UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

KLEBERSON MATHEUS CUNHA SILVA CANUTO

PROGRAMMING COURSES: PLATAFORMA DE APOIO AO ENSINO DE PROGRAMAÇÃO

CAMPINA GRANDE - PB

2022

KLEBERSON MATHEUS CUNHA SILVA CANUTO

PROGRAMMING COURSES: PLATAFORMA DE APOIO AO ENSINO DE PROGRAMAÇÃO

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientadora: Professora Dra. Eliane Cristina Araújo

CAMPINA GRANDE - PB

2022

KLEBERSON MATHEUS CUNHA SILVA CANUTO

PROGRAMMING COURSES: PLATAFORMA DE APOIO AO ENSINO DE PROGRAMAÇÃO

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professora Dra. Eliane Cristina Araújo

Orientador – UASC/CEEI/UFCG

Professor Dr. Maxwell Guimarães de Oliveira

Examinador – UASC/CEEI/UFCG

Professor Dr. Tiago Lima Massoni

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 30 de Março de 2022.

CAMPINA GRANDE - PB

ABSTRACT

Learning logic and programming can be essential to a good start in an undergraduate computing course. Currently, learning programming has encouraged people with different interests, whether in the technological area professionally, or just out of curiosity and the desire to acquire new knowledge. There are already platforms that seek to support the teaching of programming, but mostly with little study material, limited availability, without practical tests, among other problems. Considering these problems, we proposed the Programming Courses platform, which seeks to help and encourage, in an interactive way, the learning of introductory programming. It allows the creation of Python programming courses, using different resources. Among them, we can mention: modules of teaching materials, multiple choice questions, recommendations for complementary materials and automatic tests, which also has the help of an oracle to validate inputs and outputs proposed by the student. In addition to these, there are gamification features, already available on most platforms, such as scoring, rewards and tips.

Programming Courses: Plataforma de apoio ao ensino de programação

Kleberson Matheus Cunha Silva Canuto
Universidade Federal de Campina Grande
kleberson.canuto@ccc.ufcg.edu.br

Eliane Cristina de Araújo
Universidade Federal de Campina Grande
eliane@computacao.ufcg.edu.br

RESUMO

Aprender lógica e programação pode ser essencial para começar bem um curso de graduação em computação. Atualmente, o aprendizado de programação tem incentivado pessoas com distintos interesses, seja na área tecnológica profissionalmente, ou apenas por curiosidade e vontade de adquirir novos conhecimentos. Já existem plataformas que buscam apoiar o ensino de programação, porém em sua maioria com pouco material de estudo, disponibilidade limitada, sem testes práticos, entre outros problemas. Tendo em vista estas lacunas, propusemos a plataforma Programming Courses, que busca auxiliar e incentivar, de forma interativa, o aprendizado de programação introdutória. Ele permite que sejam criados cursos de programação em Python [1], utilizando diferentes recursos. Dentre eles, podemos citar: módulos de materiais didáticos, questões de múltipla escolha, recomendações de materiais complementares e testes automáticos, que também conta com o auxílio de um oráculo para validar entradas e saídas propostas pelo aluno. Além destas, há funcionalidades de gamificação, já disponíveis na maioria das plataformas, como pontuação, recompensas e dicas.

Palavras-chave

Ensino de programação, cursos, testes automáticos, ferramentas de apoio ao ensino, oráculo, Python, programação introdutória.

Link para o repositório

https://github.com/KlebersonCanuto/programming_courses

1. INTRODUÇÃO

Plataformas de apoio ao ensino de programação e lógica podem ser de grande ajuda, principalmente para quem tem interesse acadêmico na área de computação. Existem disponíveis várias plataformas com este propósito, porém identificamos nelas algumas limitações. As mais comuns são a falta de vídeos para auxiliar o aprendizado, a disponibilidade limitada de acesso a partir de dispositivos móveis e a falta de testes automáticos através de um juiz de software para validar o conhecimento do aluno.

Considerando a baixa quantidade de aplicativos completos e o desejo de incentivar o ensino de programação introdutória, tanto para quem quer seguir carreira na área quanto para quem deseja, por curiosidade, aprender a programar, seja pelo uso diário de tecnologias ou pelo conhecimento, foi idealizada a plataforma Programming Courses, que busca, além de preencher as lacunas deixadas por outras plataformas, manter pontos positivos que estão presentes, como pontuação e dicas, e oferecer mais funcionalidades para auxiliar no aprendizado do aluno.

O Programming Courses conta com uma área livre para programar, depuração do código via Python Tutor [2] e um sistema de oráculo, onde o aluno pode validar entradas e saídas e até ser recompensado com pontos em alguns casos em que as entradas e as saídas estão corretas.

Essas funcionalidades adicionais, principalmente o oráculo, foram idealizadas partindo da suposição de que oferecer auxílios para um aluno entender melhor um problema, seus casos de borda e testar curiosidades relacionadas ao problema ou a linguagem em geral, com o passo a passo da execução do código a nível de variáveis, podem ajudar um aluno a concluir um problema e melhorar a visão geral dele como programador.

Com isso, o Programming Courses se propõe a ser uma ferramenta de suporte ao ensino de programação Python, tanto para pessoas que não estão vinculadas a algum curso, e querem aprender de forma independente, quanto para alunos de alguma disciplina Neste último caso, professores podem cadastrar seus cursos e incluir o material que desejem.

2. TRABALHOS RELACIONADOS

Foram analisadas mais a fundo 4 outras plataformas de auxílio de aprendizado em computação, sendo elas o Sololearn [3], o Mimo [4], o Programming Hero [5] e o Programming Hub [6]. Todos possuem muitos Downloads e são bem avaliados em lojas de aplicativos, com nota superior a 4.5.

Foi observado que cada um tem seus pontos positivos e negativos. O Programming Hero é focado em ser um jogo e possui uma área de desenvolvimento, porém não possui juiz de software para serem resolvidos pelo aluno e está disponível para dispositivos móveis, assim o Mimo, que também possui essas limitações, além de não possuir vídeos como material de ensino e área de desenvolvimento, e conta a seu favor a facilidade de uso, agradável ao usuário.

Quanto às outras 2 plataformas, ambas são mais completas e podem ser acessadas tanto por dispositivos móveis quanto por navegadores web. O Programming Hub encontra como principais dificuldades a falta de um juiz de software e conteúdo limitado no nível gratuito, enquanto que o Sololearn apresenta como ponto negativo a falta de vídeos como conteúdo.

3. SOLUÇÃO

Buscando manter as principais funcionalidades das plataformas citadas e acrescentar mais algumas, como a depuração do código e o oráculo, o Programming Courses propõe ser um site para estudantes aprenderem os assuntos iniciais de programação, fornecendo conteúdos e formas de praticar, além de incentivar através de recompensa, na versão inicial conta apenas com a linguagem Python, não sendo necessário a instalação da mesma.

Mesmo com foco para assuntos iniciais, nada impede que sejam cadastrados cursos intermediários ou avançados, desde que não atinja alguma limitação do sistema, como limite de 3 segundos de execução, a importação de bibliotecas não-padrão do Python, etc.

Figura 1: Tela de problema

3.1 Visão Geral

O Programming Courses permite o cadastro de diferentes cursos por parte de um administrador, onde cada curso possui módulos, que por sua vez é um conjunto de materiais instrucionais sobre um tema ou assunto.

Um módulo possui materiais, que são conteúdos textuais ou em forma de vídeo sobre o assunto do módulo, questões, que são questões de múltipla escolha com opções de dica, e problemas (Figura 1), que são problemas de programação que devem ser resolvidos através da criação de código por parte do aluno.

Os problemas contam ainda com um oráculo, uma opção para o usuário validar entrada e saída antes de enviar o código da solução, esse código é julgado por um juiz de software, que verifica se as saídas a partir de um conjunto de entradas são iguais às do código apresentado como solução pelo administrador.

Para ter acesso ao conteúdo de um curso, é necessário que o aluno tenha se cadastrado na plataforma e realizado login, além disso, a cada progresso em um curso é dada uma pontuação ao aluno, na qual ele pode ver no ranking e comparar com outros alunos como forma de incentivo.

Apesar de ser necessário uma conta para utilizar grande parte do sistema, ainda é possível utilizar a plataforma sem fazer o cadastro. Neste caso, o acesso é limitado apenas à área de desenvolvimento livre (Figura 2). Esta área funciona como um emulador de IDE (ambiente de desenvolvimento integrado), onde é possível escrever e executar código em Python.

Na área de desenvolvimento, o usuário pode digitar as entradas, caso haja necessidade, e receber a saída da execução do seu código. Uma outra opção, ao invés de executar o código normalmente, é executá-lo com o Python Tutor, que é um serviço que permite a depuração do código, ou seja, é possível ver a execução do código passo a passo, acompanhando o valor das variáveis em memória e suas atualizações.

Figura 2: Tela de desenvolvimento livre

3.2 Arquitetura

A arquitetura escolhida foi a padrão de desenvolvimento web (Figura 3), com um cliente (Frontend) se comunicando com um servidor (Backend) via protocolo HTTP [7], e esse servidor persiste os dados em um Banco de Dados.

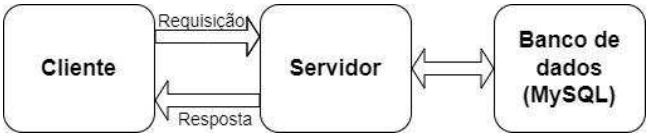

Figura 3: Fluxo básico do sistema

Essa arquitetura permite o isolamento das responsabilidades, de forma que o cliente precisa se preocupar com a parte visual e fazer as requisições para o servidor, que por sua vez faz as operações necessárias, mantendo a consistência da informação com o banco de dados.

3.2.1 Frontend

No lado do frontend, foi utilizado o framework React [8], que é uma biblioteca de Javascript que auxilia a criação de interfaces de usuário, em conjunto com o framework Bootstrap [9], uma biblioteca com foco em facilitar a estilização de componentes do frontend.

O frontend ainda faz integração com outros sistemas, como diversos reprodutores de vídeo, como Youtube, Facebook, Vimeo, entre outros, além do Python Tutor, que fornece um meio de redirecionar para o seu site com o código pré-preenchido.

Quanto a hospedagem, foi escolhido hospedar esse lado do sistema em um bucket do S3 (Simple Storage Service) [10] da AWS (Amazon Web Services), devido à escalabilidade, disponibilidade e baixa latência.

3.2.2 Backend

A estrutura do backend foi baseada na linguagem NodeJS, com o framework Express [11], que oferece flexibilidade e oferece diversos recursos para facilitar o desenvolvimento.

Para o banco de dados, foi escolhido um banco relacional, o MySQL [12], devido às relações entre os dados do sistema (Figura 4). Para fazer a ligação entre o servidor e o banco de dados, foi utilizado o ORM (Mapeamento Objeto-Relacional) Sequelize [13], que oferece facilidade para obter e inserir dados no banco de dados em formato de objeto.

Quanto à autenticação do usuário, foi utilizado o protocolo OAuth 2.0 [14], não fazendo necessário as credenciais serem expostas, sendo necessário apenas realizar login uma vez por sessão (definida nesse caso para 72 horas), e utilizando o JSON Web Token para as requisições. A senha do usuário é salva no banco de dados em forma de hash a partir do método bcrypt [15], não sendo possível ser revertida, apenas comparada, por ser uma função adaptativa, ou seja, o número de iterações pode aumentar para tornar o método mais lento e, consequentemente, resistente a ataques de força bruta.

Em termo armazenamento dos arquivos necessários (imagens e código da solução de um problema), foi utilizado o S3 da AWS, escolha essa devida à escalabilidade e disponibilidade.

Como ferramentas de auxílio ao desenvolvimento, foi utilizado o eslint [16] para análise do código, enquanto que para teste foram utilizadas as bibliotecas jest e supertest, a cobertura média foi de 97,04% das linhas de código.

A hospedagem do backend foi feita em uma instância do EC2 (Elastic Compute Cloud) [17], da AWS, que oferece segurança, escalabilidade e alta performance, o deploy foi facilitado pela ferramenta Docker Compose [18], que ajudou em toda a configuração do sistema na instância do EC2. Já o banco de dados foi hospedado no RDS (Relational Database Service) [19], também da AWS, facilitando a configuração e fornecendo escalabilidade, disponibilidade, resiliência, segurança e velocidade.

3.2.3 Adicionais

Além do Frontend e Backend e suas ferramentas, existe um outro componente que vale a pena ser citado, que é a parte responsável por executar um código em Python e retornar sua saída, esse componente é uma função sem servidor, também conhecida por serverless function, que nada mais é do que um trecho de código que é executado sem ser necessário a hospedagem via servidor.

Essa abordagem foi escolhida para manter a segurança e isolar a responsabilidade do servidor, que não precisa se preocupar com essa parte, além de não se fazer necessário a instalação de Python na máquina do servidor.

O serviço de nuvem escolhido foi o AWS Lambda [20], que fornece boas opções de configuração, como a quantidade de memória alocada, e uma boa escalabilidade.

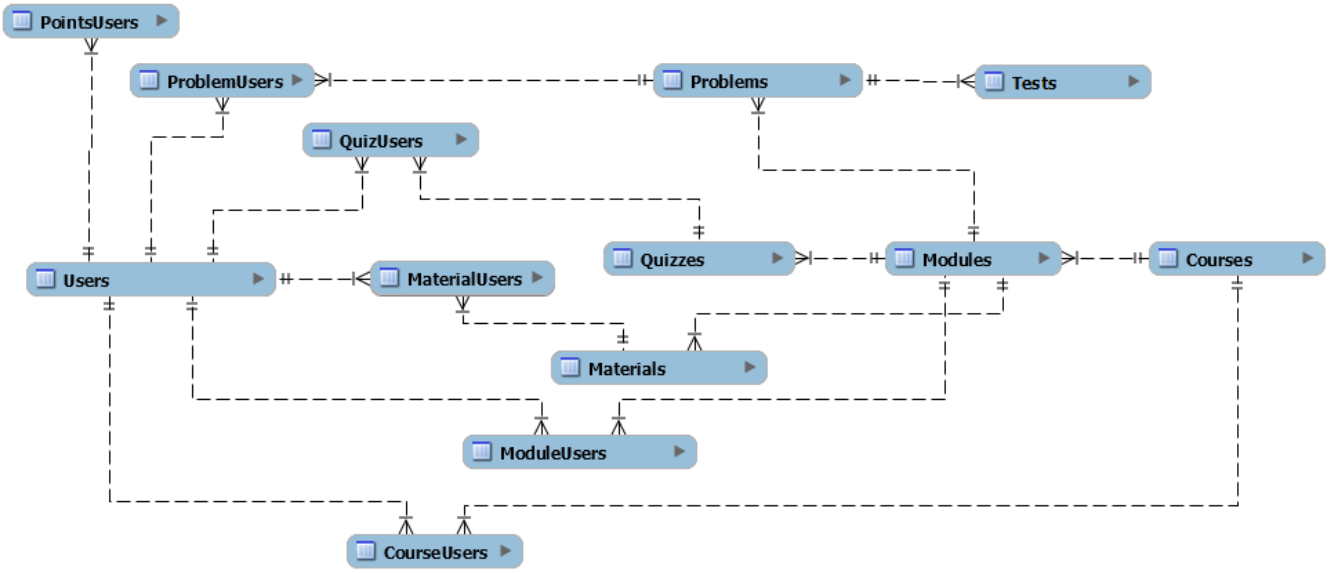

Figura 4: Relações do banco de dados

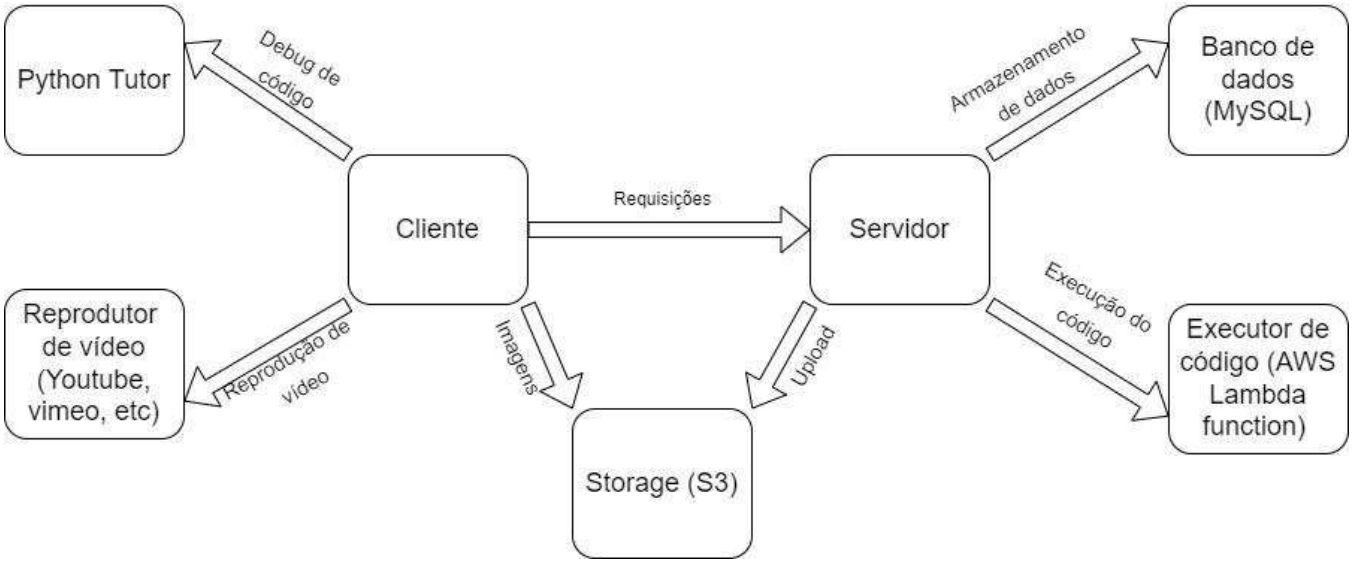

Figura 5: Arquitetura completa da comunicação do sistema

3.2.4 Sistema completo

A comunicação do sistema como um todo (Figura 5) é realizada de forma que cada componente tenha sua própria função.

É possível observar que o Cliente tem apenas a responsabilidade de comunicar-se com outros serviços, focando na visualização, o Servidor cuida da parte lógica, fazendo as operações necessárias e se comunicando com serviços de persistência de dados, mantendo os dados sensíveis seguros, os serviços de armazenamento de dados (Storage e Banco de dados) apenas mantêm os dados e retorna-os quando solicitado.

Além disso, o código do usuário é executado em um ambiente totalmente isolado, evitando ataques ao sistema e aos dados sigilosos, tendo em vista que o usuário é livre para escrever o código que desejar. Tendo em vista essa divisão de responsabilidades, nenhum dos serviços fica sobrecarregado e o entendimento do sistema e futuras melhorias no sistema.

4. AVALIAÇÃO

Com o sistema concluído, foram cadastrados materiais, questões e problemas de programação para a realização de um experimento de avaliação do Programming Courses (PC).

Foram cadastrados dois cursos com propósitos distintos. O primeiro, o curso de boas vindas, teve como objetivo auxiliar o usuário a entender o sistema e aprender a dar os primeiros passos na plataforma. O segundo, de programação Python introdutória, abordou assuntos de Entradas e Saídas, Tipos, Condicionais, Listas, Laços e Funções.

Foi criado um plano para a avaliação da plataforma e um roteiro para a realização do experimento. O plano de avaliação é um documento para explicar ao usuário como funciona o sistema e o que é esperado que seja testado por ele, enquanto que o roteiro do experimento diz respeito a como será a aplicação do experimento e quais os dados a serem obtidos, dos quais foram elencados dados relacionados a problemas e a pontuação do usuário.

Após essa etapa, foi feito o teste do sistema com usuários para obter feedbacks. Para isso, foi disponibilizado o plano de avaliação e um formulário para verificar o grau de satisfação dos alunos com o sistema e receber feedbacks abertos sobre quais foram os pontos positivos e negativos na visão dos usuários, além de sugestões pro sistema.

4.1 Resultados obtidos

O experimento de avaliação foi realizado em uma turma de programação introdutória da UFCG, com alunos provenientes de diversos cursos de graduação, tais como: Engenharia Mecânica, Engenharia Civil, bacharelado em Física, bacharelado em Matemática, dentre outros.

Ao todo, 17 pessoas utilizaram o sistema, das quais 13 responderam o formulário. No formulário, obteve-se uma média de satisfação geral de 4,15, em uma escala de likert de 1-5 (Figura 6), sendo 5 a melhor nota, e foi observado que o oráculo foi útil para 69,2% (Figura 7) dos usuários, além de ter recebido feedbacks sobre o sistema.

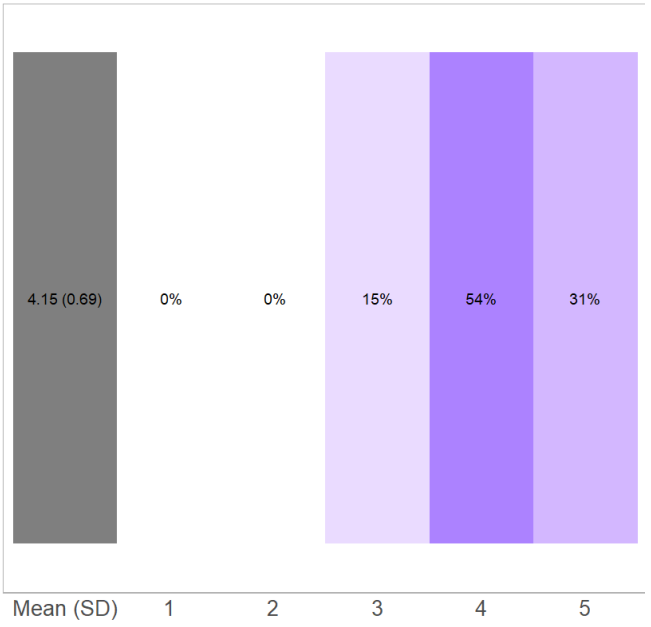

Figura 6: Satisfação geral dos usuários

Figura 7: Utilidade do oráculo.

Conforme mostrado nas tabelas 1 e 2, com relação aos Problemas, apenas 2 usuários não fizeram tentativas de resolução. Os outros 15, concluíram ao menos um problema, dos quais 12 deles concluíram de 1 a 3 problemas no total, e 3 deles concluíram ao menos 4 (Tabela 1). Ao todo 29 problemas (não necessariamente distintos) foram resolvidos, dos quais 25 foram resolvidos na primeira tentativa, 3 precisaram de 2 tentativas e em 1 caso se fez necessário 3 tentativas (Tabela 2).

Os alunos receberam pontuação de acordo com o progresso, ao ver materiais didáticos e resolver questões e problemas, além de receberem bônus por concluir algum módulo ou cursos, sendo as pontuações de problemas mais importantes, fornecendo ao usuário 10 pontos no rank.

De acordo com o roteiro de avaliação, era esperado uma pontuação mínima 53 pontos, caso o aluno concluísse os módulos recomendados, e uma máxima de 297 pontos, por completar todos os cursos. Conforme pode ser observado na Tabela 3, dos 17 alunos que utilizaram o sistema, 10 atingiram a meta, dos quais 3 conseguiram pontuar pelo menos o dobro da meta (106 pontos).

Problemas resolvidos	Usuários
0	2
1	9
2	3
3	0
4	0
5	0
6	1

Tabela 1: Distribuição da quantidade de problemas resolvidos.

Tentativas até o sucesso	Problemas
1	25
2	3
3	1

Tabela 2: Distribuição das tentativas de resolução de problemas.

Pontuação	Usuários
Menos da metade da meta (0-26)	4
Mais da metade da meta (27-52)	3
Atingiu a meta (53-106)	7
Ultrapassou o dobro da meta (107-153)	2
Mais da metade da pontuação máxima (154-297)	1

Tabela 3: Distribuição da pontuação dos usuários.

5. LIÇÕES APRENDIDAS

Nesta seção será detalhado como foi realizado o planejamento e suas etapas, quais dificuldades encontradas e como foram resolvidas, e quais são as perspectivas que podem ser vislumbradas para o futuro do Programming Courses.

5.1 Desenvolvimento

Primeiramente, foi levantado o tema e o escopo do projeto a partir da premissa de ser um site para oferecer cursos e possuir um juíz de software, após isso foram escolhidas as tecnologias a serem utilizadas, elas foram escolhidas com base no domínio do desenvolvedor e após visto que seria viável utilizar tais.

Em seguida foi feita a descrição das tarefas, junto aos requisitos e estimativas de tempo, priorizando tarefas com maior relevância para produto mínimo viável, ou MVP, e então prosseguir para o desenvolvimento do sistema.

O código do sistema foi armazenado no GitHub, que também ficou responsável pelo controle de versão. Após o desenvolvimento ser concluído, foram preparados testes e, após algumas correções, foi realizado o deploy, de forma que o Frontend está hospedado por um bucket no S3, o Backend em uma instância do EC2 e o Banco de Dados no RDS, todos da AWS.

5.2 Desafios

No desenvolvimento, houve algumas dificuldades técnicas, umas foram superadas e outras tiveram que ser contornadas.

Em relação ao upload de arquivos (imagem de problema e código oficial da solução), a princípio foi utilizado a API do Google Drive [24], onde localmente houve êxito, porém ao colocar em produção, houve algumas dificuldades, a primeira foi a necessidade de gerar novas chaves e transportá-las com segurança, o transporte foi feito através de download do arquivo via bucket do S3, porém outra dificuldade surgiu, que foi em relação a expiração das chaves, juntando esses problemas e uma relativa lentidão ao utilizar essa API, foi decidido migrar completamente para o S3, resolvendo esses problemas.

Já na área de execução do código houve alguns problemas, primeiramente uma dificuldade em encontrar formas de executar o código, após muitas pesquisas e tentativas, foi decidido utilizar um pacote chamado python-shell, que cumpre bem o proposto, porém

não era uma escolha muito agradável, já que poderia acabar expondo dados indesejados ou causar problemas ao sistema, então foi decidido fazer uma alteração e migrar para uma solução serverless, oferecendo um isolamento do sistema e segurança, além de algumas melhorias menores, como a redução de pacotes do sistema, não ser mais necessário ter o Python instalado no sistema principal.

Por fim, outro desafio encontrado foi na criação de um terminal interativo, não foi encontrado como fazer de forma completa, então esse desafio teve que ser contornado com a área de desenvolvimento livre, que já era planejada, porém não era previsto fornecer entradas, já que a princípio as entradas seriam via terminal.

5.3 Trabalhos futuros

Para trabalhos futuros, é considerado que existem pequenas e grandes melhorias que acrescentariam bastante ao sistema, a começar pelas funcionalidades que foram planejadas pro sistema, porém não pro MVP, o Fórum, um sistema de Login com Google e a possibilidade de utilizar outras linguagens de programação, além de tentar novamente a criação de um terminal interativo.

Outras possibilidades futuras são o envio de Email para confirmar contas, materiais com fotos, medir o tempo do aluno para resolver o problema, customizar a pontuação de problemas e questões, ajudar o aluno a ver qual foi o erro, entre outros.

Além disso, do ponto de vista administrativo, é almejado que em uma versão futura seja fornecido níveis de permissão, facilitando com que professores criem cursos sem necessariamente ter acesso a alterar outros cursos.

Outras melhorias também foram sugeridas pelos usuários, das mais relevantes, podemos citar melhorias no sistema de login, na especificação dos erros dados pelo sistema e melhorias visuais.

É visto uma boa possibilidade de expansão do sistema, o processo de melhoria é contínuo, portanto sempre poderá haver evolução, como as funcionalidades citadas e melhorias de usabilidade e visibilidade para agradar mais usuários.

6. AGRADECIMENTOS

Agradeço à minha família e amigos, em especial à minha esposa e meus pais, pelo apoio desde o início do curso, e principalmente nesta etapa final, agradeço também aos meus colegas de curso, professores e todos que fizeram parte dessa jornada e me ajudaram a chegar até aqui, um agradecimento especial ao aluno de mestrado André Almeida, pelo auxílio com a funcionalidade do oráculo, à professora Eliane Araújo, pela orientação durante todo o TCC e disponibilizou um horário da aula de ICC para aplicar o experimento, e a todos que contribuíram com a avaliação do sistema.

7. REFERÊNCIAS

[1] Linguagem de programação Python. https://www.python.org/.

[2] Python Tutor, visualizador de execução de código. https://pythontutor.com/

[3] Plataforma Sololearn. https://www.sololearn.com/home

[4] Plataforma Mimo. https://getmimo.com/

[5] Plataforma Programming Hero. https://www.programming-hero.com/

[6] Plataforma Programming Hub. https://programminghub.io/

[7] Visão geral sobre HTTP. https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Overview

[8] Framework de criação de interface de usuário React. https://pt-br.reactjs.org/

[9] Framework de estilização de componentes Bootstrap. https://getbootstrap.com.br/

[10] Serviço de armazenamento S3. https://aws.amazon.com/pt/s3/

[11] Framework de aplicativo web Express. https://expressjs.com/pt-br/

[12] Banco de dados Relacional MySQL. https://www.mysql.com/

[13] ORM Sequelize. https://sequelize.org/

[14] Protocolo OAuth 2.0. https://oauth.net/2/

[15] Introdução sobre Bcrypt https://medium.com/reprogramabr/uma-breve-introdu%C3%A7%C3%A3o-sobre-bcrypt-f2fad91a7420

[16] Ferramenta de análise de código eslint. https://eslint.org/

[17] Serviço de computação em nuvem EC2. https://aws.amazon.com/pt/ec2/

[18] Ferramenta de gerenciamento de containers Docker Compose. https://docs.docker.com/compose/

[19] Serviço de banco de dados relacional RDS. https://aws.amazon.com/pt/rds/

[20] Serviço de funções sem servidor AWS Lambda. https://aws.amazon.com/pt/lambda/