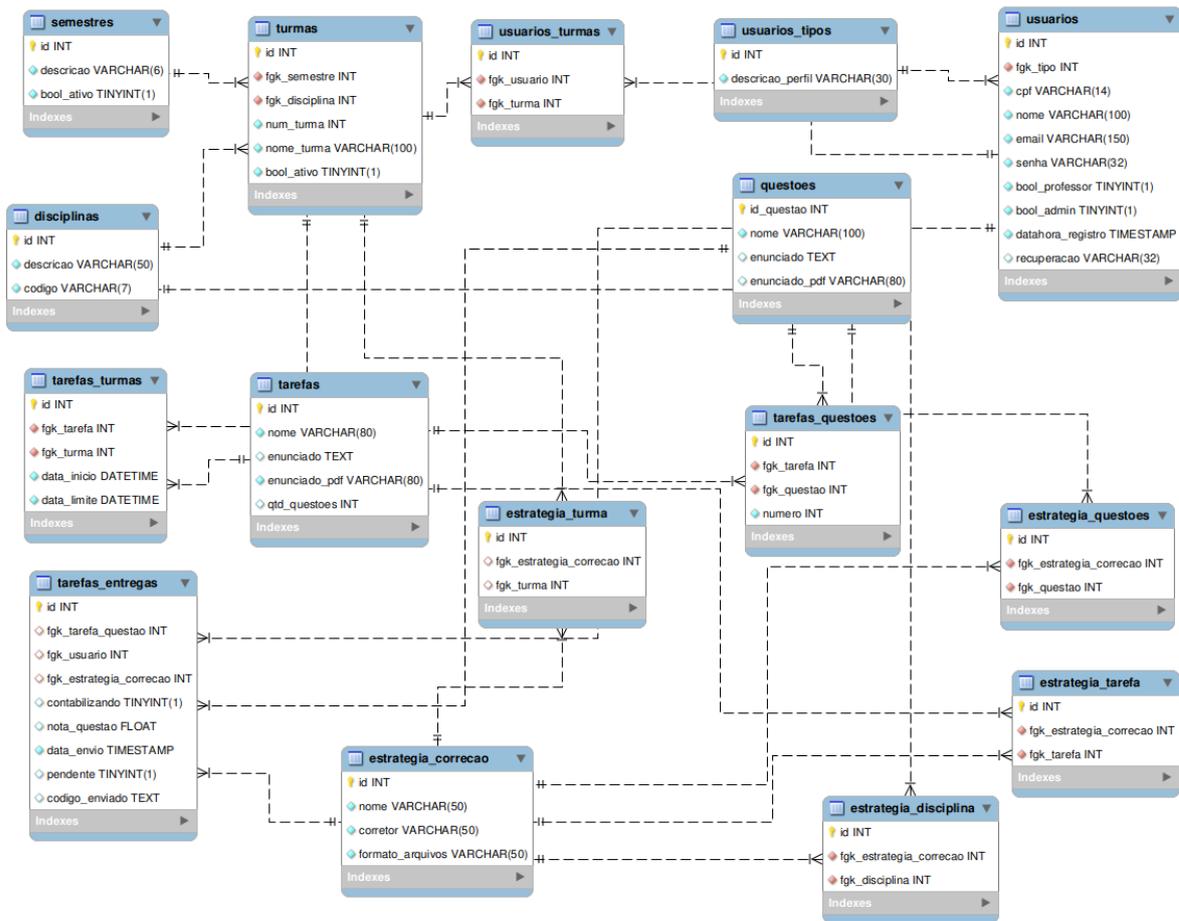


Figura 3.26 – Versão final do diagrama ER - Entidade relacionamento do banco de dados

4 Considerações Finais

Este estudo teve como objetivo fornecer uma interface web para o corretor de código automático do *opCoders Judge*. Além do refinamento e criação de funcionalidades para a plataforma, tal que os professores possam, de forma prática, gerenciar questões e tarefas, bem como visualizar os resultados das entregas de atividade de programação realizadas pelos alunos. Para os alunos, o objetivo principal era prover uma interface web que os permitisse se programar para a realização e entregas de atividades, bem como permiti-los receber a avaliação das tarefas de forma mais imediata. Além disto, é esperado que o uso da ferramenta auxilie no engajamento nas disciplinas de programação que, conseqüentemente, gerará uma melhora das notas e dos índices de aprovação.

Nas subseções a seguir, apresenta-se a conclusão e sugestões de trabalhos futuros.

4.1 Conclusão

Grande parte deste estudo ocorreu de forma prática, em torno de refinamentos do código previamente existente e da implementação das novas funcionalidades demandadas. O módulo *web* do *opCoders Judge* recebeu ajustes, correções e novas funcionalidades, de forma que os objetivos pretendidos foram alcançados, fornecendo uma interface web aprimorada para os professores e alunos.

O corretor de código automático do *opCoders Judge* foi inicialmente projetado para funcionar apenas corrigindo códigos fonte na linguagem *Python* atendendo às disciplinas de programação de computadores I (BCC701), após este estudo, a interface do corretor está preparada para suportar novas linguagens de programação, possibilitando que o corretor atenda outras disciplinas onde envolvam a prática de programação de computadores. Para tal, é necessário que exista o *backend* da linguagem hospedado no servidor. Um outro ponto importante para destacar, é a conclusão do módulo administrativo, que agora possui todas as funcionalidades necessárias para o gerenciamento do *opCoders Judge* por meio de sua própria interface.

4.2 Trabalhos Futuros

Embora a versão atual do corretor automático *opCoders Judge* tenha alcançado maturidade satisfatória através das etapas concluídas neste estudo, cumprindo os objetivos propostos inicialmente, diversos aprimoramentos podem ser realizados. Por exemplo, integração da autenticação dos usuários com a base do portal Minha UFOP, integração do editor de código já na plataforma, permitindo a edição do código fonte diretamente na interface ao invés do *upload* de

arquivo. Na edição de questões e tarefas o uso de um componente que permita que a edição do código HTML (Linguagem de Marcação de HiperTexto) de forma visual, e também a criação de tarefas com uso de questões aleatórias, baseado em critérios como tópicos e grau de dificuldade.

Referências

- ALVES, F. P.; JAQUES, P. Um ambiente virtual com feedback personalizado para apoio a disciplinas de programação. In: *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*. [S.l.: s.n.], 2014. v. 3, n. 1, p. 51.
- BENTO, E. J. *Desenvolvimento web com PHP e MySQL*. [S.l.]: Editora Casa do Código, 2021.
- BOOTSTRAP. *Get Bootstrap*. 2022. Url <https://getbootstrap.com/>.
- BRITO, P. S. S. Monografia ii. *UFOP*, v. 1, p. 51, 2019.
- GAMMA, E. *Padrões de projetos: soluções reutilizáveis*. [S.l.]: Bookman editora, 2009.
- GIL, A. C. et al. *Como elaborar projetos de pesquisa*. [S.l.]: Atlas São Paulo, 2002. v. 4.
- JÚNIOR, H. B. de F.; PEREIRA, F. D.; OLIVEIRA, E. H. T. de; OLIVEIRA, D. B. F. de; CARVALHO, L. S. G. de. Recomendação automática de problemas em juízes online usando processamento de linguagem natural e análise dirigida aos dados. In: SBC. *Anais do XXXI simpósio brasileiro de informática na educação*. [S.l.], 2020. p. 1152–1161.
- NAZÁRIO, D. C.; SOUZA, A. de. Boca-lab: Corretor automático de código adaptado ao ensino de linguagem de programação. *Anais do Computer on the Beach*, p. 42–46, 2010.
- PALHARES, R. D. A.; SANTOS, D. D. D. S. D.; VASCONCELOS, N. V. C. D. Uso de ferramentas computacionais para o auxílio do ensino: O estudo de caso da engenharia de produção. 2017.
- PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de software-9*. [S.l.]: McGraw Hill Brasil, 2021.
- SANTOS, J. C.; RIBEIRO, A. R. Jonline: proposta preliminar de um juiz online didático para o ensino de programação. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2012. v. 1, n. 1.
- SUTHERLAND, J.; SCHWABER, K. "Guia do Scrum". 2013. Disponível em: <<https://scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>>. Acesso em: 14 de março 2023.