



Universidade Federal de Pernambuco
Centro de Informática

Graduação em Ciência da Computação

**Uma ferramenta para a elaboração de
feedbacks de atividades iniciais de
programação integrada a um árbitro
virtual**

Bruna Alves Wanderley de Siqueira

Trabalho de Graduação

Recife
9 de janeiro de 2024

Universidade Federal de Pernambuco
Centro de Informática

Bruna Alves Wanderley de Siqueira

**Uma ferramenta para a elaboração de feedbacks de
atividades iniciais de programação integrada a um árbitro
virtual**

Trabalho apresentado ao Programa de Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: *Ricardo Massa Ferreira Lima*

Recife
9 de janeiro de 2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Siqueira, Bruna Alves Wanderley de.

Uma ferramenta para a elaboração de feedbacks de atividades iniciais de programação integrada a um árbitro virtual / Bruna Alves Wanderley de Siqueira. - Recife, 2024.

41 p. : il.

Orientador(a): Ricardo Massa Ferreira Lima

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Ciências da Computação - Bacharelado, 2024.

1. Ensino de Programação. 2. Feedbacks. 3. Monitoria. 4. Árbitros Virtuais. I. Lima, Ricardo Massa Ferreira. (Orientação). II. Título.

000 CDD (22.ed.)

*À monitoria de Introdução à Programação, onde me
encontrei.
Em memória do meu tio avô, Gui.*

Agradecimentos

Primeiramente, agradeço a meus pais, Maria do Socorro e Luis Roberto, minhas avós Maurineth e Roberta, e todos os meus familiares que me deram amor, carinho, valores e me incentivaram a seguir meus objetivos, me ajudando de todas as formas possíveis. Muitas vezes continuei a jornada muito mais por vocês do que por mim mesma.

Agradeço também aos professores da disciplina de Introdução à Programação, Sérgio Soares, Filipe Calegário e Ricardo Massa, ao qual agradeço especialmente pela orientação neste trabalho e em outros momentos da graduação. Obrigada pelos conhecimentos e pela oportunidade de fazer parte da equipe de monitoria, onde certamente me encontrei. Agradeço a todos os meus companheiros de monitoria, pelas trocas, experiências vividas juntos e pelas contribuições para este trabalho. Meu muito obrigada aos meus monitores de toda a graduação, certamente não seria possível chegar aqui sem vocês. Um agradecimento especial a Daniel Bastos, criador do Dikastis, que foi meu monitor no primeiro período e me deu todo o suporte para desenvolver esse trabalho.

Minha gratidão a todos os meus colegas de curso, por dividirmos aprendizados, preocupações e boas risadas. Vocês tornaram a trajetória mais leve, principalmente quando estávamos isolados em casa durante a pandemia. Agradeço ao meu namorado, Matheus, pelo apoio incondicional e por dividir a graduação e a vida comigo. Vocês foram os melhores presentes da faculdade.

Agradeço profundamente a todos os meus professores, dos que me ensinaram quando criança até os que participaram de alguma forma da minha experiência no curso de Ciência da Computação, sou grata por todos os ensinamentos e conselhos que diploma nenhum pode atestar. Registro minha admiração a vocês e à docência.

Ensinar é um exercício de imortalidade. De alguma forma continuamos a viver naqueles cujos olhos aprenderam a ver o mundo pela magia da nossa palavra.

—RUBEM ALVES (A Alegria de Ensinar, 1994)

Resumo

As disciplinas introdutórias de programação são essenciais na formação de profissionais da computação. Nos cursos de Ciência de Computação e Sistemas de Informação da Universidade Federal de Pernambuco, esses conhecimentos são consolidados nas disciplinas de Introdução à Programação e Programação 1, que utilizam o árbitro virtual Dikastis no processo de ensino-aprendizagem. Além disso, as disciplinas contam com uma equipe de cerca de 50 monitores que desempenham um papel essencial no aprendizado dos alunos, fornecendo feedback técnico e dando conselhos a cada atividade avaliativa. Esse trabalho propõe o desenvolvimento de uma ferramenta integrada ao Dikastis para facilitar o processo de elaboração dos feedbacks, garantindo mais qualidade e fortalecendo o vínculo entre monitores e alunos.

Palavras-chave: Árbitros Virtuais, Ensino de Programação, Feedback, Monitoria.

Abstract

The introductory programming courses are essential in the education of computer science professionals. In the Computer Science and Information Systems programs at UFPE, these concepts are reinforced in the courses of Introduction to Programming and Programming 1, which utilize the virtual judge Dikastis in the teaching-learning process. Additionally, these courses have a team of approximately 50 teaching assistants who play a crucial role in students' learning by providing technical feedback and advice on each evaluative activity. This project proposes the development of a tool integrated with Dikastis to streamline the feedback generation process, ensuring higher quality and strengthening the bond between teaching assistants and students.

Keywords: Feedback, Programming Teaching, Teaching Assistant, Virtual Judge

Sumário

1	Introdução	1
2	Trabalhos Relacionados	3
2.1	Árbitros Virtuais e Juízes Online	3
2.2	Aprendizado de Programação	4
2.3	Monitoria	5
3	Metodologia	7
3.1	Análise do problema	7
3.2	Definição de Requisitos	8
3.3	Proposta de Solução	9
3.3.1	Descrição da Aplicação	9
3.3.2	Descrição da Experiência	10
3.3.2.1	Professor/Monitor	11
3.3.2.2	Aluno	13
3.4	Validação	15
4	Resultados	17
4.1	Análise do processo de envio de Feedbacks	17
4.1.1	Pontos positivos	17
4.1.2	Desafios do processo	19
4.1.3	Sugestões de melhoria	21
4.2	Validação da solução proposta	21
5	Conclusão	23

Lista de Figuras

3.1	Arquitetura do Dikastis	9
3.2	Representação das entidades do módulo de Feedbacks	10
3.3	Tela com listagem de alunos para monitoramento do envio de Feedbacks	11
3.4	Tela de escrita de Feedbacks	12
3.5	Tela de resumo do Feedback concluído	12
3.6	Tela de problemas de uma atividade ativa com aba de feedbacks	13
3.7	Tela de resumo de feedback com possibilidade de resposta	14
3.8	Tela de resumo de feedback recebido com resposta finalizada	14

CAPÍTULO 1

Introdução

As disciplinas introdutórias de programação estão presentes em todos os cursos de computação e pode-se dizer que são essenciais para o desenvolvimento de habilidades exigidas para se tornar um profissional da área. Dentro do curso de Ciência da Computação da Universidade Federal de Pernambuco, a disciplina de Introdução à Programação exerce esse papel no primeiro período do curso, tendo em sua ementa fundamentos da lógica de programação e suas aplicações com a linguagem *Python*. Analogamente, o curso de Sistemas de Informação possui a disciplina Programação 1, que possui carga horária menor, mas é lecionada pela mesma equipe docente e possui mesma ementa que Introdução à Programação.

Para dar suporte ao ensino presencial, é utilizado um ambiente virtual de aprendizagem específico para o ensino e prática de programação chamado de juiz online ou árbitro virtual. A estrutura dessas ferramentas se baseia na correção automática de programas submetidos como resolução de um problema a partir da comparação da saída esperada com a saída fornecida pelo programa do aluno, mediante uma mesma entrada. No Dikastis, árbitro virtual usado nas disciplinas supracitadas do Centro de Informática e desenvolvido por alunos do centro, são disponibilizados os problemas a serem resolvidos pelos alunos classificados por assunto. Além do ambiente do aluno, em que ele consegue ver suas atividades e escrever os programas para cada atividade, o Dikastis também possui um ambiente para o professor ou monitor, no qual é possível acompanhar o desempenho dos estudantes em uma determinada atividades, verificar os códigos que ele já enviou e criar novas atividades para a turma.

Além das aulas presenciais e do apoio do juiz online, as disciplinas de Introdução à Programação e Programação 1 contam com uma equipe de cerca de 50 monitores, um diferencial no processo de ensino-aprendizagem. A monitoria é responsável por elaborar os problemas que irão compor as atividades avaliativas dos alunos, pela revisão desses problemas, pela avaliação dos códigos submetidos pelos alunos a fim de enviar um feedback personalizado sobre o desempenho de cada um e por dar suporte presencial e online, tirando dúvidas dos alunos, todas essas atividades são supervisionadas pela equipe docente. A interação entre alunos e monitores cria um vínculo de confiança que é fundamental para o processo de aprendizagem (Cunha Júnior, 2017) e as trocas de conhecimento são oportunidades para os monitores aprenderem e fixarem os conteúdos que estão ensinando (Gouveia, 2018). Apesar da importância citada, existem desafios na coordenação das atividades da monitoria por conta da quantidade de monitores envolvidos e da logística de acompanhar a qualidade das atividades por eles realizadas, como apontado por professores e monitores da disciplina durante uma pesquisa qualitativa.

Diante disso, as disciplinas introdutórias mencionadas se utilizam dos monitores para fornecer feedbacks individuais aos alunos sobre seus códigos, indo além do que pode ser apontado automaticamente pelo Dikastis, dando conselhos e sugestões de melhoria. Para além de um fe-

edback técnico, esse momento também cria um vínculo entre monitor e aluno, desenvolvendo um canal confiável para pedir ajuda, o que favorece o aprendizado e facilita as interações presenciais (Cunha Júnior, 2017). Entende-se, portanto, que o conhecimento de um bom programador não está apenas em conseguir resolver os problemas especificados, mas também escrever códigos com qualidade, organização, legibilidade e simplicidade, reconhecendo que essas habilidades são fundamentais para a construção de softwares no mercado e o trabalho em equipe (Robert C. Martin, 2009).

Reconhecendo a importância dos feedbacks para o processo de ensino-aprendizagem, vê-se a oportunidade de analisar o processo de elaboração e envio destes a fim de encontrar pontos de melhoria e conseguir propor uma ferramenta integrada ao árbitro virtual já utilizado, na intenção de facilitar esse processo, garantindo mais qualidade e oportunizando o fortalecimento do vínculo entre monitor e aluno.

Trabalhos Relacionados

Os trabalhos que podem ser relacionados com o objetivo deste trabalho foram categorizados pelas áreas que a temática em questão envolve, apresentando um pouco do estado da arte em cada um dos contextos.

2.1 Árbitros Virtuais e Juízes Online

Ramadhan e Shibab[7] definem diagnosticadores automáticos de programas como sistemas capazes de identificar problemas em programas e ajudar no aprendizado de programação para iniciantes, tendo o papel de ajudar a encontrar problemas semânticos, ou seja, se a lógica do programa está de acordo com o esperado, e erros de execução, sintaxe e compilação, provendo feedback automático para o usuário, assim como é feito no Dikastis, árbitro virtual usado nas disciplinas supracitadas do Centro de Informática. Nessa definição, considera-se feedback automático como qualquer retorno da plataforma ao usuário acerca da corretude, desempenho e execução do código. É possível classificar os diagnosticadores de programas de acordo com o momento em que a análise do código acontece: diagnosticadores *post-event* são aqueles em que a análise ocorre depois que o código está pronto e diagnosticadores *in-event* são aqueles em que a análise ocorre durante a criação do código. Podemos afirmar que o Dikastis é um diagnosticador *post-event*, pois a análise dos códigos só acontece após o aluno sinalizar que deseja fazer uma submissão ao árbitro virtual. O trabalho também comparada diferentes estratégias de análises de programas, incluindo análise semântica, que direciona o usuário e sinaliza caso ele não esteja utilizando a lógica correta para resolver o que foi pedido. O Dikastis, por sua vez, não possui essa funcionalidade, tendo as capacidades de: avaliar as saídas esperadas do programa e sinalizar caso estejam ou não de acordo com o esperado, entregando um feedback automático sobre a corretude do *output*, executar o programa e sinalizar problemas de compilação ou erros de execução, entregando outro tipo de feedback sobre a execução do programa, e avaliar o tempo de execução, sinalizando com um outro feedback caso o tempo tenha excedido o limite previsto na atividade, como forma de alertar sobre aspectos de otimização.

Na revisão da literatura de Francisco *et. al.*[3], são debatidos vários aspectos em relação à utilização de juízes online nas disciplinas iniciais do curso de Ciência da Computação, a fim de conseguir definir qual a especificação mais adequada para um juiz online. Entre as vantagens relatadas nos trabalhos selecionados, estão: oferecer um ambiente que permite ao aluno regular sua aprendizagem, aumentando sua motivação, reduz a carga de trabalho do professor, que tem à sua disposição um conjunto de problemas já elaborados e testados, a possibilidade de automatizar a detecção de plágio e a emissão de relatórios. Entre as vantagens citadas, o

Dikastis ainda não consegue formalmente oferecer a última, a emissão de relatórios, mas é possível acompanhar o desempenho dos estudantes na página da turma. Além disso, são listados alguns requisitos funcionais dos juízes online apontados pelos trabalhos, o principal foi o feedback automático, resultado da execução do programa do aluno aos testes definidos previamente no árbitro virtual, funcionalidade também contida no Dikastis. Em relação às desvantagens apontadas, as principais foram: falta de validação e maturidade dos sistemas, dificuldade de instalação e manipulação dos programas, usabilidade, visibilidade das notas e ocorrências de plágio. Apesar do Dikastis ser um sistema recente, ele foi validado dentro do ambiente em que é utilizado e diversas melhorias foram feitas a partir do feedback dos usuários, além de se ter a possibilidade de fazer mudanças no sistema sempre que surge uma nova demanda ou dificuldade. Em relação a casos de plágio, observou-se que há uma facilidade maior de encontrar os casos já que todos os códigos são submetidos no sistema, apesar de não se ter uma forma automatizada de verificar, é uma ferramenta central e que dá diversos indícios de comportamento fora do padrão, permitindo identificar cópias. O último aspecto analisado são os requisitos não-funcionais que árbitros virtuais deveriam ter, foram eles: integração, usabilidade, escalabilidade, disponibilidade e segurança. O Dikastis está hospedado atualmente no servidor do Centro de Informática, dependendo diretamente do funcionamento dos sistemas do centro e seguindo os padrões de segurança exigidos por esse ambiente. Em relação aos outros requisitos, integração e escalabilidade não são críticos no contexto pois não temos uma demanda muito alta, normalmente cada semestre tem cerca de 100 alunos, e não integramos a ferramenta com nenhum outro ambiente virtual de aprendizado.

Este trabalho busca ir além das funcionalidades discutidas no estado da arte para árbitros virtuais, propondo uma funcionalidade que consiga facilitar o processo de feedback manual, oferecendo suporte para uma prática fundamental no processo de aprendizagem proposto nas disciplinas do contexto analisado que complementa o que o Dikastis é capaz de oferecer enquanto ferramenta de auxílio de ensino de programação.

2.2 Aprendizado de Programação

A área de metodologias de ensino de programação possui diversos trabalhos acerca das estratégias que podem ser utilizadas para garantir o aprendizado dos conceitos essenciais para a formação de profissionais de tecnologia. A metodologia de ensino utilizada nas disciplinas de Introdução à Programação e Programação 1 dos cursos do Centro de Informática tem como base a resolução de problemas como principal forma de aprender os conceitos apresentados em sala. Looi e Seyal[5] analisaram o impacto do *Problem-Based Learning* (PBL) no ensino de disciplinas introdutórias de programação, definindo essa metodologia como uma estratégia de ensino centrada no aluno em que o objetivo é que o conhecimento seja aprendido ativamente por meio da resolução de problemas, de forma que o estudante desenvolva a capacidade de construir soluções através do conhecimento adquirido em vez de receber soluções prontas (Looi e Seyal, 2014 apud Hmelo e Ferrari 1997; Delisle 1997). Como resultado da pesquisa realizada com 43 estudantes, comparando o impacto da utilização do PBL no desenvolvimento de habilidades de comunicação e resolução de problemas, no processo de aprendizado e no engajamento e auto-motivação, foi verificado que os estudantes melhoraram seu desempenho

geral no curso, destacando a melhora na habilidade de resolver problemas e o desenvolvimento da autonomia no processo de aprendizado. É possível relacionar a metodologia descrita com a utilizada pelos docentes das disciplinas de Introdução à Programação e Programação 1, ressaltando a importância do desenvolvimento de habilidades que extrapolam o conhecimento técnico de uma linguagem de programação e a participação crucial dos monitores para que o processo de aprendizado envolva a construção de soluções por meio de trocas com outras pessoas, momentos que acontecem em sala de aula, durante as monitorias.

Faheem *et. al.*[1] discutem o papel de competências sociais para profissionais do desenvolvimento de software ressaltando a importância do trabalho em equipe, adaptabilidade, organização, habilidades interpessoais e comunicação para desenvolver bem as atribuições de um programador. Reconhecendo essa importância, as disciplinas introdutórias dos cursos de Ciência da Computação e Sistemas de Informação se utilizam dos feedbacks individuais para desenvolver nos monitores e nos alunos habilidades como as supracitadas, que vão além dos conteúdos técnicos programados. Os alunos recebem conselhos dos monitores, que precisam analisar os códigos enviados e dar sugestões sobre pontos de melhoria de forma empática e didática, praticando habilidades de comunicação e simulando situações que acontecem na vida profissional, em que se deve dar e receber sugestões, pondo em prática o que se achar adequado. As habilidades citadas, junto a conhecimentos técnicos, formam boas práticas de programação que moldam os critérios de qualidade de software e devem ser introduzidas desde o início da formação do profissional, como uso de nomes significativos, modularidade, simplicidade, organização de código, entre outros (Robert C. Martin, 2008).

2.3 Monitoria

O diferencial da experiência de aprendizado do contexto descrito neste trabalho certamente é o envolvimento da equipe de monitoria no processo de ensino-aprendizagem. Cunha Junior[2] explora a relação entre o conceito de Zona de Desenvolvimento Proximal de Lev Vygotsky[8] e as atividades de monitoria, ressaltando a importância da interação com outros alunos no processo de educação, sendo um momento de troca e aprendizado não só para os alunos mas também para os monitores, ao serem espelho para os alunos e se espelharem em seus professores da mesma forma, sendo possível observar o aprendizado por repetição de padrões. É importante ressaltar a identificação que há entre alunos e monitores, por se tratar de uma relação mais horizontal e de pessoas que estão na mesma realidade, já que os monitores também são alunos. Essa identificação abre espaço para uma relação de confiança e conforto, que permite que o aluno consiga falar sobre suas dificuldades, pedir ajuda e compartilhar problemas. O monitor, vendo no aluno a si mesmo, consegue ter uma percepção melhor sobre estratégias para aumentar o engajamento e interesse, além de conseguir ajudar o aluno com dificuldades pelas quais já passou. Todos esses fatores tornam a experiência do feedback individual e das monitorias presenciais cruciais para ter um ambiente de aprendizado seguro e mais próximo dos alunos.

O estudo de Gouveia [4], que analisa a experiência de monitoria na disciplina de Programação de Computadores, apontou como a experiência de monitoria é importante para a consolidação do conhecimento de programação para os ex-alunos das disciplinas iniciais, que

conseguem exercitar seus conhecimentos ao exercerem a função de monitores. Como supracitado, as atividades da monitoria são bastante diversas e englobam a prática de atividades que envolvem conhecimentos técnicos já estudados e o desenvolvimentos de outras habilidades também fundamentais para a formação de profissionais de tecnologia.

CAPÍTULO 3

Metodologia

3.1 Análise do problema

No contexto supracitado, o processo do envio de feedbacks aos alunos a cada atividade avaliativa realizada é feito manualmente por um monitor da disciplina. Ao final do prazo de realização de uma atividade, o monitor deve procurar em uma planilha disponibilizada pela equipe docente qual aluno ele deve enviar o feedback e então começar a analisar os códigos enviados por este aluno na atividade em questão. Após analisar pontos como organização, legibilidade do código, utilização dos conhecimentos estudados em sala, boas práticas e desempenho geral na atividade, o monitor deve enviar por *e-mail* um feedback comentando sobre o que foi observado de maneira informal e sinalizar que enviou o feedback, copiando o texto em uma página de registro da equipe de monitoria.

Para entender melhor a realidade contextualizada, detalhando as experiências e desafios de alunos, monitores e professores das disciplinas de Introdução à Programação e Programação 1, foram realizadas entrevistas e pesquisas qualitativas com ex-alunos, monitores, ex-monitores e professores das disciplinas. Foram entrevistados 2 monitores e ex-alunos e um professor, a pesquisa qualitativa foi aplicada a 16 monitores e ex-alunos. As perguntas realizadas aos monitores, ex-monitores e ex-alunos buscavam obter respostas em relação à experiência que eles vivenciaram como alunos ao receberem feedbacks e como monitores ao enviarem feedbacks, identificando pontos positivos e desafios nesse processo. As perguntas realizadas foram:

- Quando você foi aluno, achava os feedbacks que recebeu úteis?
- Se sim, pode citar algum exemplo de como os feedbacks contribuíram com seu aprendizado, desempenho ou engajamento na disciplinas?
- Se não, pode citar pontos de melhoria dos feedbacks que você recebeu ou explicar porque não foram úteis?
- Como monitor, você sentia/sente dificuldade para escrever bons feedbacks aos alunos?
- Se sim, pode citar alguma dificuldade?
- Quais habilidades a elaboração de feedbacks te ajudou a desenvolver?
- Tem algum outro fator no processo de envio dos feedbacks que você acha que pode melhorar? Se possível, dê exemplos

As perguntas realizadas na entrevista para o professor e os monitores que assume o papel de monitores-chefe foram diferentes, pois buscavam identificar pontos de melhoria no processo de gestão da equipe em relação ao envio e qualidade dos feedbacks.

- Quais os maiores desafios enfrentados atualmente no envio de feedbacks?
- Existe alguma ferramenta que é utilizada para acompanhar o envio, verificar pendências ou atrasos?
- Existe alguma forma de garantir que os alunos estão recebendo os feedbacks?
- Qual a importância do feedback na experiência de ensino que as disciplinas desejam propor?

A partir dos resultados coletados, foi possível identificar os problemas no processo de envio dos feedbacks, tanto na perspectiva dos monitores e alunos, quanto na dos professores e monitores-chefe, e consolidar os principais motivos pelos quais essa prática se torna tão importante no contexto com relatos reais dos envolvidos.

3.2 Definição de Requisitos

Com base nos resultados das entrevistas e da pesquisa, foram compilados os principais problemas encontrados no processo de envio de feedbacks e, a partir disso, foi possível definir requisitos mínimos que uma ferramenta voltada para a solução desses problemas deveria ter. São eles:

- Uma forma de escrever o feedback enquanto se analisa o código, tendo a possibilidade de fazer referências a trechos específicos ou mencionar alguma parte diretamente
- Um mecanismo de monitoramento do status do feedback em relação ao prazo de envio
- Sinalização de que o aluno leu o feedback
- Possibilidade de registrar uma resposta do aluno ao feedback
- Disponibilizar orientações acerca dos critérios que devem ser analisados nos feedbacks

Os requisitos foram detalhados tecnicamente para que pudessem se tornar uma funcionalidade completa que existisse dentro do Dikastis, visto que seria a estratégia mais viável e que facilitaria o processo justamente por conseguir atender a esses requisitos utilizando a ferramenta cotidiana da disciplina. Utilizando o Dikastis para a escrita dos feedbacks, os monitores não precisariam utilizar o *e-mail* para enviar os feedbacks e já estariam utilizando a plataforma que contém os códigos dos alunos, e os alunos, por sua vez, não precisariam abrir o email para ler o feedback, tudo ficaria concentrado em um único lugar. Além disso, os professores e monitores-chefe consegue ter controle e monitorar os feedbacks que já foram enviados pois isso também poderia ser visto no Dikastis, diferente da situação atual em que se depende da confirmação do aluno ou do monitor sobre o envio. Essas definições já atacam os problemas relacionados à dificuldades com o meio de envio, interação entre monitores e alunos, controle de qualidade e padronização dos feedbacks.

3.3 Proposta de Solução

3.3.1 Descrição da Aplicação

A definição detalhada da solução a nível de implementação foi baseada na arquitetura do Dikastis, pois por se tratar de um sistema em funcionamento e estável, entende-se que era mais prudente seguir o padrão já utilizado, de forma a facilitar o desenvolvimento e evitar problemas de integração. A plataforma está organizada em módulos que possuem objetivos particulares e se relacionam entre si, são eles: *Problems*, *Tasks*, *Teams* e *People*. Dado essa organização semântica e seguindo o padrão utilizado, foi criado um novo módulo de Feedbacks.

Atualmente, a arquitetura do Dikastis (Figura 3.1) está dividida em 3 partes: a aplicação *FrontEnd*, que será executada no navegador do usuário, uma *API* responsável por gerenciar os dados e o executor de códigos, que recebe os códigos para executar e realizar as verificações necessárias. O módulo proposto nesse trabalho envolve apenas a criação e registro de novas informações (textos do feedbacks) vinculando-as com entidades já existentes como Pessoas e Problemas, por isso, as modificações necessárias foram realizadas apenas na aplicação *frontend* para a criação de telas e na *API* para consulta e registro dos novos dados.

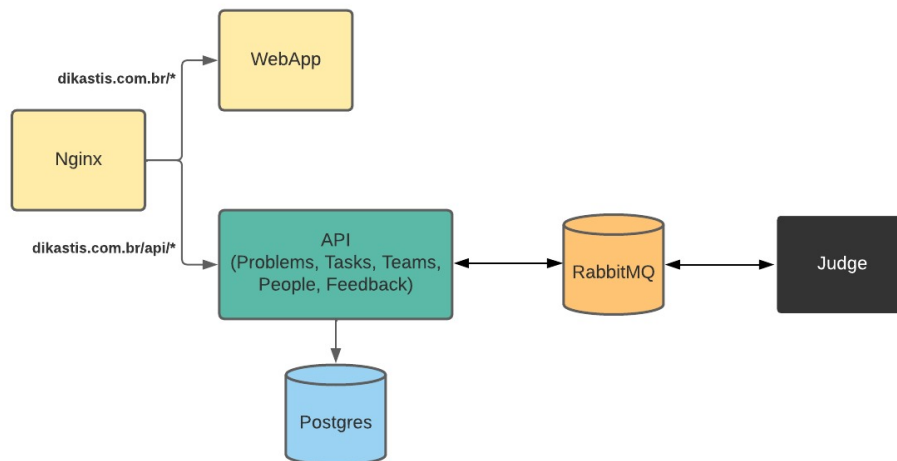


Figura 3.1 Arquitetura do Dikastis

A aplicação *frontend* foi desenvolvida na linguagem *Javascript* com o *framework React*, que se baseia fortemente na componentização dos elementos, o que foi bastante útil para a criação das novas telas para o módulo de feedbacks, já que os componentes já utilizados puderam ser aproveitados mantendo o padrão de tecnologia e design, evitando retrabalho e problemas de compatibilidade. Já a *API* foi implementada em *Java* com auxílio do *framework Micronaut* e outras bibliotecas de desenvolvimento *web*, seguindo o padrão de projeto já utilizado para desenvolvimento do novo módulo. O banco de dados utilizado pelo sistema é *Postgres*, um banco de dados relacional de código aberto. O novo módulo se utilizou da criação de novas tabelas para armazenar as novas entidades e relacionamentos, definidos como ilustrado na Figura 3.2.

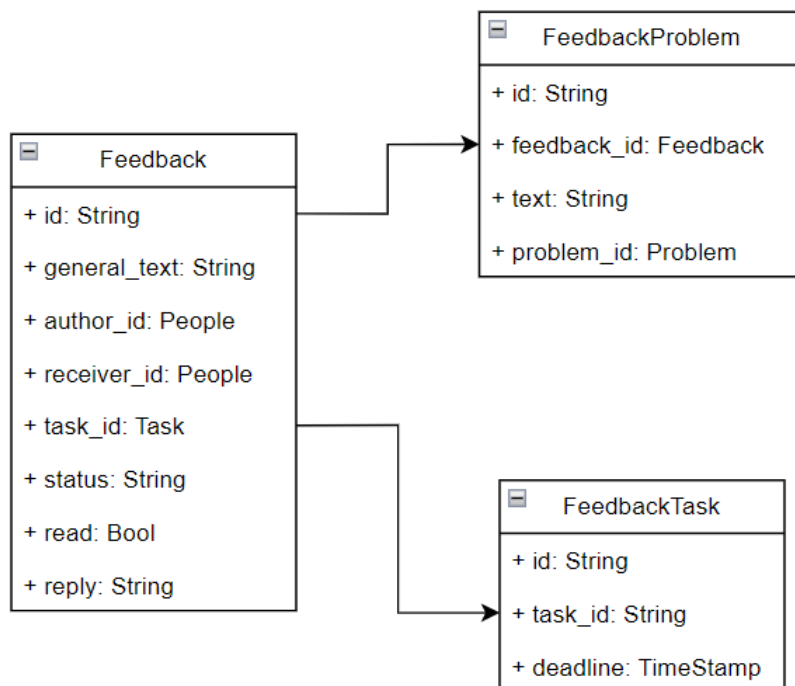


Figura 3.2 Representação das entidades do módulo de Feedbacks

A escolha desse relacionamento entre os dados se deu para garantir o maior aproveitamento possível dos módulos já existentes e assegurar a existência de um feedback por aluno em cada atividade, além de possibilitar a escrita do feedback individual por problema de forma isolada, permitindo a implementação de funcionalidades como o envio parcial e rascunho do feedback.

Toda estrutura já estava configurada em *containers Docker* para facilitar a integração de novos desenvolvedores ao projeto, assim como sua implantação no ambiente de produção. Dentro do sistema também é utilizado o *NGINX* como *proxy* para gerenciamento e validação das requisições HTTP, e *RabbitMQ* como *middleware* de mensageria para a comunicação entre o serviço executor de códigos e a *API*.

3.3.2 Descrição da Experiência

A experiência proposta para o usuários é bastante simples, já que o objetivo dessa nova abordagem é facilitar o processo de envio de feedbacks e tornar mais rico em termos de interação e aproveitamento do aprendizado. A experiência para o professor/monitor leva em consideração as funcionalidades de escrita, envio e monitoramento de feedbacks, enquanto para os alunos temos a leitura e resposta do feedback.

3.3.2.1 Professor/Monitor

A entrada para o módulo de feedbacks para um usuário do tipo professor ou monitor será pela tela de acompanhamento de uma atividade. Ao selecionar uma atividade específica de uma turma, ele consegue ver uma lista com todos os alunos e os status de seus feedbacks (Figura 3.3). Todos os usuários com permissões de professor ou monitor podem acessar todos os feedbacks, permitindo a leitura do feedback escrito por outro usuário, funcionalidade necessária para que haja a verificação de qualidade.

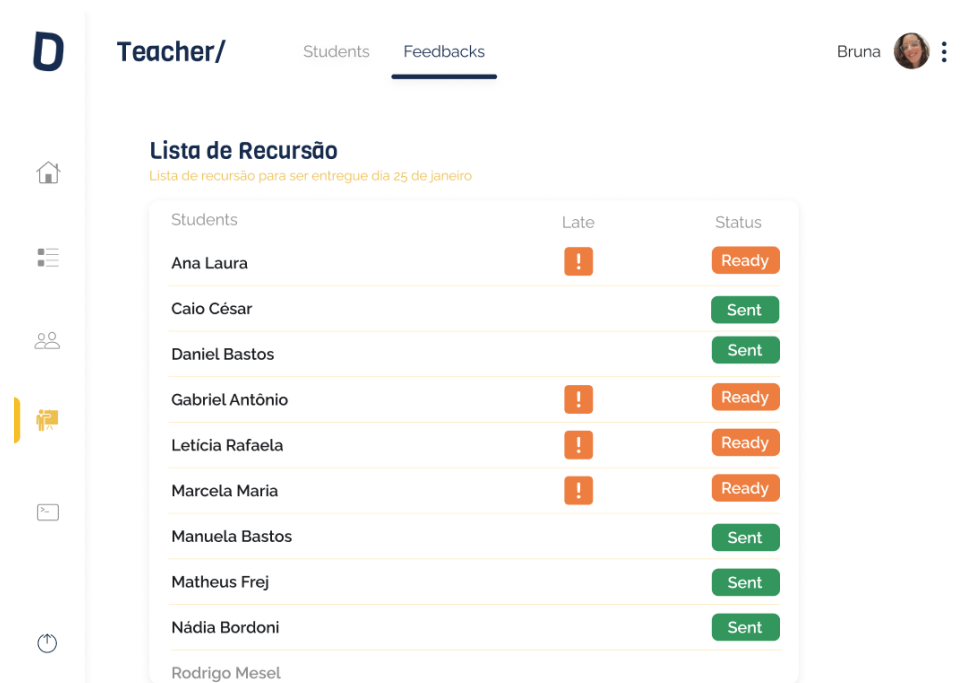


Figura 3.3 Tela com listagem de alunos para monitoramento do envio de Feedbacks

Ao escolher um aluno, o usuário tem acesso à tela de escrita de feedback (Figura 3.4), podendo escrever um feedback geral da atividade, analisar as submissões de cada problema e deixar um comentário específico por problema, tudo em uma mesma tela. É possível ainda salvar o feedback como rascunho para continuar a escrita depois, sem perder o que já foi feito, e clicar no botão com uma interrogação para consultar as orientações para a escrita de um feedback com qualidade. Essas funcionalidades atendem às demandas dos monitores em relação a consulta do código do aluno enquanto se escreve, facilitando a referência a trechos do código.

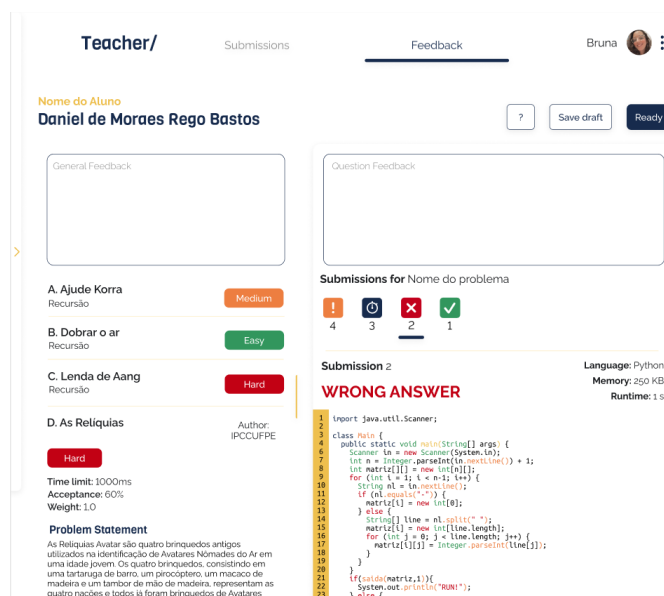


Figura 3.4 Tela de escrita de Feedbacks

Após o término do processo de escrita e confirmação do envio, a tela passa a mostrar o feedback sem possibilidade de edição, a resposta do aluno ao feedback (caso haja) e as submissões do aluno, sendo uma tela de resumo do feedback que já foi concluído (Figura 3.5). Isso possibilita que o monitor tenha um retorno do aluno, mesmo que simples, mas que ajuda a fortalecer a interação e oportuniza que o contato entre os envolvidos continue por outro meio.

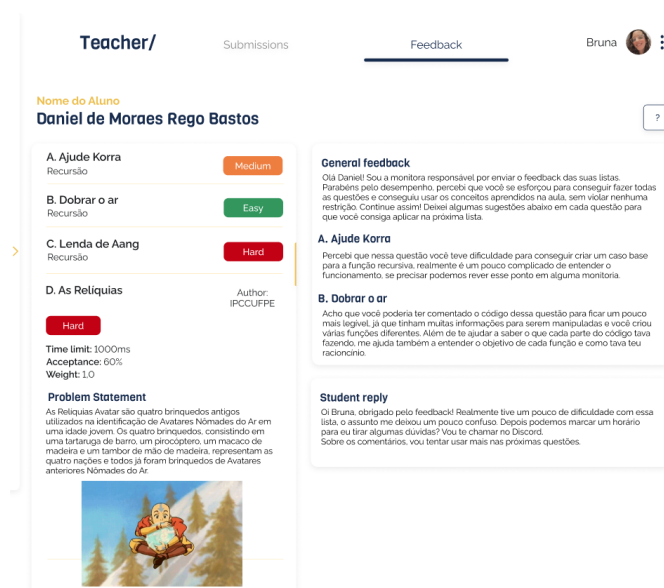


Figura 3.5 Tela de resumo do Feedback concluído

3.3.2.2 Aluno

A entrada para o módulo de feedback para um usuário do tipo aluno será pela tela de atividades. O aluno tem acesso às atividades ativas da sua turma e, ao acessar uma atividade específica, tem acesso a uma tela com problemas da atividade e a uma aba para o feedback (Figura 3.6). A centralização dos feedbacks e dos problemas de uma atividade em um mesmo ambiente é uma forma de incentivar e facilitar o acesso do aluno ao feedback, atacando o problema identificado em relação à forma de envio.

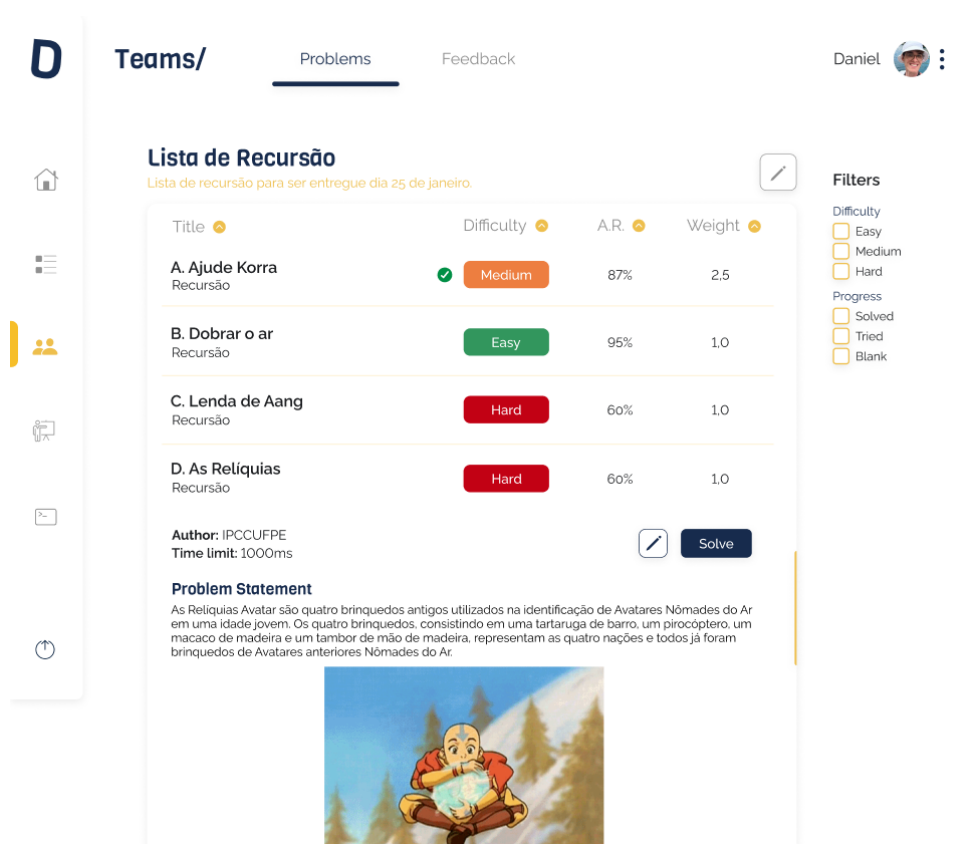


Figura 3.6 Tela de problemas de uma atividade ativa com aba de feedbacks

Ao acessar a aba de feedback, o aluno verá o resumo do feedback enviado, as questões e submissões da atividade, uma checkbox para ser marcada indicando "Eu li meu feedback (*I read my feedback*)" e uma caixa de texto onde ele pode deixar uma resposta para o feedback recebido, tudo isso em uma única tela (Figura 3.7). Essas funcionalidades foram baseadas nas sugestões recebidas e tentam resolver o problema da interação entre o aluno e monitor, de forma a incentivar que o aluno valorize o feedback recebido e demonstre que leu.

The screenshot shows a web interface with a top navigation bar containing 'Problems/' (selected), 'Problems', and 'Feedback'. The user's name 'Daniel' and a profile icon are in the top right. The main heading is 'Lista de Recursão' with a subtitle 'Lista de recursão para ser entregue dia 25 de janeiro'. A link 'I read my feedback' is on the right.

On the left, a list of problems is shown:

- A. Ajude Korra (Recursão, Medium)
- B. Dobrar o ar (Recursão, Easy)
- C. Lenda de Aang (Recursão, Hard)
- D. As Relíquias (Recursão, Hard)

Below the list, a table shows submission details for problem D:

Submission Date	Time	Memory	Language
27/08/2020 12:45	25 ms	60 MB	Python
26/08/2020 12:45	25 ms	60 MB	Python
25/08/2020 12:45	25ms	60MB	Python

The code for the last submission is shown:

```
1 import java.util.Scanner;
2
3 class Main {
4     public static void main(String[] args) {
5         Scanner tn = new Scanner(System.in);
6         int n = Integer.parseInt(tn.nextLine()) + 1;
7         int matriz[][] = new int[n][n];
8         for (int i = 1; i <= n-1; i++) {
9             String nl = tn.nextLine();
10             if (nl.equals("-")) {
11                 matriz[i] = new int[0];
12             }
13         }
14     }
15 }
```

On the right, the 'General feedback' section contains a message from the monitora and two specific feedback items:

- A. Ajude Korra:** Percebi que nessa questão você teve dificuldade para conseguir criar um caso base para a função recursiva, realmente é um pouco complicado de entender o funcionamento, se precisar podemos rever esse ponto em alguma monitoria.
- B. Dobrar o ar:** Acho que você poderia ter comentado o código dessa questão para ficar um pouco mais legível, já que tinham muitas informações para serem manipuladas e você criou várias funções diferentes. Além de te ajudar a saber o que cada parte do código tava fazendo, me ajuda também a entender o objetivo de cada função e como tava teu raciocínio.

Below the feedback is a 'Reply' section with a text area and a 'Send' button.

Figura 3.7 Tela de resumo de feedback com possibilidade de resposta

Após a confirmação de leitura e envio de resposta, a tela passa a exibir um resumo do feedback recebido com a resposta e os problemas da atividade com as submissões do aluno (Figura 3.8), estando concluído o processo.

This screenshot shows the same interface as Figure 3.7, but with the 'Your reply' section filled out. The 'General feedback' and specific feedback items remain the same.

The 'Your reply' section contains the following text:

Oi Bruna, obrigado pelo feedback! Realmente tive um pouco de dificuldade com essa lista, o assunto me deixou um pouco confuso. Depois podemos marcar um horário para eu tirar algumas dúvidas? Vou te chamar no Discord. Sobre os comentários, vou tentar usar mais nas próximas questões.

Figura 3.8 Tela de resumo de feedback recebido com resposta finalizada

3.4 Validação

A solução proposta será validada com turmas de Introdução à Programação e Programação 1 para o envio de pelo menos 3 feedbacks de atividades avaliativas, a fim de verificar a adesão da solução por professores, alunos e monitores. Como forma de dimensionar o alcance dos objetivos deste trabalho, foram definidas as seguintes métricas: taxa de alunos que responderam os feedbacks, quantidade de feedbacks com atraso e comparação da média de nota dos alunos na atividade, com e sem o uso da ferramenta desenvolvida. Essas informações serão facilmente extraídas pelos Dikastis durante o uso do módulo de Feedbacks e serão comparadas com as informações disponíveis e informadas por monitores e professores acerca do período sem a utilização da ferramenta. Para complementar as análises quantitativas, serão feitas novas pesquisas qualitativas e entrevistas com os participantes da primeira etapa, para que sejam ouvidas as percepções após o uso da solução e assim, atestar se ela cumpriu o que estava sendo proposto.

CAPÍTULO 4

Resultados

4.1 Análise do processo de envio de Feedbacks

Os resultados obtidos com as entrevistas e a pesquisa qualitativa foram divididos da seguinte forma: aspectos positivos sobre os feedbacks, desafios enfrentados no processo e sugestões de como melhorar o processo. A intenção dessa divisão é conseguir sintetizar os aspectos do processo importantes para a pesquisa: as evidências da importância dos feedbacks no processo de ensino-aprendizagem e no relacionamento entre alunos e monitores e a identificação do que pode ser resolvido com uma proposta de solução, garantindo que o que foi proposto assegura e incentiva a prática e seus aspectos educacionais e sociais.

4.1.1 Pontos positivos

Ao questionar ex-alunos sobre a experiência que tiveram com feedbacks quando cursaram a disciplina, 75% deles responderam que os feedbacks recebidos foram úteis para o aprendizado e as maiores contribuições técnicas citadas foram: ter melhor noção de organização de código, nomeação de variáveis, uso de comentários, legibilidade e uso eficiente dos recursos da linguagem (noções de otimização). Além disso, alguns participantes comentaram sobre a importância de ter um outro aluno dando suporte, incentivando e criando um canal confiável para pedir ajuda. Alguns depoimentos foram destacados abaixo:

- *"Os feedbacks foram de grande ajuda para eu saber onde eu poderia melhorar meu desempenho e formas de tornar meu código mais organizado e claro para quem estivesse lendo"*
- *"Os feedbacks me ajudaram bastante a compreender alguns pontos básicos que eu poderia melhorar e até mesmo maneiras mais fáceis (ou eficientes) de responder determinada questão"*
- *"Era uma forma de aprender quais eram os pontos que eu poderia melhorar, como comentários de código, nomeação de variáveis"*
- *"Recebi feedbacks que me ajudaram a adotar boas práticas na programação e evitar erros comuns de iniciante que cometia. Sinto que além das correções providas do feedback, uma parte muito valiosa dele era a motivação que o monitor dava para continuar me esforçando e melhorando como programador. Eu via o monitor como uma projeção minha para o futuro"*

Os resultados acima demonstram que o processo de feedback oferecido por um monitor vai além dos feedbacks automáticos disponíveis nos juízes online e é complementar para a formação de profissionais da computação, possibilitando o aprendizado de noções de boas práticas importantes (Robert C. Martin, 2009). Ademais, é notória a relevância da criação de um vínculo com um outro aluno mais experiente no processo de aprendizagem, evidenciando o que foi discutido por Cunha Júnior [2] ao ressaltar que as práticas de monitoria, como forma de espelhar o aluno em um exemplo com o qual ele se identifica, ilustram claramente o teorizado no conceito de Zona do Desenvolvimento Proximal[8].

Em relação ao depoimento coletado de monitores, os principais pontos positivos relatados acerca do processo de escrita de feedback foram: fortalecimento dos conceitos de programação por meio da análise dos códigos, prática do olhar crítico para a revisão dos códigos e exercício de habilidades de comunicação. Foram destacados os trechos a seguir:

- *"Aprendi como transmitir a eles de forma educativa e intuitiva vários conceitos e práticas de programação, sempre me lembrando como uma vez estive no lugar deles e que como era de ter de aprender a programar."*
- *"Escrever os feedbacks me ajuda a fixar alguns conceitos. Dar dicas aos alunos de melhores práticas faz você mesmo refletir sobre essas práticas para colocar elas em vigor e melhorar como programador."*
- *"A leitura de códigos dos alunos é um verdadeiro desafio, principalmente pelo fato que muito estão tendo seu primeiro contato com programação e não conhecem os padrões de boas práticas da linguagem, então aprender a ler, entender e dar dicas de como tornar o código mais legível foram habilidades importantes que desenvolvi."*
- *"Eu acho que a escrita de feedbacks ajuda a desenvolver as habilidades de leitura e revisão de código, [...] além de estimular a empatia e um lado humano em quem escreve (por exemplo, se um estudante não faz nenhuma questão da lista, o ideal é perguntar se tá tudo bem e se precisa de ajuda e estimular o aluno a fazer mais questões da próxima vez.)"*

O apresentado confirma o que é descrito por Gouveia[4], ao discorrer sobre a importância das atividades de monitoria para a consolidação do conhecimento dos monitores, que têm a oportunidade de aprender enquanto ensinam e, nesse caso particular, por em prática habilidades essenciais para os profissionais de desenvolvimento como a análise e entendimento de programas, situação que acontece na vida profissional ao se deparar com códigos desenvolvidos por outras pessoas e precisar entender seu funcionamento e analisar critérios de qualidade, e a comunicação de sugestões e pontos de melhoria com profissionalismo e empatia.

As informações obtidas com professores e monitores-chefe trouxeram uma nova perspectiva sobre o processo de feedbacks, ressaltando os seguintes pontos: importância de um canal confiável para contato direto com o aluno, possibilidade de acompanhamento individual do desempenho do aluno, indicativos de bom desempenho do monitor e possibilidade de fortalecer conceitos aprendidos. Isso pode ser ilustrado nos trechos seguintes:

- *"Quando identificamos algum problema com as atividades de um aluno, como uso de comandos que não estavam permitidos, alertamos no feedback e pedimos que o aluno refaça o código. Isso só é possível por conta do processo de análise individual."*
- *"Normalmente, os monitores enviam os feedbacks de um mesmo aluno o período inteiro e isso ajuda muito a acompanhar a evolução do aprendizado ao longo do tempo. Caso algum aluno tenha mudança súbita de desempenho, o feedback é o canal para oferecer apoio."*
- *"Tentamos acompanhar a entrega dos feedbacks para identificar monitores que enviam feedbacks rasos ou com atraso, pois precisamos garantir que os alunos recebam boas orientações a tempo de aplicarem na próxima atividade"*
- *"Nem sempre conseguimos falar sobre outras habilidade de programação que são importantes além dos conteúdos previstos, o feedback é um espaço de compartilhar alguns conhecimentos e incentivar os alunos a irem além."*

É perceptível como os feedbacks se tornaram uma ferramenta de acompanhamento individualizado em turmas grandes das disciplinas de Introdução à Programação e Programação 1, sendo uma prática que torna o processo mais acolhedor mesmo no ambiente acadêmico, que muitas vezes pode ser menos acolhedor que o ambiente escolar, amenizando o momento de adaptação nos períodos iniciais da graduação. Além disso, é possível observar as práticas do monitor e também orientá-lo em pontos de melhoria, oportunizando seu crescimento.

4.1.2 Desafios do processo

Entre os ex-alunos entrevistados, 25% deles afirmaram não ter achado úteis os feedbacks recebidos durante a disciplina, os principais motivos foram: feedbacks muito genéricos, atrasos no recebimento e poucas sugestões em caso de alunos com bom desempenho. Alguns destaques seguem abaixo:

- *"Recebi os feedbacks das listas 5 e 6 no último dia de aula da disciplina. Então, principalmente nessas duas últimas listas, não senti que eles foram muito úteis para mim."*
- *"Os feedbacks eram mais gerais com alguns pontos específicos, mas podiam faltar alguns pontos do código."*
- *"Os feedbacks que recebi foram baseados em apontar o óbvio: que eu tinha concluído a lista e com isso tinha alcançado a nota máxima."*

A partir disso, pode-se identificar claros problemas na perspectiva do aluno: o atraso no envio do feedback impossibilita que as sugestões e pontos de melhoria sejam aplicados na atividade seguinte e que é preciso uma padronização dos critérios que devem ser analisados para que todos os alunos recebam feedbacks de qualidade, inclusive para alunos que já possuem experiência com programação.

Ao questionar os monitores sobre os desafios no processo de escrita de feedbacks, 40% dos entrevistados afirmaram sentir dificuldade para escrever bons feedbacks aos alunos e as principais dificuldades apontadas foram: não conseguir se dedicar tanto e ter dificuldade em saber o que sugerir ou analisar em casos de alunos com bom desempenho. Percebe-se que esses resultados coincidem com os encontrados a partir dos relatos dos alunos, evidenciando que há a necessidade de uma orientação e padronização do que pode ser analisado e sugerido para alunos que possuem bom desempenho. Também foi perguntado aos monitores quais outros pontos de melhoria eles identificam no processo como um todo e os principais relatos foram os seguintes:

- *"Acho que o envio por email, apesar de ser mais formal, pode distanciar um pouco o relacionamento do monitor com o aluno, pois muitas vezes eu era deixado no vácuo no email, o que é comum em emails, mas dessa forma eu não sabia se o aluno leu ou se importou com o feedback. Com o email ficava um pouco ruim também de citar uma questão específica da lista, eu tinha que dizer o número ou nome da questão, e aí se o aluno quisesse olhar mais de perto o que errou deveria abrir a lista que já passou e olhar lá, sinto que muitas vezes os alunos nem abriam mais as listas que já passaram, mesmo após receber um feedback do monitor."*
- *"Algo que me incomodava era quando os alunos notoriamente não liam meus feedbacks. O que poderia melhorar talvez seria se cada monitor procurasse se aproximar mais dos alunos dos quais eles são responsáveis além de orientar/incentivar os próprios alunos a buscarem esse contato direto com o monitor."*
- *"Acredito que pode haver uma maior padronização dos feedbacks, como um guia sobre os pontos mais comuns a se abordar, a forma de lidar com o erro dos alunos e que contenha instruções para leitura de código, que é uma habilidade essencial."*

Dessa forma, três problemas principais foram identificados na perspectiva dos monitores: as barreiras que a ferramenta atual utilizada impõe, a dificuldade de estabelecer um vínculo e interagir com o aluno e a falta de orientações e padronização na escrita dos feedbacks. Esses pontos evidenciam a necessidade de uma solução que facilite o processo e leve em conta as particularidades da realidade do contexto.

Por fim, os professores e monitores-chefe destacaram os seguintes desafios: não conseguir acompanhar os envios, pois não se tem um meio de verificar diretamente se o monitor enviou ou não o e-mail ao aluno, apenas um registro que depende da confirmação do monitor, e a variação de qualidade entre os feedbacks, já que não há uma verificação ou revisão do feedback nem um "controle de qualidade".

- *"Às vezes só descobrimos que o feedback não tinha sido enviado porque o aluno vinha nos questionar quando todos os outros haviam recebido e ele não. Até tentamos registrar em uma página de controle, mas, muitos esquecem."*
- *"Vamos colocar um outro monitor para revisar o feedback para ver se conseguimos ter feedbacks melhores, mesmo que isso deixe o processo mais trabalhoso."*

O supracitado evidencia que o controle de qualidade do processo de feedbacks, incluindo o seu envio no prazo e seu conteúdo, é um ponto importantíssimo e que precisa ser resolvido, evitando mais complexidade no processo para contornar os problemas.

4.1.3 Sugestões de melhoria

Todos os participantes foram questionados em relação a que sugestões e ideias eles teriam para melhorar e facilitar o processo de envio e escrita dos feedbacks, aqui estão listados os pontos viáveis e relacionados com os problemas identificados acima:

- *"Trazer os feedbacks para o Dikastis [...] assim como as listas, eles poderiam ter o countdown e ficar registrado os que foram entregues com atraso."*
- *"Acredito que ter um retorno do aluno será bem legal pra sabermos se eles estão sentindo falta de alguma resposta."*
- *"Poder enviar o feedback por partes (1 questão por vez), pode deixar a atividade menos cansativa e fazer o aluno ter pelo menos algum retorno, se não todo, o mais rápido possível."*
- *"Acredito que a forma de envio dos feedbacks poderia mudar, buscando um canal de comunicação mais direto com os estudantes, já que o e-mail não costuma ser o meio mais visto e valorizado por eles."*

Com base em todos os problemas identificados e levando em consideração as sugestões dos envolvidos no contexto, foi proposto o módulo de feedbacks dentro do árbitro virtual Dikastis, utilizado nas práticas da disciplina.

4.2 Validação da solução proposta

Como descrito na metodologia, o processo de validação da solução foi definido de forma que os estudantes, monitores e professores conseguissem utilizar o módulo e avaliar se ele consegue resolver os problemas identificados, coletando depoimentos sobre as experiências de utilização e a partir de métricas definidas com informações extraídas sobre o uso da plataforma. Os testes ainda estão em processo de execução, por isso, uma versão completa deste trabalho ficará disponível em https://github.com/brunaalvesws/tg_bruna_alves com os resultados obtidos acerca da validação.

CAPÍTULO 5

Conclusão

Com os resultados obtidos até o presente momento, é possível concluir que dentro do contexto das disciplinas introdutórias de programação dos cursos de Ciência da Computação e Sistemas de Informação, a prática de envios de feedbacks individuais tem um papel crucial para a formação dos estudantes e monitores como profissionais e para o funcionamento das disciplinas de acordo com a metodologia de ensino proposta. Os depoimentos coletados evidenciaram como todo o processo contribui para o aprendizado dos monitores por meio da oportunidade de desenvolverem habilidades técnicas e pessoais, para o aprendizado e acompanhamento dos alunos, que têm a possibilidade de criar um vínculo com o monitor e criar um canal confiável para compartilhar suas dificuldades e ter uma avaliação sobre seu desempenho e para que os professores consigam oferecer um ambiente de aprendizado mais seguro a partir do acolhimento individual.

A análise do processo de elaboração e envio de feedbacks evidenciou problemas que traziam grande impacto para o sucesso dessa prática e embasaram a proposta de solução apresentada neste trabalho, integrando esse processo ao Dikastis, árbitro virtual utilizado nas disciplinas, ampliando suas funcionalidades e estendendo a percepção de feedback: do feedback automático fornecido pelo juiz virtual sobre a execução dos códigos até o feedback individual escrito pelos monitores.

Espera-se que, em breve, seja possível também elaborar conclusões sobre o uso da solução pelos envolvidos e atestar se ela cumpre seus objetivos.

Referências Bibliográficas

- [1] Faheem Ahmed, Luiz Fernando Capretz, Salah Bouktif, and Piers Campbell. Soft skills and software development: A reflection from the software industry. *CoRR*, abs/1507.06873, 2015.
- [2] Fernando Rezende da Cunha. Atividades de monitoria: uma possibilidade para o desenvolvimento da sala de aula. *Educação e Pesquisa*, 43(3):681–694, Jul 2017.
- [3] Rodrigo Francisco, Cleon Pereira Júnior, and Ana Ambrosio. Juiz online no ensino de programação introdutória - uma revisão sistemática da literatura. 10 2016.
- [4] Sarah Gouveia, Natalia Vasconcelos, and Rafael Palhares. Análise da metodologia de ensino da disciplina de programação de computadores sob o ponto de vista dos monitores. 10 2018.
- [5] Hong Cheong Looi and Afzaal H Seyal. Problem-based learning: An analysis of its application to the teaching of programming. *International Proceedings of Economics Development and Research*, 70:68, 2014.
- [6] Robert C. Martin and James O. Coplien. *Clean code: a handbook of agile software craftsmanship*. Prentice Hall, Upper Saddle River, NJ [etc.], 2009.
- [7] Haider Ramadhan and Khalil Shihab. Intelligent systems for active program diagnosis. *Sultan Qaboos University Journal for Science [SQUJS]*, 5:157, 12 2000.
- [8] Lev Semynovich VYGOTSKY. *Thinking and speech*. New york: Plenum, 1988.