Universidade Federal do Estado do Rio de Janeiro

Centro de Ciências Exatas e Tecnologia

Escola de Informática Aplicada

Code Quest

Um ambiente interativo web para ensino de programação

Daniel Oliveira Andrade

João Vitor Silva de Sousa

Orientador

Mariano Pimentel

Rio de Janeiro, RJ - Brasil

Março de 2022

Code Quest

Um ambiente interativo web para ensino de programação

Daniel Oliveira Andrade
João Vitor Silva de Sousa

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Orientador
Mariano Pimentel

Rio de Janeiro, RJ - Brasil

Março de 2022

Agradecimentos

Para começar, gostaríamos de agradecer ao professor Mariano Pimentel, nosso orientador, não só por nos ajudar a alinhavar nossas ideias nesse trabalho, quase num processo de tecelagem coletiva, mas também por acolher a nós e a nossa ideia, e sempre se mostrar presente quando necessário. Muito obrigado por tornar essa (longa) jornada uma experiência prazerosa!

Logo em seguida, não poderíamos deixar de mencionar todos os alunos que participaram das atividades que propusemos. O interesse e a disposição deles, tanto em participar da experiência quanto em fornecer os feedbacks coletados, foram uma peça vital para que o nosso trabalho pudesse dar frutos. Ensinar é uma das poucas atividades que escapa da lógica de soma zero dominante: você pode entregar tudo que tem e sair de lá com mais do que chegou. Obrigado por confiarem em nós.

Por último, mas definitivamente não menos importante, ao curso de Sistemas de Informação da Unirio e seus docentes e discentes, o nosso querido BSI. Sem o ambiente aqui cultivado, nenhuma dessas ideias teria encontrado o solo fértil de que precisavam. Obrigado a todos os envolvidos.

Agora, ao separar nossas narrativas, eu, João Vitor, gostaria de deixar os meus mais sinceros agradecimentos a todas as pessoas que, em algum momento nesses últimos 5 anos, tiveram um impacto positivo na minha vida, independente de quão grande tenha sido. Felizmente, agradecer cada uma aqui não seria viável, então me restrinjo aquelas cujo impacto foi imensurável: meus pais, João Carlos e Maria Elizabeth, que sempre me apoiaram e fizeram de tudo pra que eu pudesse alcançar todos os meus objetivos; minha irmã, Joice, minha maior companheira, por sempre tornar minha vida mais leve e alegre; meu grande amigo, Daniel, com quem escrevo este trabalho, por ter me guiado em incontáveis momentos durante a jornada da graduação; as amizades que surgiram da Unirio, em especial Wesley, por ter entrado em minha vida no momento em que eu mais precisava; e as amizades que vem de muito antes, em especial

Natália, por me acompanhar numa jornada que vai muito além desses 5 anos. Muito obrigado a todos!

E, antes de encerrar, eu, Daniel, preciso agradecer o apoio que recebi nesses anos todos, e não foi pouco. Da minha família que, de alguma forma, sempre acreditou em mim. Do João, que comprou a ideia e se juntou a mim nas trincheiras. Dos meus amigos, que tornaram este caminho infinitamente mais rico (e, em especial, os que encontrei aqui e, agora, seguem comigo pra vida). Dos professores com quem pude trocar. E, claro, de todos que vieram antes e abriram meus caminhos. Obrigado.

RESUMO

Nosso trabalho busca expandir os horizontes do que é possível no ensino da programação, um desafio que tem se mostrado resiliente aos esforços dos cursos de tecnologia. Nossa inspiração veio justamente de participar desse processo e lá observar o potencial não explorado dentro de sala de aula, de ver que era possível ir além.

Com esse fim, construímos uma plataforma que pudesse ser usada dentro de sala de aula para produzir uma dinâmica de ensino diferente. Nela, o aluno primeiro tenta resolver os exercícios, e depois, caso necessário, recorre ao professor para a explicação. Essa dinâmica produz um anseio em busca da resposta no aluno, ele passa a ter uma motivação intrínseca para aprender. São os resultados desse esforço que esse trabalho relata.

Palavras chave: Informática na educação, ensino de programação, metodologias ativas, pedagogia

ABSTRACT

Our work seeks to expand the horizons of what's possible when it comes to teaching how to code, a challenge that has proven resilient to the efforts of technology courses. Orange duration came precisely from participating in this process and observing the untapped potential Westin the classroom, from seeing it was possible to go beyond.

To that end, we've built a platform that could be used within the classroom to produce a different learning dynamic, one where the student first tries to solve the problem, and later, if necessary, turns to the teacher for the explanation. This dynamic produces a longing for an answer within the students, they now have an intrinsic motivation to learn. The results of this effort compose this body of work.

Palavras chave: Educational technology, teaching programming, active methodologies, pedagogy

LISTA DE ILUSTRAÇÕES

Sumário

1. Introdução

Num tempo em que vivemos rodeados por tantos produtos e soluções para todo e qualquer problema imaginável, nem sempre o valor de construir seu próprio caminho fica evidente, é muito mais fácil ir atrás de uma solução já pronta na prateleira, já pensada e digerida pelo mercado.

Escolhemos o caminho mais difícil, e o que segue é o relato de como isso se deu e do que encontramos durante essa jornada. Existe um poema chamado "The Road not Taken", ou A Estrada não Percorrida, escrito por Robert Frost, que termina assim:

> "Eu tomei a estrada menos percorrida,
> E isso fez toda a diferença."

Hoje, muito se debate sobre o significado desse final, mas, para nós, ter tomado a estrada menos percorrida fez *mesmo* toda a diferença.

1.1 Motivação

Já tendo experimentado e alcançado resultados promissores com metodologias ativas, nas quais a prática e a experimentação são a parte principal e a teoria é exposta somente quando necessária, decidimos elaborar um meio de ensinar programação pela prática, cobrindo os conteúdos de uma disciplina de "Introdução à Programação".

A dificuldade para aprender a programar afeta incontáveis estudantes. Assim, uma das motivações principais dos autores deste trabalho é auxiliar tanto os professores quanto os alunos de programação em suas jornadas, contribuindo na criação de um modelo que produza uma experiência mais efetiva e prazerosa.

Além disso, foi identificado um problema até então pouco explorado: em vista da natureza altamente volátil da área de tecnologia da informação, identificamos que uma das competências

mais importantes a serem trabalhadas no curso seria a geração de agência e autonomia por parte dos estudantes na aquisição de conhecimento, e o ensino de programação se mostrou, inicialmente, um momento interessante para estimular esse tipo de habilidade.

1.2 Problema

O ensino de programação tem sido amplamente estudado. Muitos professores ainda encontram dificuldades nessa tarefa, fazendo com que o aprendizado dos alunos acabe sendo, muitas vezes, comprometido. Diversos autores apontam a disciplina de Introdução a Programação como sendo uma das disciplinas que mais reprovam nos cursos de Computação (BOSSE; GEROSA, 2015).

Os problemas enfrentados pelos professores são bastante variados, indo de limitações quanto à disponibilidade de computadores para aulas práticas à falta de boas metodologias para o desenvolvimento da habilidade de programar. A utilização de aulas puramente expositivas com exercícios esparsos é um dos principais desafios reconhecidos no ensino de programação (HOLANDA; FREIRE; COUTINHO, 2019), sendo esse o problema que procuramos mitigar com o presente trabalho.

É comum que os professores utilizem o ambiente de sala de aula primariamente como um espaço para trabalhar os aspectos teóricos das disciplinas de programação, deixando a prática para momentos isolados após uma exposição longa de conceitos, ou como exercícios extraclasse. Nossa ideia é que **entremear melhor prática e teoria durante as aulas seria um modelo mais eficaz**. Para pôr a teste esta teoria, desenvolvemos o ambiente interativo Code Quest. O processo de desenvolvimento e os aprendizados obtidos construindo e testando-o é o que discutiremos a seguir.

1.3 Proposta de solução

Em resposta ao problema descrito, desenvolvemos uma plataforma que apresenta uma série de desafios de programação sequencialmente mais difíceis, cobrindo as aptidões fundamentais no

aprendizado da programação, desde operações sobre inteiros e manipulação de textos até a lógica por trás de funções e estruturas simples de dados.

Em especial, a plataforma apresenta não só os problemas, como também uma forma de testar se a solução criada pelo aluno está correta através de uma série de casos de teste, permitindo assim que o aluno possa obter um retorno sobre o seu progresso sem depender da intervenção imediata de um professor.

1.4 Objetivos

Nossas expectativas eram que os alunos, uma vez expostos a essa metodologia, consigam não só adquirir conhecimentos técnicos sobre a linguagem utilizada e seus construtos, como também construir conhecimentos sobre o funcionamento de algoritmos, estruturas de dados e do pensamento computacional como um todo.

Concomitantemente ao processo de aprender a programar programando e tendo *feedback* automático sobre o código produzido, espera-se que os alunos se habituem com a ideia de buscar o conhecimento que precisam para resolver problemas apresentados em outras fontes, como por exemplo na internet ou consultando seus pares, criando assim agência sobre a aquisição de conhecimento e estimulando sua auto-confiança na sua capacidade de aprender.

1.5 Metodologia

Para produzir este trabalho, desenvolvemos uma plataforma de apoio ao processo de ensino-aprendizagem de programação. Em vez de desenvolvê-la por completo antes de aplicá-la em um cenário real, optamos por fazer a mínima aplicação viável e testá-la o mais rápido possível. Dessa forma, pudemos coletar *feedbacks* importantes durante o processo de desenvolvimento e evitar retrabalho, testando nossas hipóteses o mais cedo possível.

No semestre em que aplicamos nossa plataforma, sabíamos que ela não seria capaz de atender ao propósito de cobrir todos os conteúdos previstos no conteúdo programático da disciplina. Assim, focamos nos conteúdos mais fundamentais que compunham o início do curso. Ao término do semestre, realizamos uma pesquisa com os alunos que participaram do estudo de caso para colher dados quantitativos e qualitativos sobre o uso da plataforma.

Com base nos resultados obtidos através dessa experiência, prosseguimos desenvolvendo a plataforma para então apresentar uma versão mais robusta no período subsequente, durante o qual esperamos conseguir analisar o que os alunos acharam da plataforma experienciada, focando nos aspectos positivos e negativos que a plataforma proposta teve na jornada de aprendizado dos mesmos. Infelizmente, devido a pandemia do Coronavírus (COVID-19), não foi possível realizar a aplicação da plataforma em um segundo momento.

1.6 Organização do texto

Este trabalho está segmentado em 6 seções diferentes. A seção inicial tem o objetivo de apresentar o projeto e as ideias por trás dele. A seção seguinte trata da fundamentação teórica, abordando os principais conceitos e evidências que embasam este projeto. A terceira descreve o processo de desenvolvimento da plataforma em si, abordando desde sua conceitualização às tecnologias que foram utilizadas para a implementação da mesma. Na seção 4, falamos sobre a aplicação da plataforma em um cenário real e analisamos seus resultados. Na seção 5 discutimos sobre o processo de refinamento aplicado na plataforma após sua aplicação. Por fim, a seção 6 abrange as conclusões obtidas através da realização deste trabalho.

2. Fundamentação teórica

Seria arrogância nossa acreditar que nossas ideias são originais. Reconhecendo o interminável processo que é o desenvolvimento do conhecimento nas sociedades humanas, buscamos refletir e reproduzir em cima dos acúmulos que nos precederam.

Nessa jornada, nos deparamos com boas lições que nos ajudaram a navegar o que estávamos fazendo e evitar certos perigos. Assim, aqui compartilhamos o que aprendemos ao longo do caminho, e como esses aprendizados moldaram o nosso trabalho.

2.1 Máquinas de ensinar e o cibertecnicismo

A ideia de que uma máquina pode ser usada para ensinar emerge junto da modernidade, no final do séc. XIX, no auge da revolução industrial, e o registro de centenas de patentes de máquinas de ensinar ilustra bem o que desejava nossa sociedade à época (WATTERS, 2015).

Com suas engrenagens, molas e alavancas, a sociedade estava deslumbrada com a automação de tudo, estava ansiosa para tirar o humano de cena, considerando-o quase obsceno[1] em meio a tanta inovação. Com a revolução digital, essas máquinas de ensinar alcançaram um outro patamar. Ludy Benjamin (1988) estabelece os seguintes critérios para definir/caracterizar uma máquina de ensinar:

(a) Apresenta uma unidade de informação. (b) provê algum meio para que o aluno responda a essa informação, e (c) retorna feedback sobre a corretude das respostas do aluno.

As máquinas de ensinar, populares no século passado, não pararam no tempo. A revolução digital que vivemos está levando essas ideias novamente a alçar voo dentro dos círculos acadêmicos e escolares. O computador tornou-se o meio ideal para produzir essas máquinas de

[1] Obsceno, inclusive, é aquilo que não seria encenado no teatro grego e sim narrado, “ob cena”, fora da cena.

ensinar-aprender, inclusive dando-nos a possibilidade de medir e controlar cada vez mais o processo.

Cabe ressaltar que esse modelo, que busca substituir um professor por uma máquina que entrega estímulos negativos ou positivos, limita o potencial da sala de aula à mera mudança de comportamento, reduz toda a complexidade do fenômeno de ensino-aprendizagem à mera instrução, tendo como base a teoria comportamentalista promovida nos experimentos de B. F. Skinner.

Famoso por ter demonstrado que é possível condicionar o comportamento de animais (cães, pombos, ratos e crianças) por meio de recompensas positivas ou negativas em função da resposta dada pelo animal a um estímulo (condicionamento operante), a teoria comportamentalista e a caixa de Skinner também possibilitam a realização de um experimento sobre o potencial da recompensa ilimitada na formação do vício durante a década de 1960, no auge do proibicionismo às drogas ao demonstrar que um rato escolheria a droga à vida se dado a escolha; desde a década de 80, os resultados desse experimento tem seus resultados questionados por pesquisas que demonstraram que, quando esses ratos não estavam isolados em gaiolas, e sim vivendo em comunidades saudáveis, o fenômeno observado por ele não se reproduzia (ALEXANDER, 2010). No campo da educação, a corrente tecnicista que foi impulsionada pela teoria comportamentalista e pela técnica de instrução programada foi amplamente questionada pelas teorias críticas e pós-críticas do currículo, principalmente a partir da década de 1970.

Ao esquecer o caráter social do aprendizado, as soluções baseadas na "técnica (comportamentalista) de ensino" permanecem incompletas, seus potenciais esmagados sob os anseios pelo que Freitas chama de "neotecnicismo digital" (FREITAS, 2021), o velho desejo de aproximar cada vez mais a sala de aula de mais uma linha de produção dentro do complexo industrial da sociedade.

Ao falar das principais recomendações feitas durante o 1º Seminário Nacional de Informática na Educação em 1981, Moraes destaca uma que segue sendo relevante: *"O*

computador foi reconhecido como um meio de ampliação das funções do professor e jamais como forma de substituí-lo" (MORAES, 1997, p. 4)

Reconhecemos que a plataforma que desenvolvemos neste trabalho pode ser caracterizada como uma máquina de ensinar. Contudo, compreendemos seu lugar dentro do contexto de uma turma, promovendo a discussão em pares de programadores e com mediação de um professor de programação, conforme discutido nos próximos capítulos.

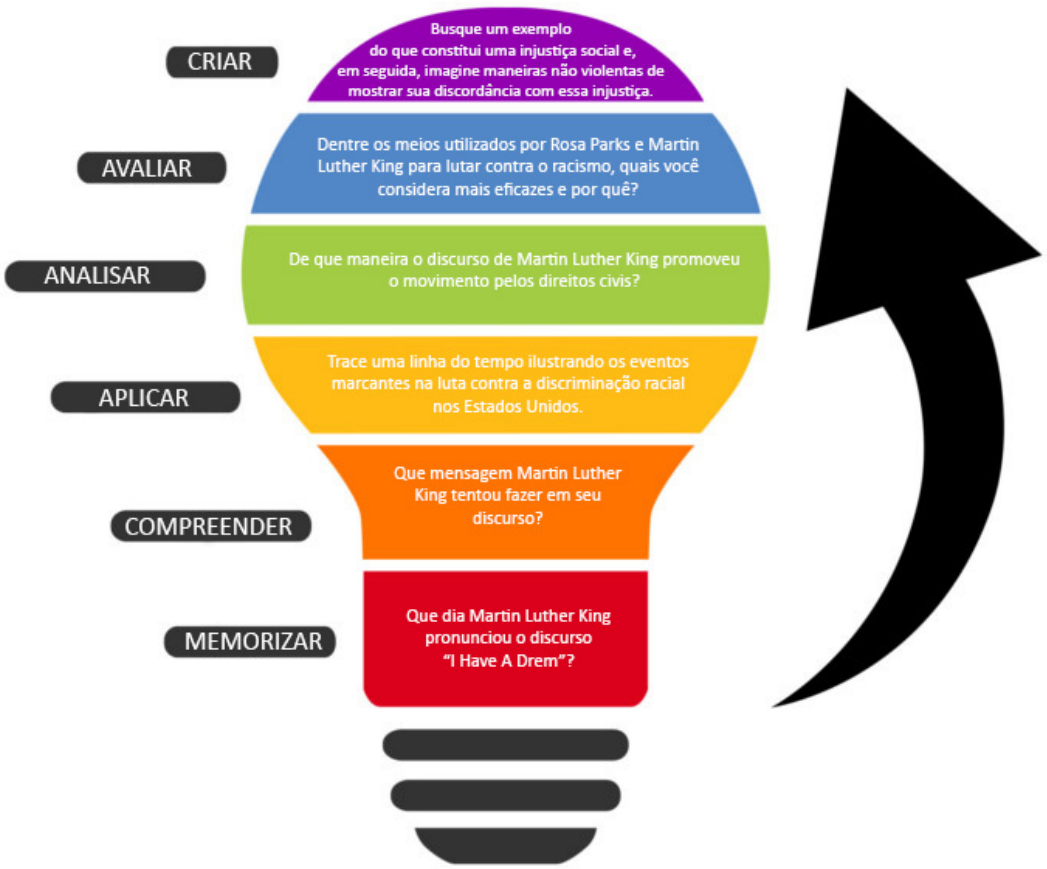
2.2 Metodologias ativas e a zona de desenvolvimento próximo

Conceder maior independência para o aluno no seu processo de aprendizado não significa, necessariamente, torná-lo um autodidata em um processo individual. Desde que Vygotsky introduz a noção de zona de desenvolvimento proximal, observando que o processo de aprendizado se potencializa pela relação entre um aluno e um outro indivíduo que detém o conhecimento mais avançado, normalmente o professor, mas também seus colegas de turma, se evidenciou que o aprendizado é uma atividade coletiva, uma relação dialética.

De lá para cá, nos aprofundamos mais nos meandros do que isso implica. Hoje dispomos do conceito de metodologias ativas, que põem o aluno como agente do seu aprendizado. Esse modelo se distancia da educação bancária, criticada por Paulo Freire, em que o aluno é tratado como uma entidade passiva, que nada sabe e no qual é despejado conteúdo: o aluno lê textos, assiste (vídeo) aulas, sempre absorvendo o conhecimento, mas sem promover seu pensamento crítico e criativo.

Entendemos que aprender não é apenas conseguir recordar e compreender informações (níveis cognitivos inferiores), mas também aplicá-las a problemas concretos, analisá-las de forma crítica, avaliando-as diante dos seus demais conhecimentos e, por fim, produzindo novas informações a partir desse conhecimento. É escalar a pirâmide taxonômica de Bloom, presente na Figura 01 a seguir, e alcançar os níveis superiores dessa pirâmide é uma tarefa que requer atribuir ao aluno um papel ativo/autoral no processo de construção de conhecimentos novos (GAROFALO, 2018).

Figura 01. Taxonomia de Bloom



Fonte: Internet (https://www.tactileo.com/br/educacao-basica/taxonomia-de-bloom-e-digital-learning/)

2.3 A batalha pelo "nós"

No final do dia, o que lutamos aqui é uma das batalhas contra a sociedade do "eu" e pelo resgate do "nós". Acreditamos que a tecnologia da informação, como todas as ferramentas produzidas por nós, tem um vasto potencial, pro bem e pro mal, e esse potencial foi sempre melhor aproveitado quando usadas para nos unir ao invés de separar.

Em uma das pontas, a tecnologia pode ser vista como uma ponte até um futuro distópico onde somos mais máquinas do que homem, onde o pensamento crítico não tem mais espaço e a primazia da dicotomia entre certo e errado impera.

Mas, na outra, existe um lugar onde a tecnologia nos empodera a expressar nossa criatividade, onde nossa capacidade crítica sai enriquecida e estimulada, um lugar no qual o professor pode ser mais um par e menos um chefe, e onde seus pares, por sua vez, deixam de ser competidores para ser colaboradores.

Muitos diriam que a área da tecnologia da informação está dentro das ciências exatas, aquela onde apenas uma resposta é correta, e as outras são consideradas erradas. Porém, ao realizar uma atividade como corrigir uma prova de programação, é possível perceber que isso não é necessariamente verdade. Programar é uma atividade onde o aluno tem a capacidade de se expressar, de montar o seu raciocínio como um artista com um pincel. Mil algoritmos podem produzir as mesmas respostas de formas inteiramente diferentes, e poucas coisas são mais ricas que o processo de entender os raciocínios que os produziram.

Nossa luta aqui é para que os alunos futuros não sejam privados dessa experiência criativa.

3. Desenvolvimento da plataforma

Nesta seção, iremos apresentar o processo de desenvolvimento do Code Quest, decorrendo sobre etapas como conceitualização, metodologias consideradas e tecnologias adotadas. Além disso, também apresentaremos a plataforma nesta seção.

3.1 Usando o Code Quest

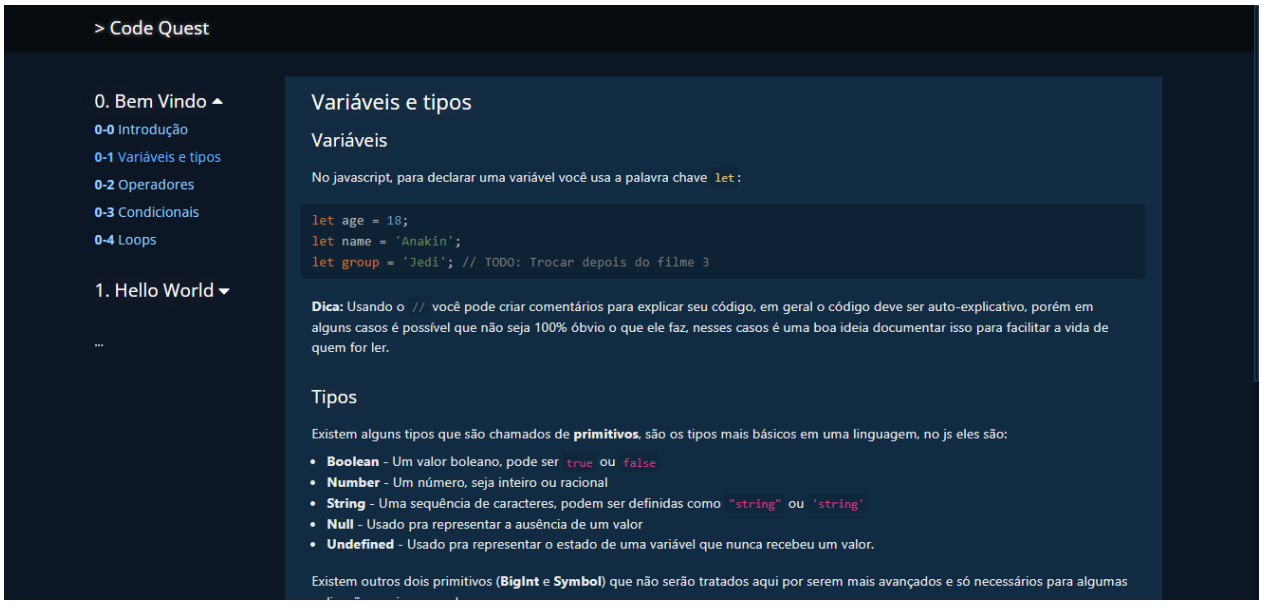
O Code Quest foi pensado para ser uma plataforma de estudo e exercícios. O aluno, ao acessar a plataforma, visualiza a página inicial de boas vindas ilustrada na Figura 02. A partir dessa tela, pode começar a estudar e resolver exercícios sem ter que criar uma conta ou realizar um login.

Figura 02. Tela inicial do Code Quest

Ao entrar no Code Quest, na página inicial o usuário encontra uma rápida introdução sobre a plataforma. O usuário pode navegar entre os tópicos pela barra lateral de navegação ou por meio dos botões que encontram-se no final de cada página, que levam para os tópicos adjacentes na linha de progressão planejada (anterior e próximo).

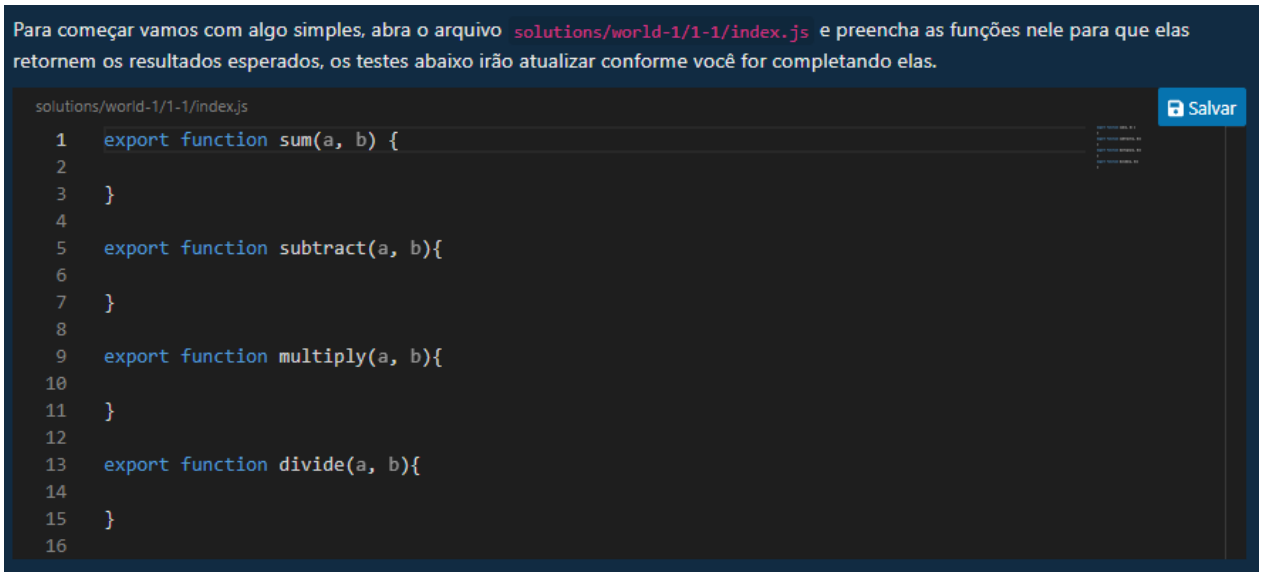
Os tópicos encontrados na plataforma estão divididos em duas seções (ou “níveis”). A primeira seção aborda os conceitos de programação e Javascript; serve como fonte de informações para fazer uma revisão antes que o usuário comece a fazer os exercícios, ou para tirar eventuais dúvidas durante a interação com o sistema (caso o usuário esqueça ou se confunda em relação aos conceitos de programação). A Figura 03, sobre “Variáveis e tipos”, exemplifica essa seção conceitual.

Figura 03. Página “Variáveis e tipos” da seção conceitual do Code Quest



Na segunda seção, o usuário se depara com exercícios propostos. Cada exercício consiste em uma função que deve ser implementada de tal modo que o código tenha o comportamento esperado. O usuário adiciona código dentro da função previamente criada (os cabeçalhos das funções-exercícios já encontram-se criados na plataforma), conforme exemplificado na Figura 04.

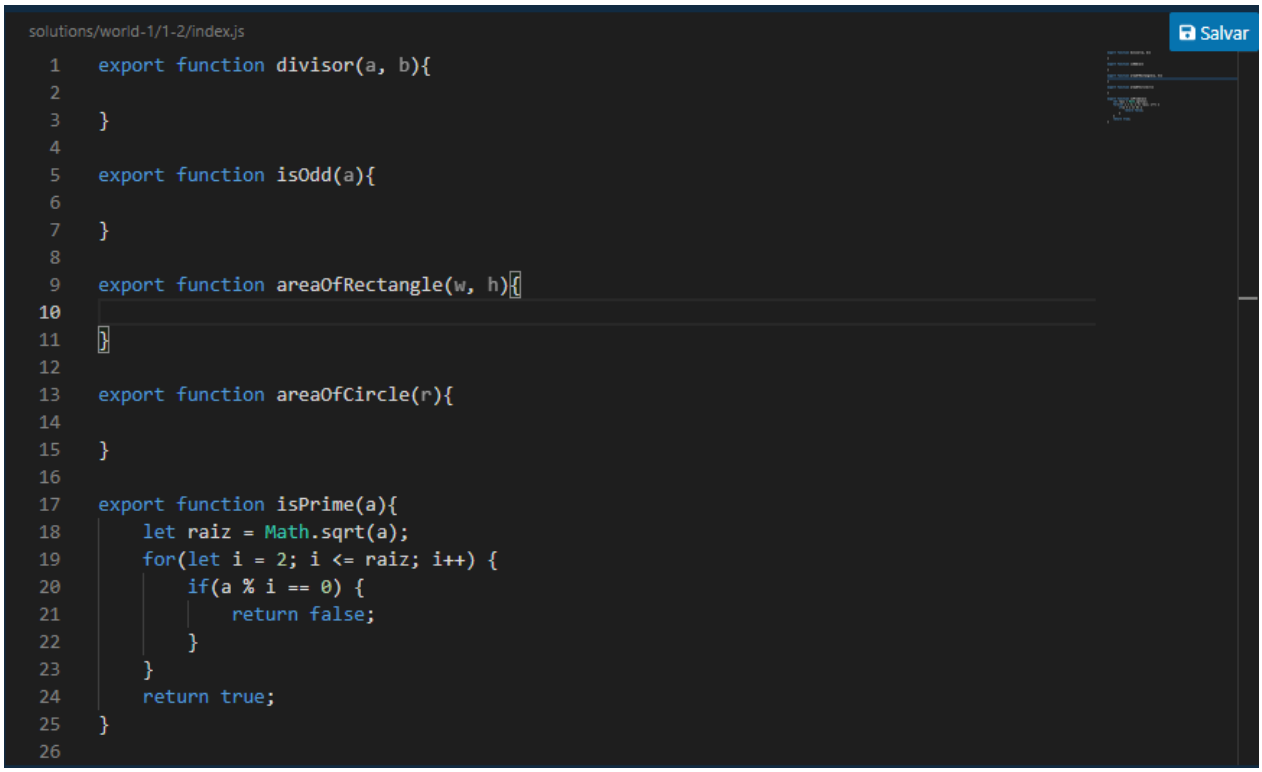
Figura 04. Funções-exercícios para o usuário codificar (possuem o cabeçalho previamente declarado)



Para a codificação, o usuário tem acesso a um editor dentro da própria plataforma. Caso deseje, o aluno também pode utilizar uma IDE de sua escolha e editar os arquivos presentes na pasta do projeto, em vez de utilizar o editor de código disponível na plataforma.

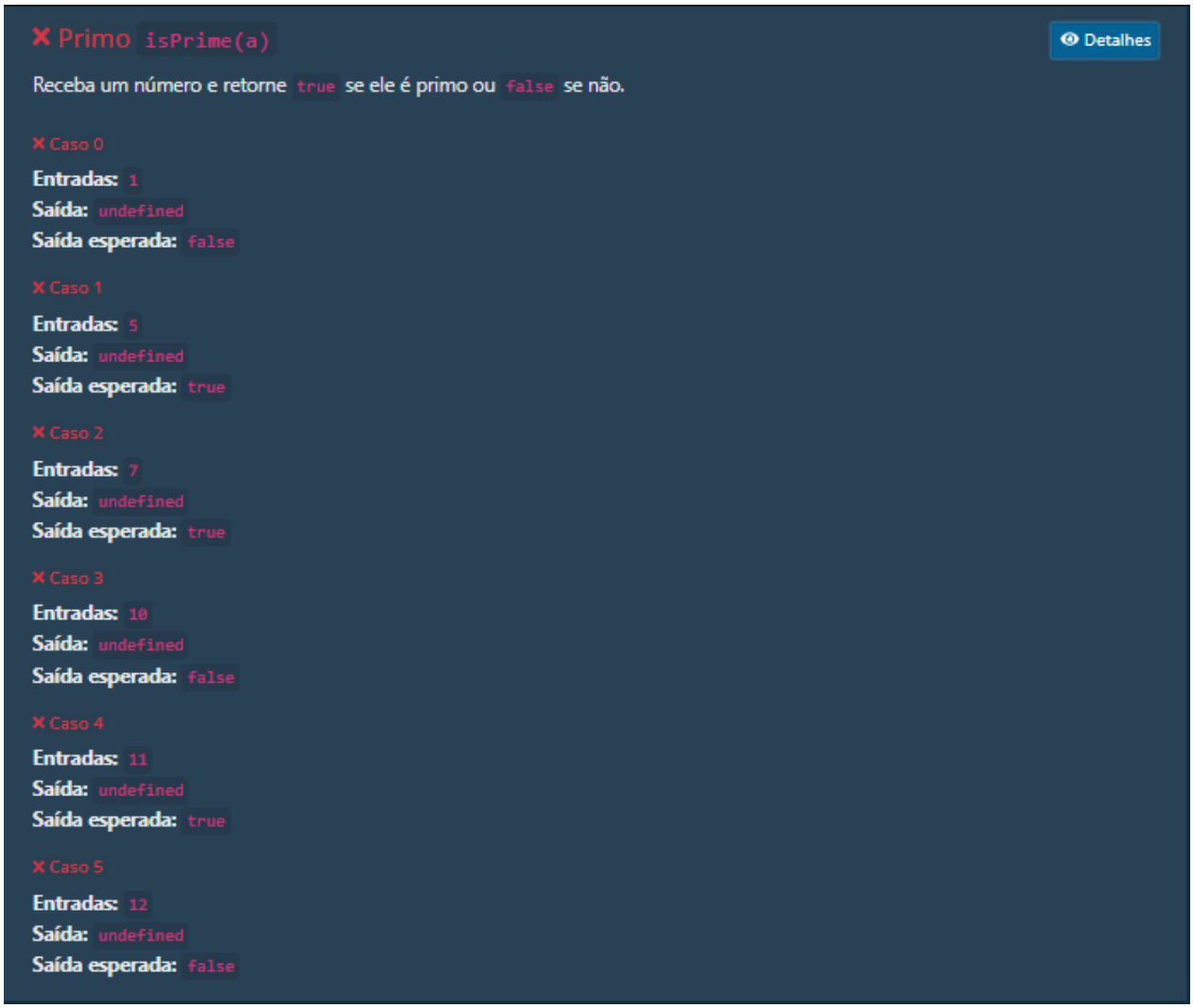
O usuário deve implementar um código para cada função-exercício. Na Figura 05 está ilustrado um código para a função isPrime.

Figura 05. Código introduzido pelo usuário na função isPrime



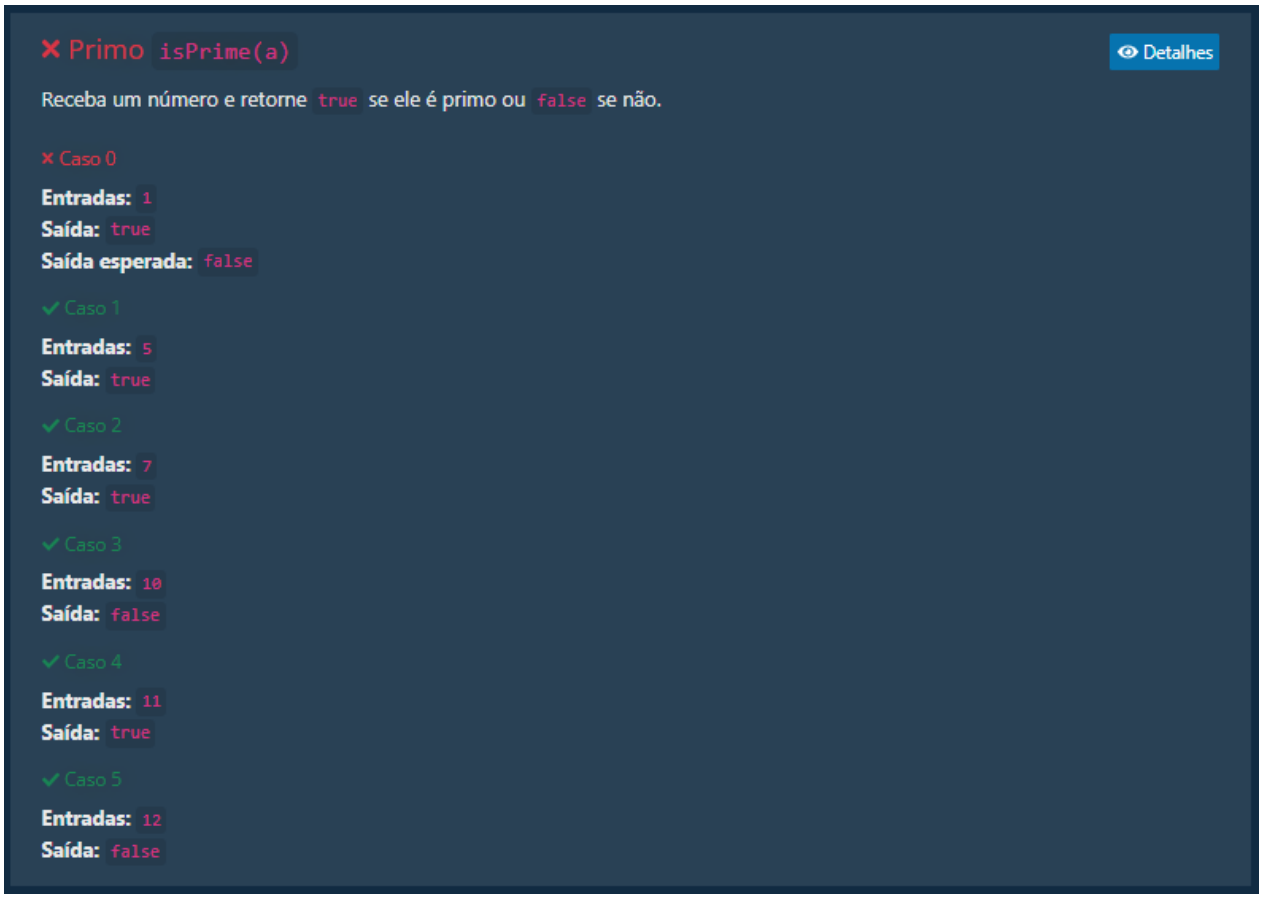
Após codificar, o usuário deve clicar no botão “salvar” (canto superior direito na tela da Figura 05). Ao salvar o código criado, a plataforma é atualizada para indicar se a função parece estar codificada corretamente. Para isso, o Code Quest utiliza múltiplos casos de teste. Um caso de teste especifica o que deve ser retornado da função em um caso específico, i. e., a resposta esperada para uma dada entrada. Na Figura 06 são apresentados os casos de teste projetados para a função-exercício isPrime.

Figura 06. Casos de teste para a função isPrime (sem implementação ainda)



Após o botão Salvar ser clicado, a plataforma apresenta um relatório informando se o código do usuário produziu a resposta esperada em cada caso de teste, conforme exemplificado na Figura 07.

Figura 07. Resultados da verificação se o código implementado passou em cada caso de teste



Não é feita uma verificação se o código do usuário está realmente correto; o que é verificado é se o código implementado produz o resultado esperado em cada caso de teste. Dessa maneira, o usuário é informado sobre quais casos o código já está resultando na resposta correta e quais casos estão resultando em resposta errada (não esperada). Ao perceber que um caso de teste não está resultando na resposta esperada, o aluno pode tentar identificar um problema no código com relação àquela entrada específica. Com esse processo de codificação auxiliado com correções da plataforma com base em casos de teste, o aluno consegue ter informações sobre a corretude de seu código e assim progredir para exercícios cada vez mais difíceis de serem resolvidos.

3.2 A conceitualização da plataforma

A ideia surgiu inicialmente a partir dos desafios encontrados enquanto atuando como monitores de disciplinas como Técnicas de Programação 1 e Desenvolvimento de Páginas da Web, ao percebermos que ambas necessitavam de um componente prático muito forte para efetivamente consolidar a aquisição do conhecimento que se desejava passar. Porém, por se tratarem de disciplinas iniciais da grade curricular, pedir que os alunos realizassem tarefas práticas sem uma estrutura bem estabelecida criava múltiplas potenciais armadilhas que as tornavam pouco eficientes do ponto de vista do ensino e desestimulava sua realização.

Dessa forma, a solução que encontramos está no meio do caminho entre uma tarefa prática livre, ou seja, uma na qual o aluno deve ser responsável por criar as condições para realizá-la, assim como definir como irá solucionar o problema proposto, e as questões mais tradicionalmente propostas pelo sistema educacional, onde reduz-se esse universo de possibilidades a um subconjunto de opções. Além disso, a plataforma agrega um componente de feedback que permite que o aluno saiba como está se saindo sem depender de um agente externo como um monitor ou professor.

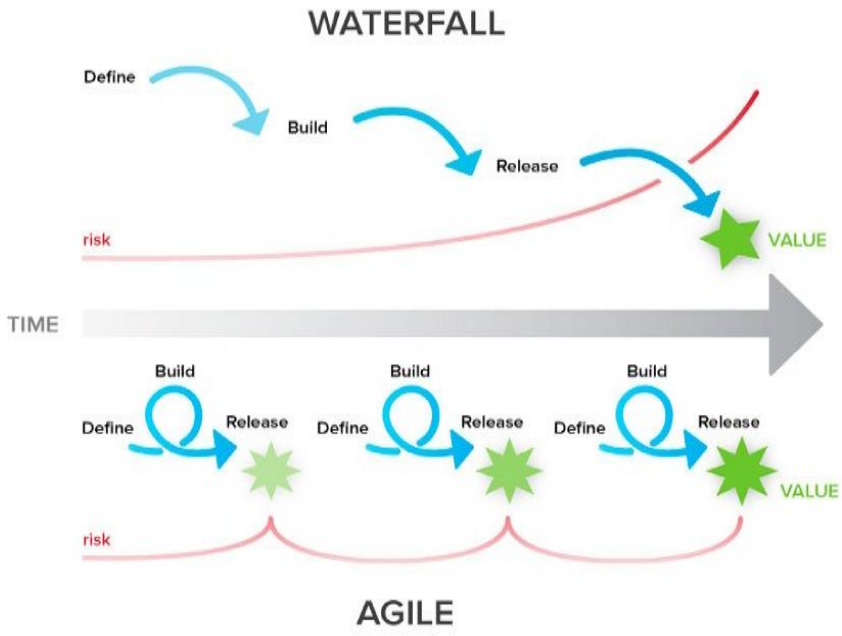
Ainda assim, a plataforma não pretende, nem conseguiria, substituir o papel do educador no processo de aprendizado, dado que ainda é muito comum que o aluno sinta a necessidade de tirar dúvidas e esclarecer os conceitos, principalmente enquanto ele ainda está se acostumando a aplicar os conhecimentos de programação na prática. Nesse sentido, a plataforma age enquanto facilitadora desse processo de refinamento do aprendizado, que precisa acontecer após a exposição tradicional. Seu objetivo é proporcionar uma experiência de aprendizado mais independente para o aluno, ao diminuir o grau de dependência que o aluno tem do professor durante o processo de aprendizado, ao invés de acabar totalmente com o mesmo.

3.3 O processo de desenvolvimento utilizado

Sabendo que estávamos construindo algo cuja forma ainda não era perfeitamente conhecida *a priori*, estávamos cientes de que o processo de desenvolvimento seria bastante iterativo, ou seja, progressivas de camadas de complexidade seriam adicionadas conforme se demonstrassem necessárias.

O único software que não muda, dizem, é o que não vai ser utilizado, e isso é particularmente válido durante a infância de um software, no momento frágil enquanto ele ainda não foi defrontado com as demandas da realidade. Portanto, ao invés do modelo tradicional de desenvolvimento em cascata, nosso processo de desenvolvimento de software se assemelhou mais aos processos ágeis, usando dos seus ciclos mais curtos de desenvolvimento intercalados com testes práticos que, por sua vez, informariam os objetivos dos ciclos seguintes de desenvolvimento. A Figura 08 a seguir ilustra uma comparação entre o modelo cascata e um modelo ágil.

Figura 08. Comparação entre modelo cascata e modelo ágil



Fonte: Internet (https://www.zeitspace.com/blog/how-your-mvp-is-failing-you-and-your-users)

3.4 Tecnologias utilizadas

A plataforma é construída em cima de tecnologias web, nominalmente: Javascript, html e css. A aplicação usa o runtime do Node para inicializar um servidor local de desenvolvimento que serve as páginas da web, acessíveis de qualquer navegador na máquina. A decisão por usar esta pilha de tecnologias se deve a sua maturidade e facilidade de uso, ambos devidos a anos de trabalho e infinitas contribuições pela comunidade, além de sua relativa universalidade hoje, o que aumenta a facilidade de manter o código vivo mesmo conforme o ambiente tecnológico a sua volta evolua (sistemas operacionais e demais contextos).

Mais especificamente, no front-end utilizamos o Vue, uma framework que torna o desenvolvimento de aplicações web muito mais expressivo, ou seja, facilita com que o desenvolvedor expresse a sua intenção através do código. O servidor de desenvolvimento dentro do qual a aplicação roda é o Vite, que por sua vez se baseia no Rollup, e se encarrega de ler os

arquivos de código fonte, processá-los e prepará-los para serem consumidos em um navegador, inclusive fazendo isso em tempo real de forma que qualquer modificação realizada poderá ser vista imediatamente. O nome dessa técnica é Hot Module Replacement, HMR, e é através dela que conseguimos que o código produzido pelos alunos seja processado e avaliado em tempo real no navegador.

4. Aplicação da plataforma e seus resultados

Neste capítulo iremos discutir sobre a aplicação do Code Quest em um cenário prático, mais especificamente, a utilização do sistema durante a disciplina Técnicas de Programação 1 do curso de Sistemas de Informação da UNIRIO, ofertada no segundo semestre de 2019.

4.1 A utilização da plataforma em uma disciplina

Durante o segundo semestre de 2019, o professor Mariano Pimentel foi o professor responsável por lecionar a disciplina obrigatória de Técnicas de Programação 1 para os alunos do primeiro período do curso de Sistemas de Informação da UNIRIO. Sendo assim, foi acordado que os alunos responsáveis por este trabalho poderiam utilizar das aulas da disciplina para testar a plataforma desenvolvida, avaliar seu desempenho e coletar feedback dos alunos.

Foram realizados três encontros no total, e em cada encontro os alunos da disciplina tiveram um limite de tempo para resolverem exercícios da plataforma, com a presença do professor e dos criadores do Code Quest para auxiliar no uso da plataforma, tirar eventuais dúvidas que os alunos pudessem ter, analisar o desempenho dos alunos, tanto com relação a disciplina quanto com relação a utilização da plataforma, e coletar o feedback dos alunos.

4.1.1 O primeiro encontro

No primeiro encontro os alunos da disciplina foram apresentados ao Code Quest e foram instruídos a como instalar e acessar a plataforma. Após o período inicial de configuração, os alunos seguiram realizando os exercícios iniciais presentes na plataforma (níveis 1 e 2 da segunda seção). Durante o encontro, ajudamos os alunos com as dúvidas encontradas, tanto com relação à plataforma quanto com os exercícios propostos, analisando os problemas encontrados. No final da atividade, foram feitas perguntas aos alunos sobre as atividades realizadas e sobre a utilização da plataforma.

Com relação aos pontos negativos, os alunos mencionaram a dificuldade de começar a utilizar a plataforma, devido a necessidade de instalação das dependências e a utilização de múltiplos comandos desconhecidos; a falta de uma explicação detalhada sobre o que cada exercício espera deles; e a falta de uma fonte de informação dentro da própria plataforma que ajude eles a resolver os exercícios, como dicas sobre a sintaxe da linguagem Javascript. Como pontos positivos, os alunos citaram que a plataforma proporcionou um bom engajamento, além de incentivar a criatividade na resolução dos exercícios. Além disso, a interface foi elogiada, e a funcionalidade de feedback automático das resoluções também foi considerada um ponto positivo. Por fim, alguns alunos disseram terem gostado da presença de exercícios envolvendo matemática, e acharam os exemplos desafiadores.

4.1.2 O segundo encontro

Para o segundo encontro, foram feitas algumas modificações na plataforma (que serão discutidas detalhadamente no capítulo seguinte) com o objetivo de aprimorar a experiência dos alunos com base no feedback coletado ao final do primeiro encontro. As atividades propostas foram semelhantes as do encontro anterior, porém dessa vez os alunos alcançaram exercícios cada vez mais complexos dentro da plataforma.

Quando questionados com relação aos pontos negativos mencionados no primeiro encontro, os alunos chegaram a um consenso de que os esses problemas foram resolvidos pelas alterações feitas para a versão atualizada. Porém, quando questionados sobre possíveis novos pontos negativos, alguns alunos mencionaram que cada exercício poderia conter uma indicação de uma ferramenta ou aparato que pode ser utilizado para resolvê-lo (e.g. um link para a seção que fala sobre laços nos exercícios que utilizam dos mesmos). Além disso, foi mencionado que adicionar algumas coisas a mais sobre a sintaxe e as palavras chave de Javascript poderia ser útil (e.g. uma aluna mencionou que tinha esquecido do comando “return”).

Já nos pontos positivos, os alunos se mostraram gratos pela adição de detalhamento dos exercícios, especialmente pelos mesmos especificarem o que cada função deve retornar, algo que diminui a possibilidade de confusão.

4.1.3 O terceiro encontro

No terceiro e último encontro, os alunos utilizaram novamente uma versão atualizada da plataforma, com o objetivo de corrigir os problemas encontrados anteriormente. Durante o último encontro, a maioria dos alunos conseguiu chegar ao fim dos exercícios disponíveis, concluindo assim todo o conteúdo presente no Code Quest. Ao serem questionados sobre a nova versão apresentada da plataforma, os alunos disseram que as dificuldades levantadas no segundo encontro foram eliminadas com a nova versão, e ao serem perguntados sobre novos empecilhos, os mesmos disseram que não encontraram nenhum, resposta que atribuímos também em parte ao uso contínuo da plataforma.

4.2 Resultados do uso em Técnicas de Programação 1/2019.2

Após o terceiro encontro, foi realizado um questionário com o objetivo de obter feedback dos alunos com relação não somente à plataforma em si, mas à sua utilização junto à disciplina de Técnicas de Programação 1. O questionário se dividia em duas seções, uma quantitativa e outra qualitativa.

Na seção quantitativa, foram feitas três perguntas com o objetivo de obter uma medida sobre o grau de aceitação dos alunos em relação à utilização do Code Quest durante a disciplina. Foram feitas afirmações positivas sobre a plataforma, e os alunos deveriam indicar o nível de concordância com cada afirmação proposta de acordo com a Escala de Likert[2], a qual varia entre os valores de 1 (concordo totalmente) a 5 (discordo totalmente), conforme exibido nas Figuras 09, 10 e 11 a seguir.

[2] https://pt.wikipedia.org/wiki/Escala_Likert

Figura 09. Primeira pergunta quantitativa do questionário

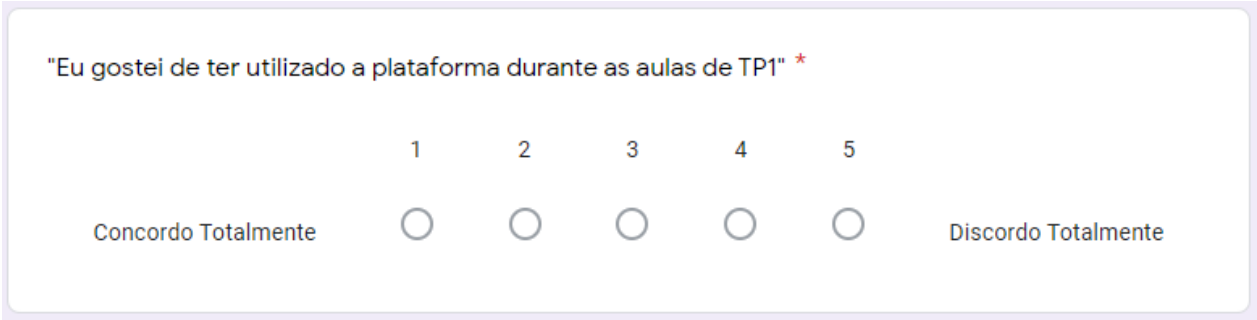


Figura 10. Segunda pergunta quantitativa do questionário

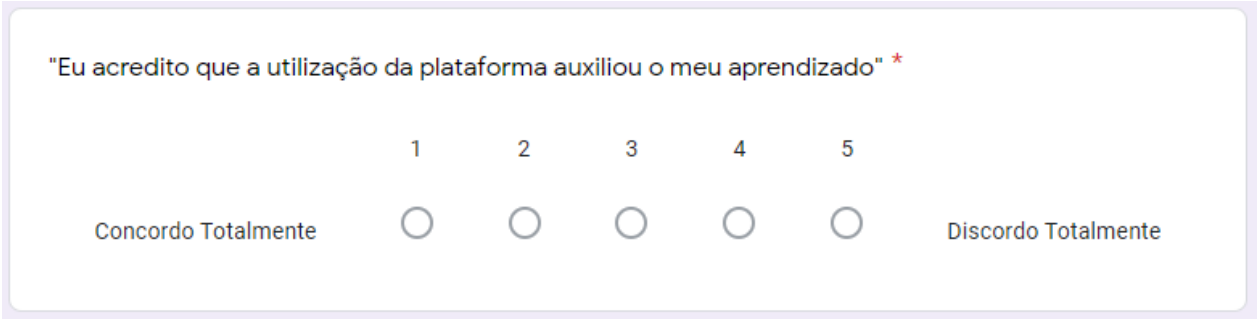
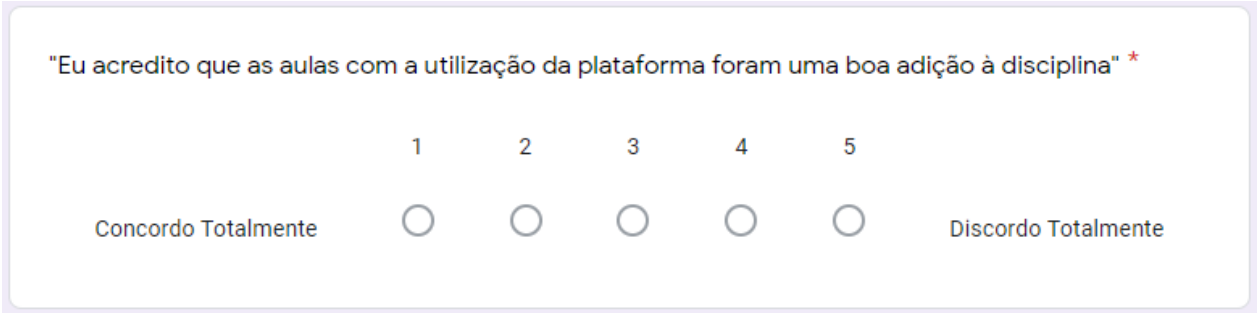
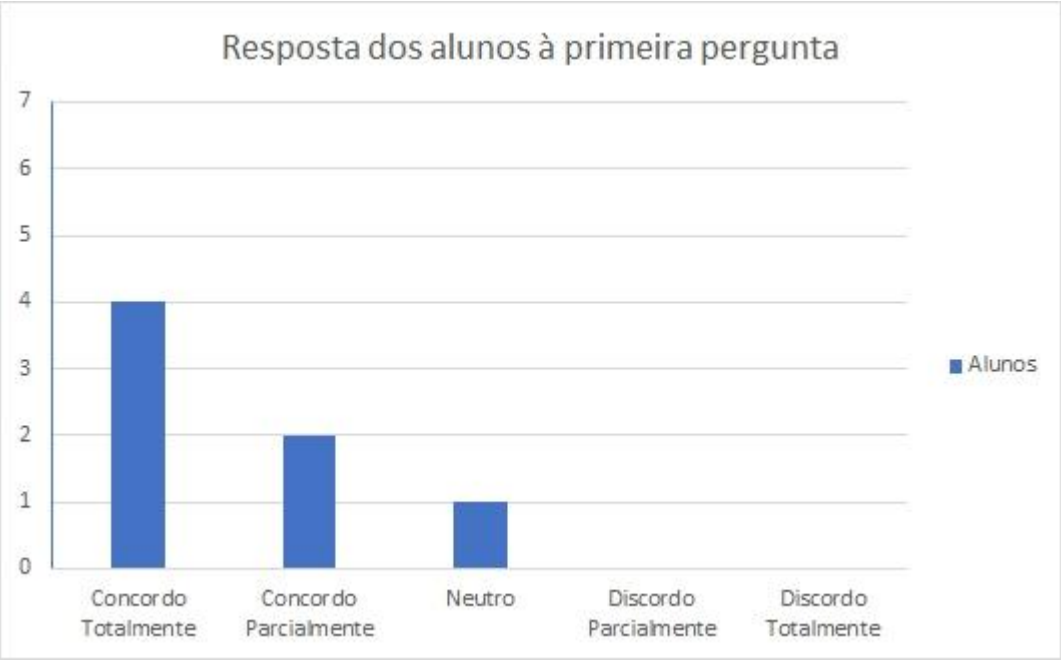


Figura 11. Terceira pergunta quantitativa do questionário



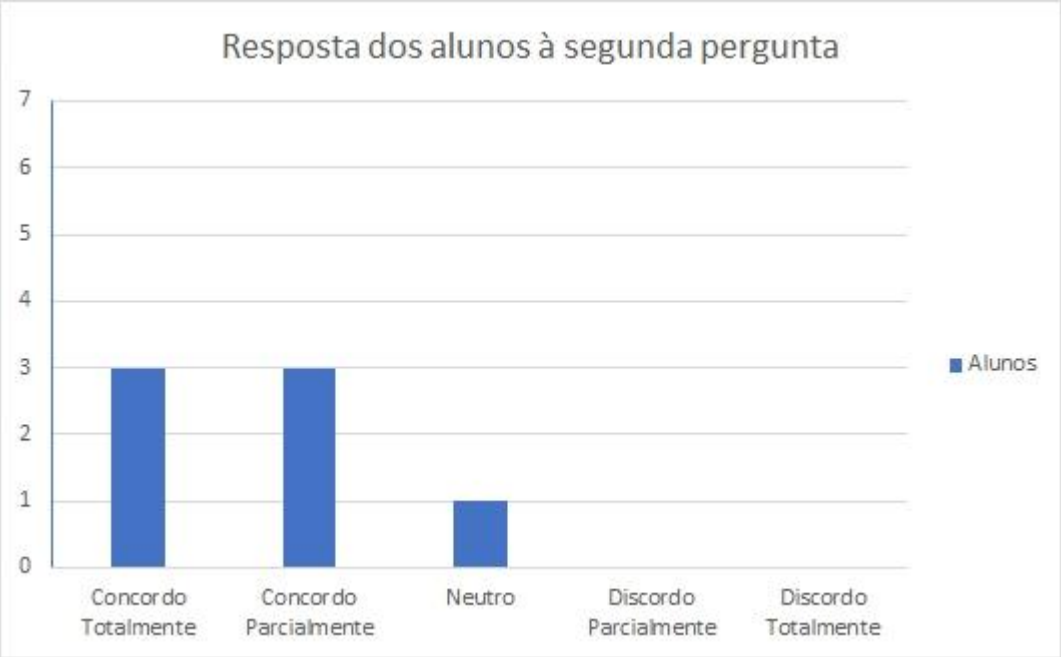
Ao observarmos as respostas enviadas pelos alunos, exibidas nas Figuras 12, 13 e 14 a seguir, podemos perceber que o grau de aceitação da plataforma foi, no geral, positivo.

Figura 12. Resposta dos alunos à pergunta "Eu gostei de ter utilizado a plataforma durante as aulas de TP1"



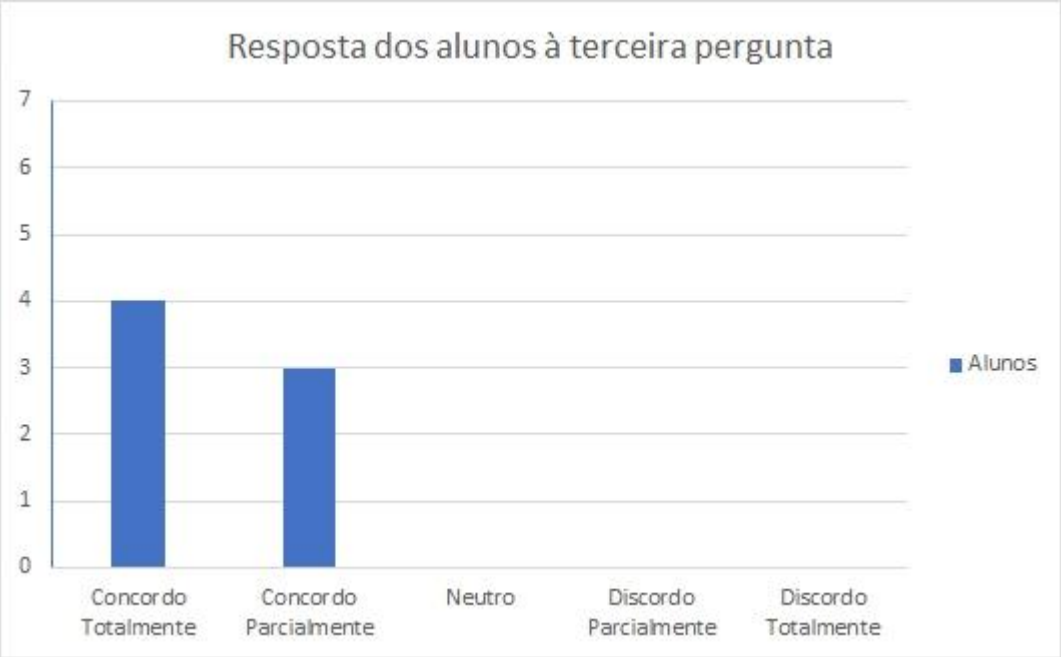
Com uma resposta majoritariamente positiva, podemos concluir que a experiência de usar o Code Quest foi prazerosa. Os alunos gostaram da plataforma e da sua usabilidade.

Figura 13. Resposta dos alunos à pergunta "Eu acredito que a utilização da plataforma auxiliou o meu aprendizado"



Aqui, a resposta novamente positiva nos diz que, além da satisfação dos alunos em utilizar a plataforma, os mesmos acreditam que sua utilização também se mostrou eficaz em auxiliar o seu processo de aprendizagem.

Figura 14. Resposta dos alunos à pergunta "Eu acredito que as aulas com a utilização da plataforma foram uma boa adição à disciplina"



Com a última pergunta quantitativa os alunos nos mostram que, além da plataforma isoladamente providenciar uma experiência positiva em seu processo de aprendizagem, ela também é uma adição benéfica à maneira como a disciplina de Técnicas de Programação 1 é ministrada.

O objetivo das perguntas realizadas era instigar o aluno a pensar sobre a sua experiência com a plataforma e com a disciplina em questão, de como a plataforma afetou o seu processo de aprendizagem, seja de maneira positiva ou negativa. Assim, podemos observar, pelos resultados, que as conclusões dos alunos foram, no geral, positivas, o que nos leva a concluir, como decorrido acima, que esse tipo de sistema pode sim ter um impacto benéfico no processo de ensino-aprendizagem de programação.

A segunda seção do questionário continha algumas perguntas opcionais com relação aos pontos positivos e negativos que os alunos tinham encontrado durante a utilização da plataforma, assim como um espaço para deixar sugestões ou ideias para melhorá-la. As respostas coletadas estão apresentadas nas Figuras 15, 16 e 17 a seguir.

Figura 15. Resposta dos alunos à primeira pergunta qualitativa do questionário

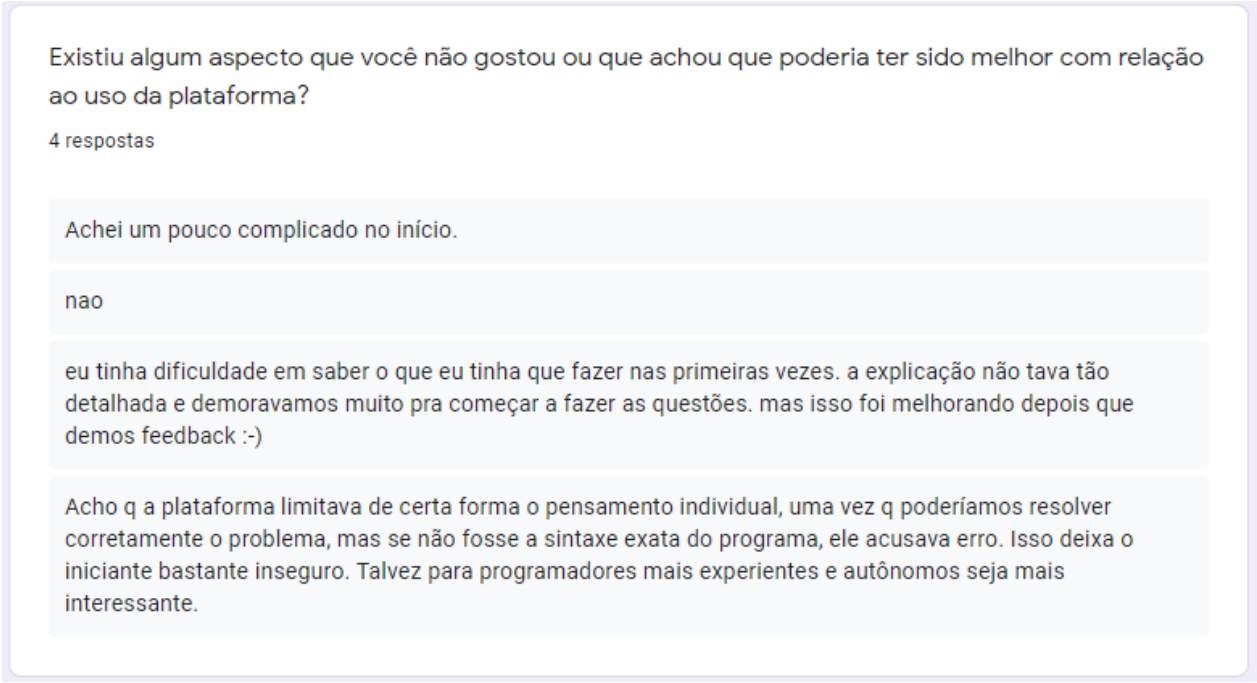


Figura 16. Resposta dos alunos à segunda pergunta qualitativa do questionário

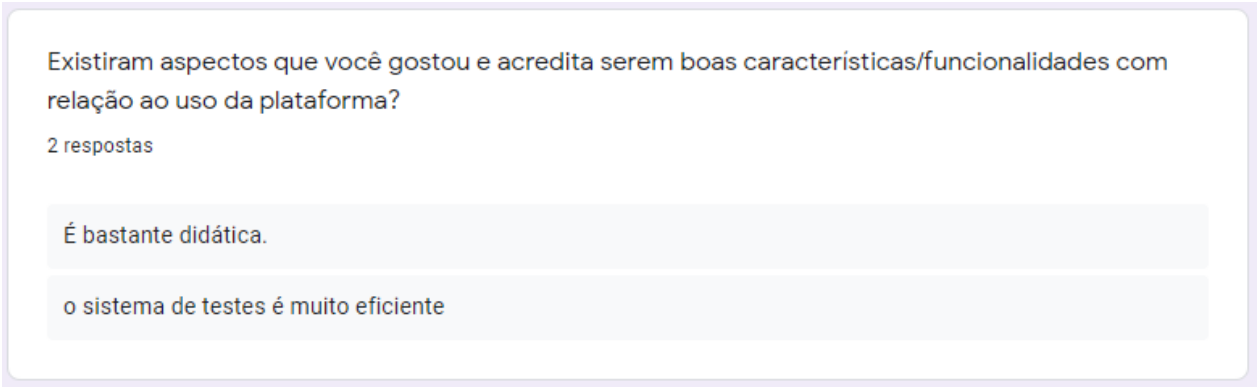
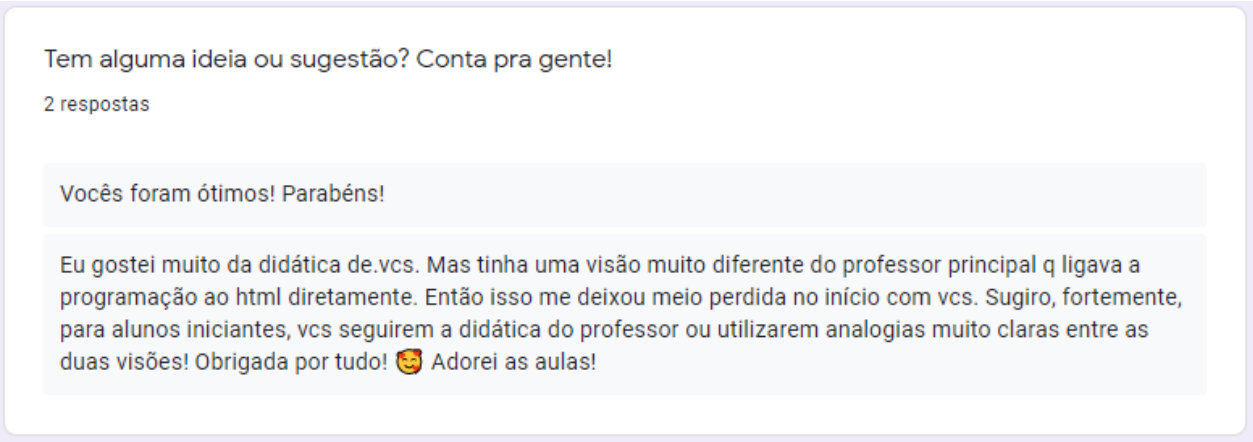


Figura 17. Resposta dos alunos à terceira pergunta qualitativa do questionário



Com relação aos pontos negativos citados na imagem 15, reconhecemos o fator da complexidade apontada pelos alunos nos contatos iniciais com a plataforma, porém com o feedback obtido em cada encontro, fomos implementando algumas das modificações sugeridas, e acreditamos que o fator de complexidade tenha sido reduzido ao longo do estudo de caso.

Quanto a declaração de que "a plataforma limitava, de certa forma, o pensamento individual", é uma conclusão baseada numa interpretação equivocada sobre o funcionamento da plataforma, pois ela não corrige de acordo com uma “sintaxe exata do programa”, conforme suposto pelo respondente, mas sim pelos casos de teste. O Code Quest não verifica o código ou a sintaxe, não impõe uma única forma de codificação, por isso consideramos que o sistema lida de maneira aceitável com o pensamento/resolução individual, dado que como a plataforma utiliza de casos de teste para avaliar a saída dos algoritmos codificados, e não o algoritmo em si, os alunos têm a liberdade de implementarem o código da maneira que acharem melhor. Com relação aos aspectos positivos citados, ficamos contentes ao saber que o sistema de testes com feedback automatizado teve o impacto positivo desejado, e também acreditamos que a seção conceitual da plataforma foi um dos fatores que a tornou mais didática.

Por fim, ao analisar as sugestões coletadas, podemos considerar que, embora javascript tenha potencial para ser uma ótima linguagem introdutória, incorporar conceitos de javascript

atrelado ao desenvolvimento web, como html por exemplo, pode ser prejudicial para o aprendizado de lógica de programação.

5. Refinamento

Com base nos resultados obtidos durante as aplicações da plataforma, trabalhamos em refinar a experiência de uso para incorporar os feedbacks recebidos ao longo do tempo. Este foi um trabalho que ocorreu em múltiplos momentos, itens mais facilmente implementáveis foram desenvolvidos entre cada um dos encontros, enquanto que aqueles que demandavam um trabalho técnico mais aprofundado foram realizados posteriormente.

5.1 Textos de referência e explicações

Conforme ouvimos dos alunos que sentiam falta de uma consulta pra sintaxe, devido à pouca experiência que tinham usando a linguagem JavaScript, decidimos por incluir uma seção que precedesse os exercícios com as principais estruturas da linguagem (declaração de variáveis, condicionais, iterações, etc) que pudessem ser facilmente consultadas em caso de dúvida. A Figura 18 a seguir ilustra um dos textos de referência adicionados.

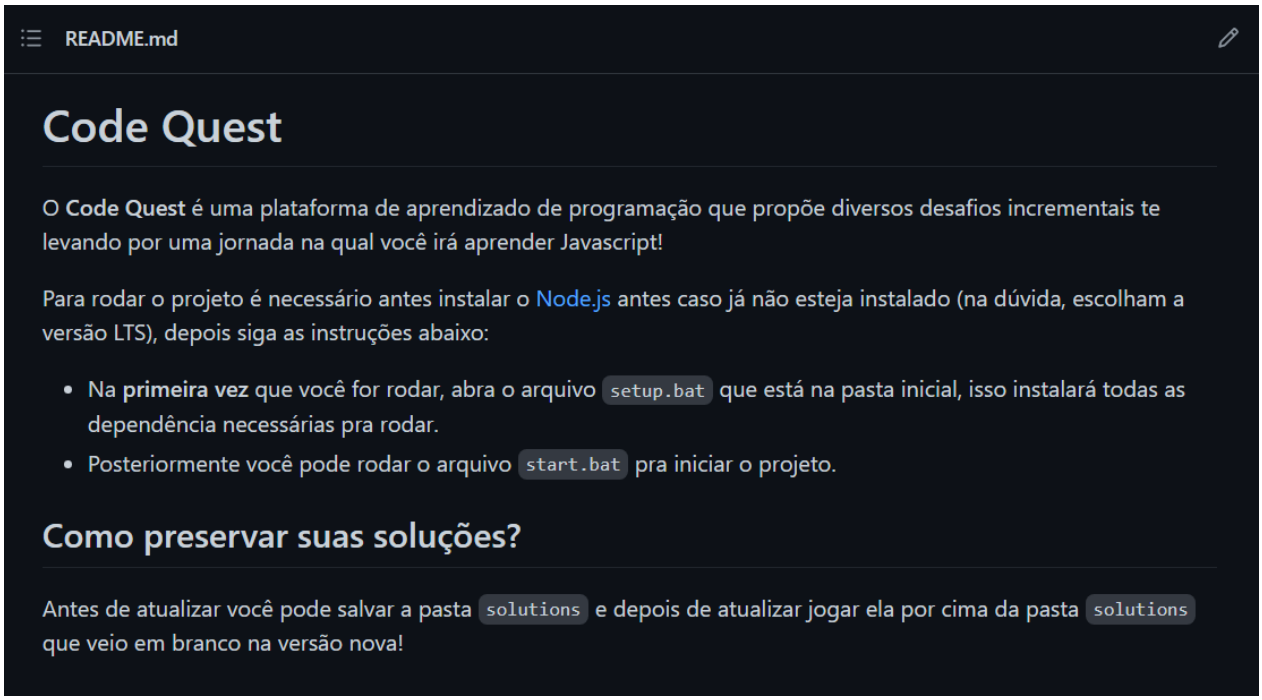
Figura 18. Textos de referência

Além disso, fizemos um trabalho de refinar as explicações dos exercícios para tentar minimizar as dúvidas que os alunos encontraram na sala de aulas.

5.2 Facilidade de uso

Outro problema relatado foi que começar a utilizar a plataforma se provou desafiador por exigir dos alunos que executassem comandos no console que desconheciam. Para sanar esse ponto de fricção, incluímos executáveis que cuidam dessa parte sem que o aluno precise saber o que está acontecendo, além de instruções no readme do projeto que explicam como iniciá-lo, como pode ser visto na Figura 19 a seguir.

Figura 19. Readme do projeto



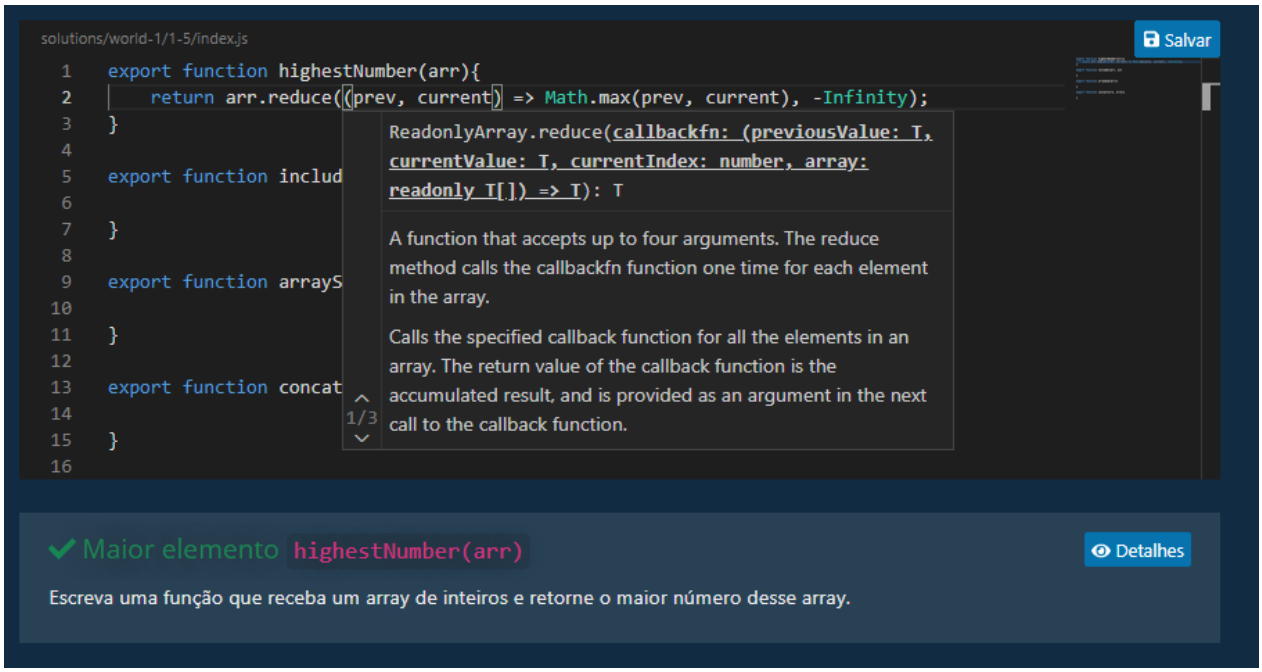
Posteriormente, também trocamos o servidor utilizado do Webpack para o Vite, este sendo muito mais rápido em inicializar a plataforma, reduzindo ainda mais a fricção para começar a usar a plataforma.

5.3 Editor de código

Uma das últimas edições foi a inclusão de um editor de código integrado nas páginas de exercício, também com a intenção de reduzir mais ainda a fricção externa ao permitir que o aluno programe sem ter que usar um editor de código externo.

Para alcançar esse objetivo, usamos a biblioteca aberta Monaco, que é o editor de código criado pela Microsoft para o VS Code. Ele é bastante completo, incluindo a documentação para várias funcionalidades nativas do javascript e a capacidade de sugerir opções baseado no seu código. A Figura 20 a seguir mostra o editor de código inserido na plataforma.

Figura 20. Editor de código



6. Conclusões

A seguir serão apresentadas as principais conclusões e considerações obtidas ao longo do processo de desenvolvimento e construção deste trabalho.

6.1 Contribuições

Quando tomamos a decisão de desenvolver o Code Quest, tínhamos ciência de que não estávamos em condição de produzir uma solução robusta. Ao construir a plataforma, nosso objetivo era mais humilde: acreditávamos em construir uma ideia, e para isso construímos uma ferramenta, não porque não existissem outras ferramentas com propósitos ou funcionalidade similares, mas porque acreditávamos que o processo de desenvolvimento da plataforma nos auxiliaria no processo de desenvolvimento dessa ideia.

Assim, mais do que qualquer outra coisa, o que desejávamos aqui era testar uma alternativa para sair dos confins tradicionais da pedagogia da nossa área e investigar se seria possível alcançar êxito na tarefa de ensinar as próximas gerações de desenvolvedores de maneira não expositiva — e conseguimos.

A plataforma que desenvolvemos se mostrou útil para ensinar programação para alunos iniciantes de maneira engajante, a abordagem proposta de “Faço antes, peço explicação depois (se precisar)" funcionou e mostrou-se desejável pelos estudantes.

6.2 Limitações

Nosso trabalho contou com limitações internas e externas, sejam os limites da nossa própria força de trabalho ou as limitações impostas pela pandemia do Coronavírus (COVID-19). Acreditamos que, dada uma disponibilidade maior de recursos (como tempo e esforço), seria possível alcançar resultados ainda mais satisfatórios.

Com o objetivo de preservar a nossa chance de êxito, mantivemos nosso escopo relativamente comprimido: focamos no ensino dos conceitos básicos de programação para alunos que estavam começando essa jornada deliberadamente. É razoável cogitar que, para ensinar outros tópicos (como outras linguagens), seriam necessárias adaptações ou mesmo novos desenvolvimentos na plataforma que produzimos.

Apesar dessas limitações, acreditamos que o trabalho produzido pode fazer parte de um corpo mais amplo de investigação que nos leve para um futuro em que o ensino acompanhe a revolução cibercultural que vivemos, se beneficiando das oportunidades que se abrem com ela.

6.3 Trabalhos futuros

Começando pelo mais simples, acreditamos que seria proveitoso investigar outros tópicos que se adequam bem ao modelo de ensino-aprendizagem que propusemos. Bons candidatos incluem disciplinas como (mas não se restringem a): Desenvolvimento de páginas web, Estruturas de dados, Algoritmos, Bancos de dados, Programação orientada a objetos, Jogos digitais entre outras. Explorar o ensino desses tópicos certamente adicionaria novos fatores que devem ser considerados na abordagem proposta.

Indo um pouco mais além, consideramos também que o trabalho evidenciou indiretamente o potencial de ensino-aprendizagem horizontalizado, onde os pares colaboram entre si no processo do aprendizado ao invés de ter no professor o único nó focal. Futuras investigações dedicadas a explorar esse tipo de abordagem nos parecem um campo fértil que poderia trazer uma miríade de benefícios, desde a possibilidade de melhorar a performance acadêmica do corpo docente, até consequências menos óbvias, como diminuir a evasão em cursos superiores ao fomentar laços sociais mais profundos.

Referências

PIMENTEL, Mariano; CARVALHO, Felipe: Máquinas de ensinar, 2021. Disponível em: <http://horizontes.sbc.org.br/index.php/2021/07/maquinas-de-ensinar/>. Acesso em: 28 de fev. de 2022.

MORAES, Maria C.: Informática educativa no Brasil: uma história vivida, algumas lições aprendidas. Revista Brasileira de Informática na Educação (RBIE), v. 1, n. 1, 1997, p. 4.

WATTERS. Audrey: The First Teaching Machines, 2015. Disponível em: <https://medium.com/the-history-of-the-future-of-education/the-first-teaching-machines-9ac9c4a1fa93>. Acesso em: 28 de fev. de 2022.

BENJAMIN, Ludy T.: A history of teaching machines. American Psychologist, 1988. 43(9), 703–712. https://doi.org/10.1037/0003-066X.43.9.703

ALEXANDER, Bruce K.: Addiction: The View from Rat Park, 2010. Disponível em: <https://brucekalexander.com/articlesspeeches/rat-park/148-addictionthe-view-from-rat-park>. Acesso em: 28 de fev. de 2022.

FREITAS, Luiz Carlos: Neotecnicismo digital. Avaliação Educacional, 2021. Disponível em: <https://avaliacaoeducacional.com/2021/07/11/neotecnicismo-digital/>. Acesso em: 28 de fev. de 2022.

GAROFALO, Débora: Como as metodologias ativas favorecem o aprendizado, 2018. Disponível em: <https://novaescola.org.br/conteudo/11897/como-as-metodologias-ativas-favorecem-o-aprendizado>. Acesso em: 28 de fev. de 2022.

BOSSE, Yorah; GEROSA, Marco Aurélio. Reprovações e trancamentos nas disciplinas de introdução à programação da Universidade de São Paulo: um estudo preliminar. In: Anais do XXIII Workshop sobre Educação em Computação. SBC, 2015. p. 426-435.

Moreira, Mireille & Favero, Eloi. (2009). Um Ambiente Para Ensino De Programação Com Feedback Automático De Exercícios. 66075-110.

Rocha, Paulo & Ferreira, Benedito & Monteiro, Dionne & Nunes, Danielle & Góes, Hugo. (2010). Ensino e Aprendizagem de Programação: Análise da Aplicação de Proposta Metodológica Baseada no Sistema Personalizado de Ensino. RENOTE. 8. 10.22456/1679-1916.18061.

Carlos, José & Júnior, Rocha & Rapkiewicz, Clevi & Antonio, José & xexéo, jose xexeo & Delgado, Carla. (2006). AVEP – Um Ambiente de Apoio ao Ensino de Algoritmos e Programação.

HOLANDA, Wallace Duarte; FREIRE, Laís de Paiva; COUTINHO, Jarbele Cássia da Silva. Estratégias de ensino-aprendizagem de programação introdutória no ensino superior: uma Revisão Sistemática da Literatura. RENOTE, v. 17, n. 1, p. 527-536, 2019.

Warren, Peter. (2004). Learning to Program: Spreadsheets, Scripting and HCI.. 327-333.