Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

Pytuga Web: Uma ferramenta de apoio ao ensino de programação em português

Autor: Andrew Lucas Guedes de Souza
Orientador: Prof. Dr. Renato Coral Sampaio

Brasília, DF
2022

Andrew Lucas Guedes de Souza

Pytuga Web: Uma ferramenta de apoio ao ensino de programação em português

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Prof. Dr. Renato Coral Sampaio

Brasília, DF

2022

Andrew Lucas Guedes de Souza
Pytuga Web: Uma ferramenta de apoio ao ensino de programação em português/ Andrew Lucas Guedes de Souza. – Brasília, DF, 2022-
56 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Renato Coral Sampaio

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2022.

1. Pytuga Web. 2. Ferramenta para ensino de programação. I. Prof. Dr. Renato Coral Sampaio. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Pytuga Web: Uma ferramenta de apoio ao ensino de programação em português

CDU 02:141:005.6

Andrew Lucas Guedes de Souza

Pytuga Web: Uma ferramenta de apoio ao ensino de programação em português

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 03 de Outubro de 2022

Prof. Dr. Renato Coral Sampaio
Orientador

Profa. Dra. Carla Silva Rocha Aguiar
Convidado 1

Prof. Dr. Bruno César Ribas
Convidado 2

Brasília, DF
2022

Agradecimentos

Agradeço a toda minha família, que me apoiou e que sempre torceu por mim, em especial aos meus pais, Viviane e João Osório, que me apoiaram desde a minha escolha de curso, apesar de não saberem exatamente o que faz um engenheiro de software. E que mesmo eu não criando o tão solicitado robozinho, acreditaram em mim e me auxiliaram durante todo o curso. Agradeço por serem essas pessoas maravilhosas as quais eu tenho o privilégio de chamar de pai e mãe, amo muito vocês.

Às minhas maninhas, Júlia Gabriely e Ana Clara, que sempre foram meu ouvido amigo, escutando eu falando sobre as coisa legais que eu aprendia nas aulas, mesmo não fazendo ideia do que eu estava falando. E que apesar das nossas brigas, não consigo viver sem elas por perto, amo vocês demais, maninhas.

A todos os professores e amigos que tive o prazer de conhecer na FGA. Aos meus irmãos de coração, Gabriel Filipe e João Lucas, pelos conselhos, pela companhia e por todas nossas conversas filosóficas da madrugada que tivemos durante esses anos que passamos juntos, amo vocês.

Obrigado a todos.

Resumo

O ensino de programação é um processo desafiador que traz algumas dificuldades adicionais tanto para o professor quanto para o estudante. Além do processo de apreensão de novos conceitos e do desenvolvimento de técnicas e habilidades, a programação impõe desafios técnicos na configuração de ambientes, ferramentas e controle de versão. A dificuldade de configuração e a heterogeneidade dos ambientes de execução traz uma série de complicações tanto para alunos quanto para professores. Dessa forma, diversas ferramentas foram desenvolvidas com o objetivo de facilitar esse processo. Entre essas ferramentas se encontra o Pytuguês, uma versão em português da linguagem de programação Python que visa facilitar o aprendizado de estudantes falantes da língua portuguesa. Porém, o ambiente de desenvolvimento do Pytuguês possui dificuldades de distribuição devido a complexidade na instalação de pacotes e preparação de ambiente. Então com o intuito de facilitar a utilização do Pytuguês, este trabalho apresenta o desenvolvimento do Pytuga Web, uma ferramenta que permite a execução de código Pytuguês através de um navegador. Esta ferramenta possibilita ao usuário aprender os conceitos básicos de programação através da criação de desenhos gráficos utilizando código em português.

Palavras-chave: Pytuguês, ensino de programação, desenvolvimento web.

Abstract

Teaching programming is a challenging process that brings some additional difficulties for both the teacher and the student. In addition to the process of apprehending new concepts and developing techniques and skills, programming imposes technical challenges in configuring environments, tools and version control. The configuration difficulty and the heterogeneity of execution environments brings a series of complications for both students and teachers. Thus, several tools were developed with the objective of facilitating this process. Among these tools is Pytuguês, a Portuguese version of the Python programming language that aims to facilitate the learning of Portuguese-speaking students. However, the Pytuguês development environment has distribution difficulties due to the complexity of installing packages and preparing the environment. So in order to facilitate the use of Pytuguês, this work presents the development of Pytuga Web, a tool that allows the execution of Pytuguês code in a browser. This tool allows the user to learn the basic concepts of programming through the creation of graphic drawings using code in Portuguese.

Key-words: Pytuguês, programming teaching, web development.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

HTML	Linguagem de Marcação de Hipertexto (HyperText Markup Language)
CSS	Folhas de Estilo em Cascata (Cascading Style Sheets)
SVG	gráficos vetoriais escalonáveis (Scalable Vector Graphics)
API	Interface de programação de aplicações (Application Programming Interface)
Fig.	Figura

Sumário

Introdução

As disciplinas de introdução a algoritmos tem como objetivo ensinar os alunos a desenvolver algoritmos computacionais através da apresentação dos conceitos básicos de programação de computadores. Por ser geralmente o primeiro contato com os conceitos de programação, essas disciplinas possuem altos índices de reprovação nas instituições de ensino brasileiras. Por esse motivo essas disciplinas tornaram-se foco de diversos estudos que buscam identificar as principais dificuldades enfrentadas pelos alunos e propor melhorias nos métodos de ensino adotados nas salas de aula (JÚNIOR, 2004).

Dentre as dificuldades apontadas por esses estudos, a maior dificuldade dos alunos se encontra na compreensão dos conceitos básicos de estruturas de controle e como aplicá-las. A segunda dificuldade mais relatada pelos alunos recai no uso da linguagem de programação adotada pela disciplina. Ao passar por essas dificuldades os alunos acabam se desmotivando, prejudicando, assim, o seu aprendizado.

Um outro problema apontado pelos alunos é a barreira do idioma. As linguagens de programação mais utilizadas se baseiam no idioma inglês. Assim, o processo de aprender uma linguagem de programação se torna mais difícil para uma pessoa cujo o inglês não é o idioma principal (RUBY; KRSMANOVIC, 2017).

Neste contexto, diversas ferramentas surgiram com o intuito de facilitar o aprendizado da programação. Dentre elas tem-se o Pytuga, uma ferramenta que permite ao aluno escrever o código utilizando uma versão do Python em língua portuguesa chamada Pytuguês, que foi desenvolvida pelo professor Fábio Macedo Mendes da Universidade de Brasília, sendo sua principal motivação, eliminar a barreira do inglês no ensino da programação (PYTUGUêS, 2015).

Além da facilidade do idioma o *Pytuga* conta com uma integração com pacote *Turtle Graphics*, uma ferramenta bastante utilizada para introduzir conceitos de programação através da criação de desenhos gráficos com código. Além disso a transição para a linguagem Python ocorre de forma suave, já que é possível utilizá-la misturada com o Pytuguês.

Contudo, o Pytuga possui algumas falhas como ferramenta de ensino que dificultam o seu uso em sala de aula. Como por exemplo, no preparo do ambiente de desenvolvimento, o iniciante pode enfrentar problemas em relação as versões dos pré-requisitos de instalação da ferramenta. Um outro problema identificado é que o Pytuga faz uso de bibliotecas depreciadas, que dificultam a implementação de novos recursos.

Com a intenção de melhorar e facilitar o uso do Pytuga por educadores e es-

tudantes de programação, surgiu a proposta deste trabalho. Construir um ambiente de desenvolvimento acessível e que permita a utilização do Pytuguês e seus recursos.

Objetivos

Objetivo Geral

Este trabalho tem como objetivo desenvolver um ambiente *web* de desenvolvimento capaz de executar código em Pytuguês, visando facilitar o uso do Pytuguês no ensino de programação.

Objetivos Específicos

- Analisar as ferramentas de ensino de programação existentes;
- Possibilitar a execução de código Pytuguês no navegador;
- Criar uma ferramenta de uso simples;
- Realizar o *deploy* de uma versão estável da ferramenta;

Questão de pesquisa

É possível desenvolver uma ferramenta de ensino de programação capaz de executar Pytuguês e que seja acessada pelo navegador?

Planejamento do trabalho

O desenvolvimento deste trabalho foi dividido em duas entregas. Na primeira entrega, foi realizada uma revisão de ferramentas já existentes que possuem uma arquitetura similar que permita a validação da proposta deste trabalho, além da criação de um protótipo para validar a viabilidade técnica. Na segunda, foi realizado o desenvolvimento da aplicação e realizado o *deploy*, assim como a apresentação desse processo de desenvolvimento e resultados obtidos.

Organização do trabalho

Este trabalho está organizado da seguinte forma:

- O Capítulo 1 fornece informações sobre o contexto e justificativa deste trabalho.

- O Capítulo 2 traz informações sobre ferramentas de ensino de programação similares a ferramenta proposta neste trabalho.
- O Capítulo 3 apresenta as soluções escolhidas para realização deste trabalho.
- O Capítulo 4 descreve o processo de desenvolvimento da ferramenta.
- O Capítulo 5 trata dos resultados gerais do trabalho, onde é apresentada a ferramenta desenvolvida.
- O Capítulo 6 traz as considerações finais sobre este trabalho.

1 Ferramentas de ensino de programação

Aprender e ensinar programação são consideradas tarefas difíceis (BENNEDSEN; CASPERSEN, 2007). Por isso, muitos projetos de pesquisa são dedicados ao desenvolvimento de ferramentas que buscam auxiliar professores e alunos nesse processo de ensino. Entretanto, apenas um pequeno número dessas ferramentas são adotadas em sala de aula (PEARS et al., 2007).

Vários motivos limitam a adoção em grande escala dessas ferramentas, pois geralmente são desenvolvidas para resolver um desafio específico (PEARS et al., 2007). Ferramentas como o Visualg e o Calango não são indicadas para construção de aplicações para produção, e por causa disso possuem um ecossistema limitado ao ensino.

Observa-se também que a grande maioria das ferramentas foram desenvolvidas como protótipos durante um projeto de pesquisa, e são abandonadas após o fim do projeto. Assim, novas ferramentas são criadas ao invés de adotar ou evoluir as existentes. Criando assim, uma série de projetos com incompletos, limitados e com uma série de falhas não corrigidas.

Além do fato de que essas ferramentas utilizam linguagens próprias que são utilizadas apenas para fins educacionais, e por isso, muitos estudantes preferem iniciar diretamente com linguagens amplamente utilizadas no mercado de trabalho visando a empregabilidade imediata.

1.1 Análise de ferramentas de ensino

Com o objetivo de analisar diferentes abordagens aos problemas enfrentados pelos alunos e professores no ensino de programação, foi realizado um levantamento de ferramentas similares à proposta deste trabalho.

As ferramentas selecionadas podem ser divididas em ferramentas que utilizam pseudo linguagem em português, e ferramentas online. Primeiro serão apresentadas as ferramentas de ensino que utilizam uma linguagem de programação em português: VisuAlg, Calango e Pytuguês. Depois serão analisadas ferramentas de ensino de programação em ambiente online: Scratch, Replit e Basthon.

1.1.1 VisuAlg

O VisuAlg é um programa criado para ajudar alunos iniciantes em programação. Ele é utilizado para edição, interpretação e execução de algoritmos escritos em Portugol.

O Portugol é uma linguagem derivada do ALGOL, que permite escrever as instruções do algoritmos em língua portuguesa estruturado de forma simples e intuitiva.

Segundo o professor Antônio Carlos Nicolodi, o idealizador do VisuAlg, a dificuldade com inglês é bastante recorrente entre seus alunos nas aulas de lógica de programação. Com isso, ele desenvolveu uma linguagem em português para utilizar em suas aulas e a chamou de Portugol. Em seguida desenvolveu o VisuAlg, que é o editor e interpretador de código Portugol (ANTÔNIO, 2017).

O VisuAlg conta com uma série de recursos úteis para o estudo de programação. Como a exibição de uma lista de todas variáveis e seus valores atribuídos, e a possibilidade da execução passo a passo ou marcar algum ponto de parada durante a execução do código. Esse conjunto de funcionalidades facilita a compreensão acerca do funcionamento do algoritmo executado e, caso não esteja funcionando corretamente, facilita a depuração do código.

O professor Antônio afirma que esse método melhorou a aprendizagem de seus alunos e que o utiliza até hoje em suas aulas. A melhora foi tão significativa, que o VisuAlg foi adotado por diversos professores em diversas instituições de ensino brasileiras.

Mas apesar de todo esse sucesso, alguns fatores limitam a sua adoção. O VisuAlg possui suporte somente ao sistema operacional Microsoft Windows. Além disso, a sintaxe utilizada pelo Portugol, é semelhante à do Pascal, tornando mais difícil a transição do aluno para a sintaxe de linguagens mais utilizadas na indústria, como o C, C++, Java, ou Python.

Na Fig. 1, é possível verificar como se apresenta a interface do Visualg que pode ser dividida em 4 partes (editor de código; terminal de entrada e saída; quadro de variáveis; menu). No editor, verifica-se uma falta de recursos que estão presentes nos principais editores de código atuais, como sugestão de palavras e múltiplos cursores.

1.1.2 Calango

O Calango é uma ferramenta educacional multiplataforma projetada para facilitar a aprendizagem de programação e fornecer uma sintaxe simples que permite o desenvolvimento de lógica avançada de programação. Dessa forma os alunos podem se concentrar em aprender a lógica da programação sem se preocupar com os detalhes da linguagem (SILVA et al., 2020).

A sintaxe do Calango foi inspirada na linguagem C, pois o C é amplamente utilizado em disciplinas de introdução a programação. Por esse motivo, a utilização do Calango é mais indicado para cursos que posteriormente ensinarão o C, pois a transição será mais suave para o aluno (SILVA et al., 2020).

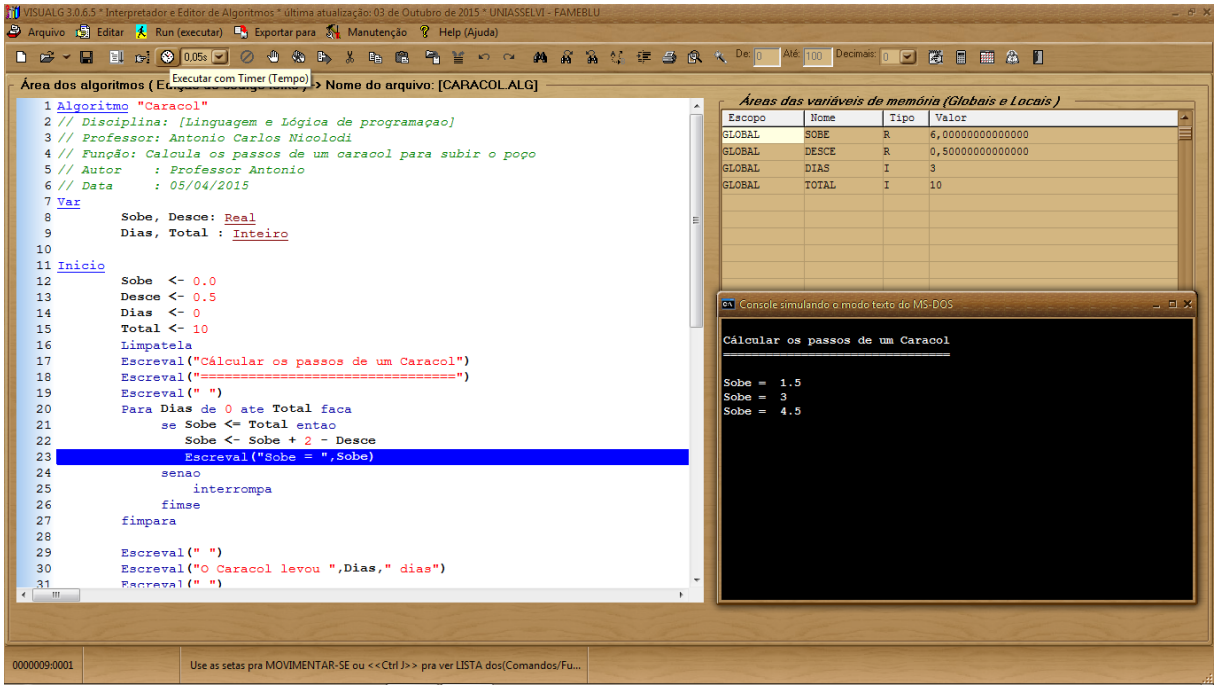

Figura 1 – Interface do VisuAlg

Apesar de possuir funcionalidades semelhantes ao VisuAlg, como execução passo-a-passo com acompanhamento das mudanças nos valores de variáveis, o Calango se diferencia do VisuAlg por ser multiplataforma e pode ser utilizado no Microsoft Windows, Linux e macOS.

Como pode-se observar na Figura 2, o Calango, opta por uma interface mais minimalista, e esconde funcionalidades menos utilizadas nos menus, exibindo-as somente quando necessário, deixando a tela simples e menos poluída.

Por ser inspirados em linguagens de tipagem estáticas, o Calango e o Portugol possuem essa mesma característica, ou seja, é necessário especificar tipos de dados para cada variável declarada. Apesar de tipos de dados serem um conhecimento essencial na programação de computadores, muitas linguagens modernas e amplamente utilizadas como Python e Javascript, possuem sistema de tipagem dinâmica, não sendo necessário explicitar o tipo das variáveis pois os interpretadores controlam o tipo das variáveis em tempo de execução.

1.1.3 Pytuguês

Em 2016, o professor Fábio Mendes (UnB) desenvolveu o Pytuguês para auxiliá-lo em suas aulas de introdução à programação. Essa ferramenta, além de eliminar a barreira do idioma, facilita o uso de pacotes utilizados no ensino da programção, como o *Turtle Graphics*, que visa tornar o aprendizado mais lúdico para os alunos através da introdução de elementos visuais (PYTUGUêS, 2015).

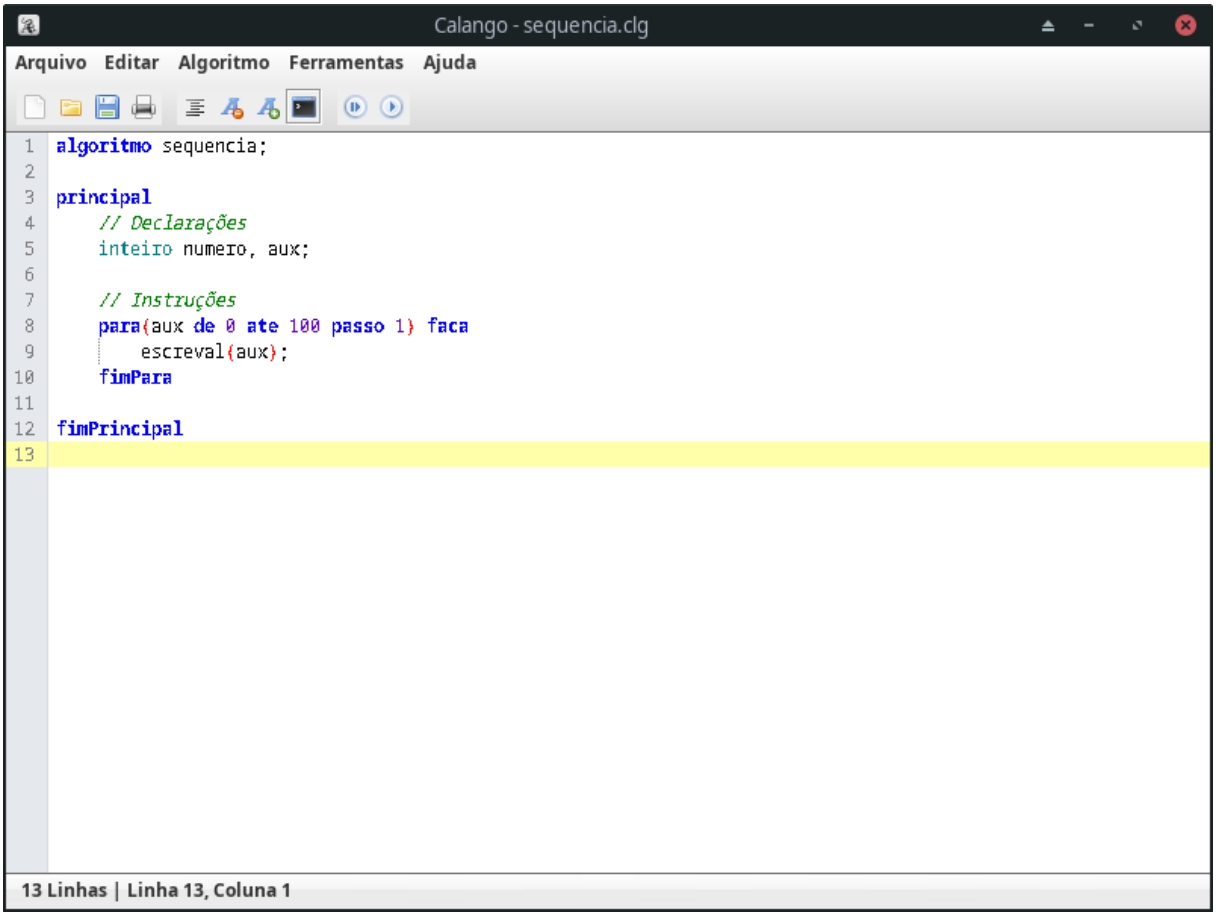

Figura 2 – Interface do Calango

Com *Turtle Graphics* gráficos vetoriais são desenhados utilizando um cursor relativo em forma de uma tartaruga, de onde vem a origem do nome. Essa biblioteca permite que os programadores controlem um ou mais cursores sobre um espaço bidimensional, possibilitando a criação de desenhos a partir da programação, isso é feito a partir de comandos para o cursor se mover para frente ou para trás, mudar o ângulo do movimento, além de alterar cor e espessura dos traços (FOUNDATION, 2021).

A Figura 3, apresenta um exemplo de um desenho gerado utilizando o *Turtle*. Na figura, o bloco de código à esquerda resulta no desenho vetorizado à direita.

No Pytuguês, os comandos do *Turtle* também são traduzidos para a língua portuguesa e podem ser executados na interface gráfica chamada “Tugalinha”. Para criação de seus desenhos o estudante pode utilizar diversas estruturas de controle, como laços e condições, como na Figura 4, em que o Pytuguês foi utilizado para desenhar uma curva de Koch.

O Pytuguês possui um interface nativa e também é multiplataforma, sendo necessário ter o Python 3 instalado na máquina. Sua interface é composta por um mural onde são exibidos os desenhos, o editor de código, e um terminal de entrada e saída. Assim

```
from turtle import *
color('red', 'yellow')
begin_fill()
while True:
    forward(200)
    left(170)
    if abs(pos()) < 1:
        break
end_fill()
done()
```

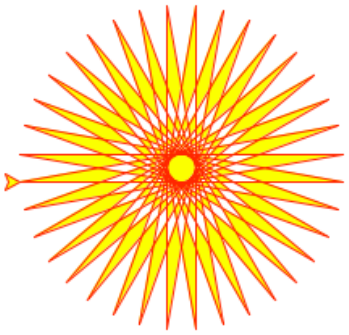

Figura 3 – Exemplo de implementação do *Turtle Graphics* com Python

como nas ferramentas anteriores, o editor também carece de recursos modernos, como a sugestão de palavras e ferramentas de refatoração. Um outro ponto negativo, é a falta de recursos de controle de depuração, recurso presente nas alternativas anteriores, como execução passo-a-passo.

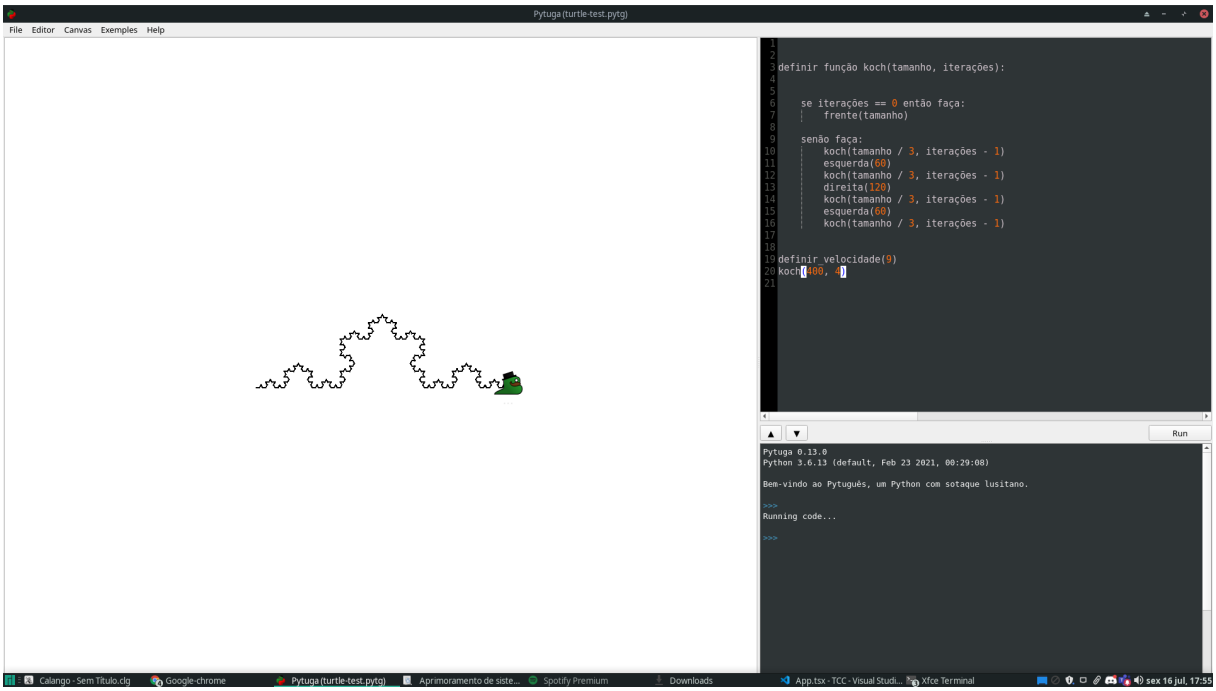

Figura 4 – Interface do Pytuguês

Entre as ferramentas que utilizam linguagem de programação em português, o Pytuguês é o que possui a maior facilidade de transição para uma linguagem de programação amplamente utilizada na indústria como o próprio Python na versão 3, pois o interpretador do Pytuguês é capaz de interpretar, também, código Python. O código Python pode ser misturado livremente com o Pytuguês já que o segundo é um superconjunto da linguagem Python (PYTUGUêS, 2015).

1.1.4 Scratch

O Scratch é uma ferramenta que permite ao aluno criar histórias interativas, jogos e animações utilizando blocos lógicos. Idealizado por Mitchel Resnick, o Scratch foi projetado principalmente para introduzir programação a crianças e adolescentes de 8 a 16 anos, mas é utilizado por pessoas de todas as idades.

Usado em mais de 150 países (SCRATCH, 2021), o Scratch utiliza uma abordagem diferente das ferramentas citadas anteriormente. Ele foca totalmente em ensinar os conceitos da programação através de elementos visuais e blocos lógicos, os quais podem ser encaixados um ao outro, criando uma sequência de comandos a serem executados. Assim, os alunos aprendem estratégias importantes para a resolução de problemas, projetos de design e comunicação de ideias.

Disponível para os principais sistemas operacionais (Windows; MacOS; ChromeOS; Android), o Scratch também possui uma versão online, em que basta o estudante acessar o link para ter um ambiente pronto para começar a aprender a programar.

Assim como o Pytuguês, apresenta uma ferramenta gráfica de desenhos vetoriais baseada na linguagem LOGO. O Scratch apresenta uma área em sua interface utilizada para exibir o resultado gráfico do código. Com o Scratch o aluno pode interagir de diversas formas, como por exemplo, adicionando sons e imagens, ou elementos especiais como exibir balões de diálogos e botões de ação.

Como é voltado para crianças, a interface do Scratch, figura 5, é simples e intuitiva. Ela conta com uma lista de opções de blocos que podem ser utilizados arrastando e encaixando um aos outros. Também há uma seção onde os blocos encaixados são exibidos e durante a execução é possível acompanhar qual bloco está sendo executado em determinado momento. Por fim, temos a seção de controle e exibição dos cenários, onde o aluno pode acompanhar o resultados da execução dos blocos semelhante ao canvas do Pytuguês, e também alterar elementos do cenário, como imagem de fundo e personagens.

A grande popularidade do Scratch em vários países é explicada, em parte, por sua disponibilidade em diversos idiomas, possibilitando o seu uso em países que não tem o inglês como idioma principal. Entretanto, o seu uso em faculdades não é muito indicado, devido ao foco em temas infantis e uma transição difícil para uma linguagens de programação industrial.

1.1.5 Replit

Criado por Amjad Masad e Faris Masad em 2016, o Replit é um ambiente online que permite que os usuários escrevam código e construam aplicações utilizando o navegador, e pode ser utilizado como ferramenta de estudo por iniciantes ou como ferramenta de trabalho para profissionais.

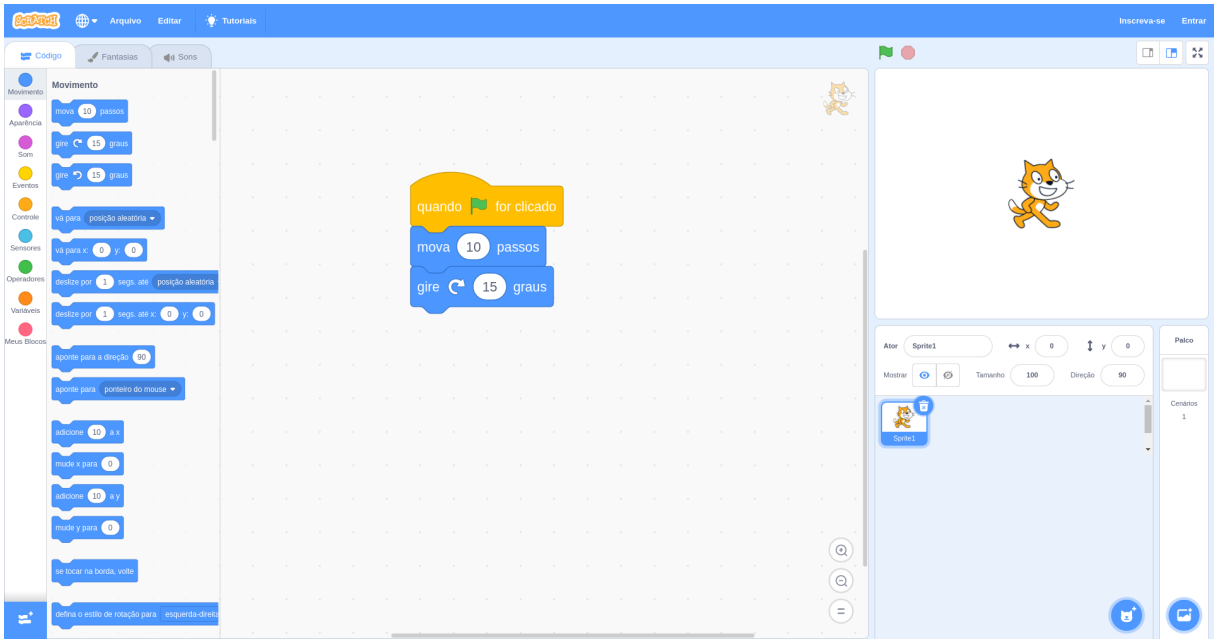

Figura 5 – Interface do Scratch

Atualmente o Replit possui suporte a mais de 50 linguagens de programação (REPLIT, 2021), incluindo as mais utilizadas na introdução à programação como C, Python e Java. Possui diferentes interfaces e ferramentas integradas a cada linguagem, se adaptando a diferentes tipos de uso. No ambiente Python, possui funcionalidades semelhantes às do Pytuguês, como a possibilidade de criar desenhos utilizando o *Turtle Graphics*.

Além disso, o professor pode criar exercícios e compartilhá-los com os alunos, através de salas de aula virtuais. Assim, o professor acompanha o desempenho individual dos alunos.

Como podemos ver na Figura 6, a interface do Replit é dividida em três partes, o editor de código, o terminal de entrada e saída, e um canvas para exibição dos desenhos criados com o *Turtle Graphics*. Diferentemente das ferramentas anteriores, o seu editor de texto possui as funcionalidades de sugestão de palavras e a possibilidade de utilizar múltiplos cursores.

A arquitetura utilizada pelo Replit é cliente-servidor, onde o código a ser executado é enviado a um servidor, que interpreta o código e retorna as saídas para o cliente, que tem a responsabilidade de exibir essas saídas ao usuário. Esta arquitetura limita a utilização do Replit a ambientes permanentemente ligados à internet e com uma conexão confiável, pois o código é executado remotamente.

1.1.6 Basthon

Basthon é um ambiente de programação em Python 3 distribuído como uma aplicação web. O Basthon utiliza uma versão do interpretador do Python adaptado para in-

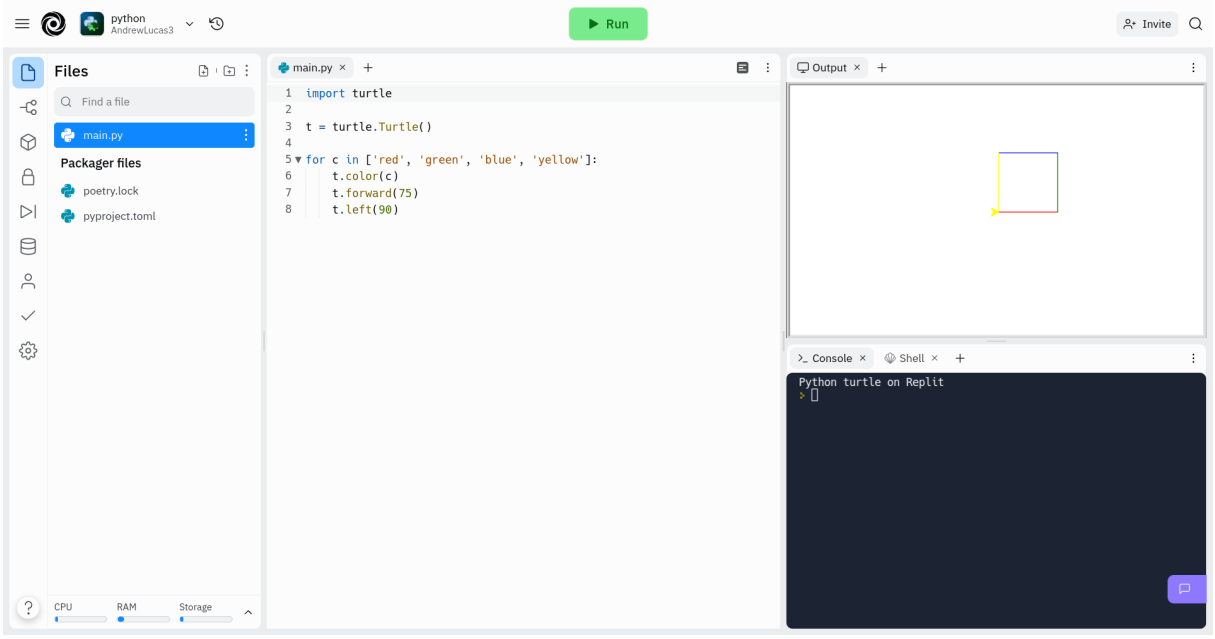

Figura 6 – Interface do Scratch

terpretar e executar o código no navegador somente no lado do cliente, sem compartilhar informação com um servidor. Segundo o autor dessa ferramenta, o Basthon foi especialmente projetado para respeitar a privacidade de seus usuários (BASTHON, 2021).

O Basthon integra várias bibliotecas populares como, Pandas e Matplotlib para tarefas relacionadas a ciência de dados, e também possui o *Turtle Graphics* integrado.

A arquitetura do Basthon é dividida em 2 módulos, o primeiro e mais importante é o Basthon-Kernel que é responsável por executar o código Python. A interação com o *kernel* é feita utilizando o Basthon-Console que é a implementação da interface gráfica e é o elemento por qual os usuários interagem.

Como o Basthon é um projeto francês utilizado em escolas na França, sua interface de usuário e documentação é escrita em francês, limitando assim, a aplicação dessa ferramenta no Brasil.

A interface do Basthon, Figura 7, possui o editor código simples somente com o destaque de cores para sintaxe do Python, e através do menu o usuário pode escolher entre visualizar o terminal de saída, ou o canvas onde são exibidos os desenhos ou gráficos gerados pelo código.

No menu do Basthon, ainda temos algumas outras funcionalidades, como a opção de baixar o código para computador, ou compartilhar o código através de um link. Outro ponto importante é que o código é salvo em tempo real no armazenamento local do navegador, assim não há risco de perder o trabalho feito durante as perdas de conexão.

Dentre as ferramentas focadas em remover a barreira do inglês, destaca-se o Py-

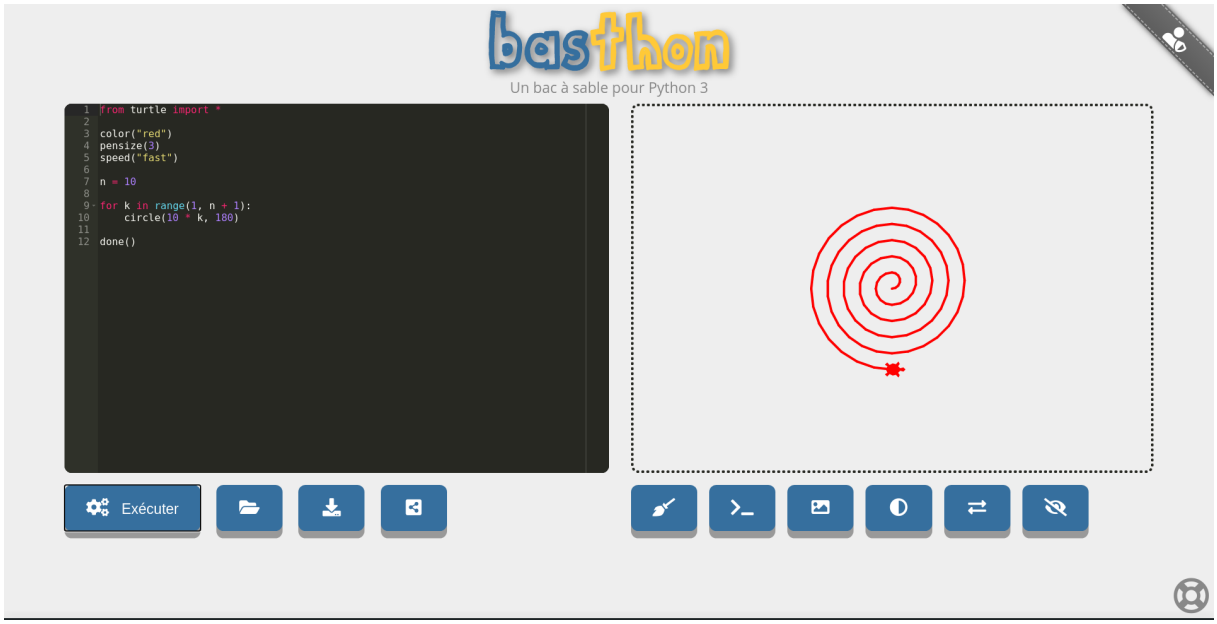

Figura 7 – Interface do Basthon

tuga, por, além de facilitar o ensino de programação para falantes de língua portuguesa, possibilita novas aplicações de conceitos de programação ao utilizar o *Turtle Graphics* como suporte para criação de desenhos gráficos, facilitando a compreensão da execução de um programa.

Dentre a ferramentas em ambiente web, podemos destacar o Basthon pela possibilidade de executar código Python diretamente no navegador, sem a necessidade de um servidor *backend*. Além de garantir a privacidade dos usuário ao não armazenar qualquer tipo de dado de acesso.

2 Proposta de desenvolvimento

O crescimento da popularidade do Python devido a várias aplicações em áreas como, aprendizado de máquina, desenvolvimento web e educação (PYPL, 2022), faz com que o Pytuguês, dentre as linguagens criadas com mesmo intuito, se torne uma ferramenta muito interessante para a introdução a programação de falantes de língua portuguesa.

Tendo isso em vista, a proposta deste trabalho é a criação de um ambiente de ensino de programação online, onde o usuário poderá executar código escrito em Pytuguês e/ou Python diretamente de um navegador web. E também, utilizar o pacote Turtle Graphics e/ou Tugalinhas para criar e visualizar desenhos vetorizados utilizando código. A utilização de um ambiente web facilita muito o processo de distribuição e manutenção do ambiente de programação dos estudantes.

Neste Capítulo será detalhado a arquitetura e soluções escolhidas para o desenvolvimento do Pytuguês online. E está estruturado da seguinte forma, a seção 1, descreve as solução para executar código Pytuguês no navegador. A seção 2, apresenta a arquitetura escolhida e seus componentes. Na seção 3 descreve as tecnologias que serão utilizada para apoiar o desenvolvimento.

2.1 Executando Pytuguês no navegador

O primeiro grande desafio para o desenvolvimento dessa ferramenta, é como executar o código em Pytuguês e seus recursos em um navegador web. Como o interpretador do Pytuguês é escrito em Python, é necessário primeiro executar o código Python no navegador. Dentre as tecnologias encontradas com esse propósito as principais alternativas de código-aberto: Brython, Skulpt e Pyodide.

Brython é um implementação do Python versão 3 escrita em Javascript. E tem como objetivo substituir o Javascript na manipulação do DOM, que são os objetos que formam uma página web. Por esse motivo seu código é executado somente durante o carregamento inicial da página, por isso não pode ser utilizado nesse trabalho que tem como foco a execução iterativa do código após o carregamento da página.

O Skulpt é, também, uma compilador do Python reescrito em Javascript, porém o Skulpt se baseou inicialmente no Python 2, e posteriormente foi gradualmente atualizado para versão 3. Diferente do Brython, no Skulpt a compilação do Python para Javascript acontece em tempo de execução, ou seja a qualquer momento após a página ser carregada. Skulpt foi originalmente criado para produzir ferramentas educacionais que precisam de uma sessão Python em uma página da web.

Até a última versão disponível, o Skulpt não possui a implementação das funções *exec* e *eval* do Python. Essas funções são essenciais para a utilização Pytuguês, pois são utilizadas para avaliar uma expressão Python.

Segundo relato da equipe do projeto Basthon, o projeto utilizou o Brython e Skulpt em seu desenvolvimento, mas por problemas de implementação dessas tecnologias, migraram para o Pyodide (BASTHON, 2021). E após testar a integração de cada uma dessas tecnologias com o Pytuguês, o Pyodide foi a tecnologia escolhida para esse trabalho.

2.1.1 Pyodide

Diferente das bibliotecas mencionadas anteriormente, o Pyodide não é uma implementação parcial do Python utilizando Javascript. Na realidade, ele fornece a implementação original do Python através da compilação do Interpretador do Python (CPython) em WebAssembly.

WebAssembly é uma linguagem de programação que pode ser executado nos navegadores modernos e fornece ganhos em performance e utilização de memória quando comparado ao Javascript. Com ele é possível compilar código em linguagens como C/C++, Rust, entre outras, e executar diretamente no navegador utilizando o Javascript como interface para interagir com os módulos em WebAssembly (MDN, 2021).

Como o CPython, que é a implementação principal da linguagem de programação Python, é escrita em linguagem C, o time responsável pelo Pyodide compilou a implementação do Python 3.9, para WebAssembly, junto com outros pacotes utilizados principalmente na ciência de dados como Pandas, Numpy e Matplotlib (PYODIDE, 2021).

O Pyodide não é uma simples recompilação do código original Python, já que modificou algumas APIs com o objetivo de fornecer uma conversão transparente entre os objetos Python e Javascript. E também possibilitar o acesso a APIs web pelo código Python (PYODIDE, 2021).

Como o Pyodide inclui aproximadamente 3,5MB no carregamento inicial da página, é necessário aplicar um esforço em diminuir o tempo de carregamento. Para isso podemos utilizar soluções como cacheamento ou a utilização de uma *build* parcial do Pyodide que inclua apenas os pacotes necessários para o projeto (PYODIDE, 2022). Ainda há a possibilidade de carregar o Pyodide de forma assíncrona utilizando um *Web Worker*.

Um *Web Worker* é uma forma de executar código Javascript de forma assíncrona. Esse tipo de execução possibilita que operações mais pesadas e demoradas possam ser executadas sem resultar em um congelamento na renderização da interface, atrapalhando a experiência do usuário (MDN, 2022).

Ao utilizar um *Web Worker* para executar o Pyodide, o tempo necessário para o

carregamento inicial é reduzido pois a interface gráfica pode ser inicializada antes do carregamento do interpretador. Porém, essa abordagem aumenta significativamente a complexidade do desenvolvimento. Desta forma a implementação utilizando um *Web Worker* será adicionado como uma melhoria futura.

2.1.2 Pytuguês

O Pytuguês, como linguagem voltada para iniciantes, traz uma sintaxe simples como a do Python, e que em alguns casos basta literalmente traduzir o idioma dos comandos de uma linguagem e obterá o código da outra, como são os casos das estruturas de repetição (Fig. 8) e condição (Fig. 9).

```
# Pytuguês
para cada número em [1,3,5,7,9,11]:
    mostrar(número)

#Python
for numero in [1,3,5,7,9,11]:
    print(numero)
```

Figura 8 – Exemplo de iteração de lista em Pytuguês

```
# Pytuguês
número = 42
se número == 42:
    mostrar('Parabéns!')
senão:
    mostrar('Errou!')

#Python
numero = 42
if numero == 42:
    mostrar('Parabéns!')
else:
    mostrar('Errou!')
```

Figura 9 – Exemplo de condicional em Pytuguês

Porém o Pytuguês traz algumas estruturas exclusivas que foram criadas com intuito de facilitar a compreensão sobre o que o código está fazendo, por exemplo, como alguns programadores iniciantes possuem uma certa dificuldade com a função range do Python, para evitar estas complicações o Pytuguês introduz a estrutura exibida na Figura 10. Ela realiza a mesma operação que a função range, porém de forma mais explicita.

Em alguns casos o Pytuguês apresenta uma sintaxe redundante como é o caso do "faça"ao final da estrutura de repetição apenas com o intuito de tornar a intenção do programador mais explícita, porém estas palavras redundantes são sempre opcionais.

```
#Pytuguês
para cada número de 0 até 10 a cada 2 faça:
    mostrar(número)

#Python
for numero in range(0,10,2):
    print(numero)
```

Figura 10 – Exemplo de repetição em um intervalo em Pytuguês

Para executar código em Pytuguês utiliza-se o Pytuga uma ferramenta de desenvolvimento e interpretador do Pytuguês. O Pytuga é um pacote escrito em Python, que expõe um método *exec* que realiza a transformação do código Pytuguês em Python, e o executa utilizando a função *exec* nativa do Python.

Uma vez que o Pyodide permite o carregamento de pacotes Python externos, podemos utiliza-lo para executar o módulo do interpretador do Pytuguês no navegador.

2.2 Arquitetura

A utilização do Pyodide permite que toda a aplicação seja completamente executada no lado do cliente, removendo assim a necessidade de um *backend*. Desta forma, podemos dividir a ferramenta em duas partes: o módulo Pyodide, responsável pela execução de todo código Python; e o módulo de interface, responsável pela interação com o usuário.

Conforme mostrado na Figura 11, o módulo de interface possui três componentes principais: o editor de código para receber as entradas do usuário, o console e o HTML Canvas para exibir os resultados da execução do código. O módulo Pyodide, apresenta dois componentes principais: o componente responsável pela execução do código em Pytuguês, e o Turtle Graphics para gerar as imagens vetoriais resultantes da execução.

2.3 Interface

Esta seção descreve as tecnologias utilizadas na camada de interface, assim como o detalhamento dos três componentes que a formam, sendo eles: console, editor de código e HTML Canvas.

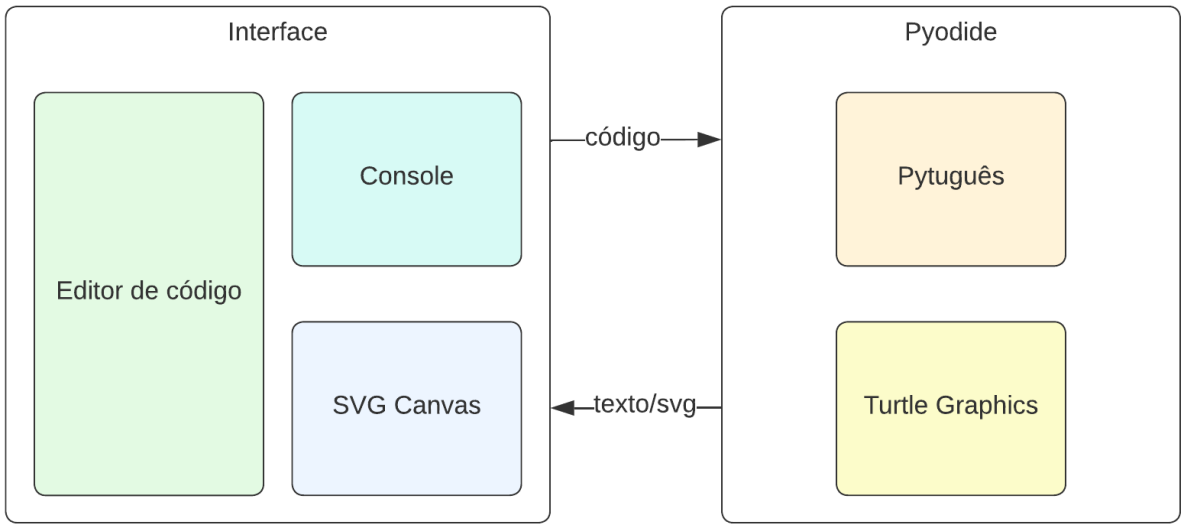

Figura 11 – Arquitetura

2.3.1 Console

O console é o componente responsável por exibir os resultados textuais da execução do programa executado. E caso ocorra um erro de execução, é nesse componente que o usuário pode visualizar os detalhes do erro. Assim, para implementação desse componente, será utilizado apenas HTML e CSS, e a utilização de Javascript para manipular o elemento HTML inserindo os resultados retornados pela execução do programa.

2.3.2 Editor de código

O editor de código é o componente por onde o usuário pode editar o código do programa. Sendo o principal componente de interação do usuário com a aplicação, ele exige uma atenção maior quanto a suas funcionalidades. Funcionalidades como auto-indentação, seleção múltipla, buscar e substituir, destaque de sintaxe e autocompletar palavras são essenciais a um editor de código moderno e também a este projeto.

Existem diversos editores de código com funcionalidades essenciais já implementadas, e que podem ser facilmente integrados à ferramenta deste trabalho. Dessa forma, faremos uma comparação entre essas as principais soluções disponíveis e que podem ser utilizadas. Além das funcionalidades essenciais já destacadas, será levado em consideração o grau de customização da funcionalidade destaque de sintaxe e autocompletar.

Três editores de código se encaixam nos requisitos definidos e foram selecionados para análise: CodeMirror, Ace Editor e Monaco Editor. São três dos editores web mais populares e serão detalhados nos próximos parágrafos.

O CodeMirror é um editor de texto versátil focado no uso em páginas web implementado em Javascript. Esse editor também apresenta as principais funcionalidades

necessárias para este projeto. E permite a implementação de modos de linguagens, possibilitando a customização de destaque de sintaxe e autocompletar palavras.

O Ace Editor é um editor focado em alto desempenho para web. Chegando a atingir desempenho semelhante ao de editores nativos como Sublime e Vim. Ele pode ser facilmente incorporado em qualquer página da web e aplicativos escritos em Javascript. Entre suas funcionalidades embutidas encontramos suporte a todas funcionalidades consideradas essenciais a este projeto. Em particular, o Ace Editor permite a customização de destaque de sintaxe e da função de autocompletar palavras.

O Monaco Editor, assim como as alternativas anteriores, é um editor que apresenta as principais funcionalidades de editores modernos, e permite um alto grau de customização. Apesar de não possuir suporte a dispositivos móveis, o Monaco Editor permite a inserção de elementos não editáveis entre as linhas de código, como mostra a Figura 12. Esta funcionalidade possibilita a exibição de elementos que auxiliem o aluno durante o seu aprendizado, como elementos explicativos ou exemplos de blocos de códigos.

Figura 12 – Monaco editor com HTML em linha

Além disso, o Monaco Editor é o editor utilizado no software VS Code da Microsoft, um dos editores de código mais utilizados entre os desenvolvedores (OVERFLOW, 2021). Assim, o iniciante que utilizar este projeto em seu aprendizado estará mais familiarizado com suas funcionalidades ao migrar para o VS Code.

Os três editores apresentam funcionalidades semelhantes, mas o Monaco Editor se destaca em possibilitar a inserção de elementos não editáveis em linha, possibilitando a implementação de diversas funcionalidades que auxiliam o aluno no processo de estudo.

Desta forma o Monaco Editor foi o editor escolhido para este trabalho.

2.3.3 HTML Canvas

O componente HTML Canvas é o componente responsável pela exibição de elementos gráficos resultantes da execução do programa. Este elemento é composto por um HTML canvas, que pode ser manipulado através da Canvas API, que provê maneiras de desenhar gráficos via Javascript. O que possibilita a utilização para animação, gráficos de jogos, visualização de dados, manipulação de fotos e processamento de vídeo em tempo real. Neste projeto este componente será utilizado para exibir os desenhos gerados a partir da biblioteca Turtle Graphics.

2.4 Tecnologias de apoio

Para auxiliar o desenvolvimento desse trabalho, algumas tecnologias auxiliares foram utilizadas, são elas: Vue.js e Tailwind css. O Vue.js é um framework Javascript com foco em alto desempenho e versatilidade para a construção de interfaces de usuário da web. Enquanto o Tailwind css é um *utility-first* framework de CSS, e tem o objetivo de facilitar a estilização da aplicação.

3 Desenvolvimento

Este capítulo relata o desenvolvimento da ferramenta proposta.

3.1 Planejamento

3.1.1 Metodologia de Desenvolvimento do Software

Segundo (SOMMERVILLE, 2010), metodologia de *software* é um conjunto de práticas recomendadas para o desenvolvimento de um *software*.

Para organizar as atividades a serem executadas neste trabalho, foi utilizado o *Kanban*. No *Kanban* utiliza-se um quadro para organizar as tarefas ao longo do desenvolvimento, deixando transparente o estado que se encontra cada uma das atividades (KNIBERG, 2010).

3.1.2 Divisão do trabalho

A Tabela 1 mostra a distribuição das atividades entre TCC1 e TCC2. Desta forma o TCC1 foi utilizado para elaboração e validação da proposta, enquanto no TCC2 ocorreu o desenvolvimento e entrega da ferramenta.

Tabela 1 – Divisão do trabalho

Tarefa	**Etapa**
Analisar do estado da arte das ferramentas de ensino de programação	TCC1
Desenhar arquitetura para ferramenta proposta	TCC1
Validar proposta de arquitetura	TCC1
Desenvolver protótipo	TCC1
Escrever TCC1	TCC1
Planejar desenvolvimento da ferramenta	TCC2
Desenvolver ferramenta	TCC2
Apresentar ferramenta	TCC2
Escrever TCC2	TCC2

3.1.3 Protótipo

Para verificar a viabilidade da proposta deste trabalho foi desenvolvido um protótipo (Fig. 13). Além de validar a proposta, este protótipo foi utilizado para experimentar

as diferentes formas de executar Python no navegador e os diferentes editores de código, afim de embasar a escolha da melhor solução para este trabalho.

Figura 13 – Protótipo

3.2 Desenvolvimento

O desenvolvimento iniciou-se com a criação e preparação inicial de um projeto *Vue.js* e instalando a biblioteca Tailwind css para auxiliar na estilização da aplicação. Com o projeto preparado o desenvolvimento seguiu a seguinte ordem, primeiro a integração do Pyodide com Pytuguês, depois a implementação do editor do código e suas funcionalidades e por fim a adição das funcionalidades extras para melhorar o uso da ferramenta. Nesta Seção será detalhado todas etapas desse processo de desenvolvimento.

3.2.1 Pyodide

Para possibilitar a execução de código Python no navegador, realizou-se a instalação e configuração do Pyodide na versão 0.20.0 que utiliza o Python na versão 3.9. Para executar código Python basta chamar a função *runPythonAsync* da API do Pyodide passando o código como argumento. Por padrão as saídas de programa são exibidas no console do navegador e as entradas são inseridas através do *prompt* nativo do navegador, porém é possível sobrescrever os métodos de entrada e saída.

Desta forma, o *stdout* e o *stderr* foram sobrescritos para direcionar as saídas para o componente console, onde são exibidos os resultados e erros da execução. Por limitações do Pyodide o método de entrada (*stdin*) não foi alterado.

3.2.2 Pytuga

Devido a falta de manutenção de algumas dependências, o Pytuga, como é chamado o interpretador de Pytuguês, se encontra incompatível com as versões Python a partir da versão 3.8. Como a versão do Pyodide utilizado neste trabalho utiliza o Python na versão 3.9, foi necessário atualizar o código do Pytuga, mais especificamente do pacote utilizado para interpretar o código em português chamado Transpyler.

Verificou-se que o método responsável pela tradução do código Pytuguês para Python faz uso da classe *CodeType* que teve a assinatura do seu método construtor alterada na versão 3.8 do Python. Desta forma foi realizada uma contribuição ao repositório do Transpyler (<https://github.com/Transpyler/transpyler>) corrigindo a assinatura do método conforme a versão mais recente.

3.2.3 Tugalinhas

A versão do Turtle Graphics embutido no Pytuga, tem como dependência o *Qt* que é um biblioteca utilizada na criação de interfaces gráficas nativas. Desta forma não foi possível utilizá-lo visto que o Pyodide não possui suporte ao *Qt*.

Então para este trabalho foi utilizado uma implementação do Turtle Graphics feita pelo time do projeto Basthon (<https://framagit.org/basthon/basthon-kernel/-/tree/master/packages/kernel-python3/src/modules/turtle>link do repositório) para ser utilizada com Pyodide. Nesta implementação as instruções do *Turtle Graphics* são utilizadas para gerar um elemento SVG que pode ser facilmente exibido pelo navegador.

Diferente do pacote original do Tugalinhas, essa implementação utilizada pelo Basthon possui apenas comandos em inglês. Porém como o Pytuga permite estender o seu dicionário de traduções, esse problema foi solucionado com a criação de um dicionário de traduções dos comandos do Turtle Graphics e incluindo-os ao inicializar o Pytuga. Desta forma, as mesmas instruções utilizadas com o Tugalinhas podem ser utilizadas com essa reimplementação.

Buscando manter a essência do Tugalinhas original foi realizada uma modificação no código da implementação utilizada pelo Basthon para permitir a alteração da figura utilizada como pincel. Para isso foi adicionado um método que permite inserir uma imagem ao SVG gerado, então a imagem da tartaruga do Tugalinhas (Figura 14) foi inserida como opção padrão do cursor dos desenhos gráficos.

3.2.4 Executando o Pytuga no Pyodide

o Pyodide permite o carregamento de pacotes externos utilizando o módulo *micropip*, que um instalador de pacotes Python. Com este módulo embutido do Pyodide

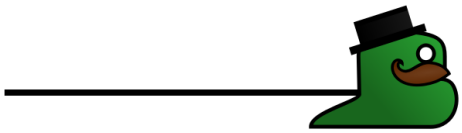

Figura 14 – Pincel do Tugalinhas

é possível o carregamento de pacotes através de uma URL desde que a distribuição do pacote esteja no formato *wheel*, que é um formato de distribuição Python mais moderno, leve e rápido (PEP427, 2022).

Assim, para carregar o Pytuga e suas dependências no Pyodide foi necessário gerar novas distribuições compatíveis com o *micropip*. Para isso foi utilizado o pacote *wheel* que permite utilizar o comando *bdist_wheel* para gerar as distribuições no formato *wheel*.

Com as distribuições geradas, é preciso torna-las acessíveis através de uma URL, para isso bastou inserir os arquivos dentro da pasta *public* do projeto Vue.js, todos arquivos dentro desta pasta podem ser acessados a partir da URL raiz da aplicação. Desta forma para carregar os pacotes com o *micropip*

Para carregar os pacotes é preciso passar uma lista de URL para o método *micropip.install()* conforme mostra a Figura 15, a variável *VITE_BASE_PATH* representa o caminho raiz do projeto. Com as dependências carregadas basta importar e utiliza-las no Pyodide.

```
async def setup_python():
    await micropip.install(
        [
            VITE_BASE_PATH + 'lazyutils-0.3.3-py3-none-any.whl',
            VITE_BASE_PATH + 'transpyler-0.5.0-py3-none-any.whl',
            VITE_BASE_PATH + 'colortools-0.1.2-py3-none-any.whl',
            VITE_BASE_PATH + 'pytuga-0.13.0-py3-none-any.whl',
            VITE_BASE_PATH + 'turtle-0.0.1-py3-none-any.whl'
        ],
        keep_going=True)
```

Figura 15 – Carregando pacotes com *micropip*

Com os pacotes carregados, é definida a função que importa e inicializa o Pytuga passando o dicionário de traduções do Turtle Graphics, e então executa o código Pytuguês. Os resultados e erros da execução são exibido conforme os métodos *stdout* e o *stderr* definidos na inicialização do Pyodide.

Por fim, conforme ocorre no Pytuga nativo, Turtle Graphics é importado, de forma implícita, em toda execução de código, desta forma o usuário pode usar os comandos sem a necessidade de realizar importação o pacote de forma explícita. Para isso é inserido na primeira linha do código a ser executado, a importação do pacote Turtle.

Para exibir o desenho resultante da execução, foi definido um método que modifica a função responsável por exibir o SVG, fazendo com que um evento DOM seja disparado contendo o desenho gerado. Assim o componente de exibição pode consumir esse evento e exibir o desenho no local correto.

Com todo esse código Python que precisa ser executado durante a inicialização da aplicação, foi criado um arquivo *.py* onde esses métodos são definidos. Assim, após a execução desse arquivo pelo Pyodide, a Pytuga está pronto para ser utilizado.

3.2.5 Editor de código

Para o editor de código foi utilizado o *Monaco Editor*. Esse editor possui suporte a diversas linguagens de programação, porém o Pytuguês não é uma delas. Desta forma, para fazer uso das funcionalidades disponíveis no *Monaco Editor* foi necessário criar uma configuração customizada para o Pytuguês.

O *Monaco Editor* fornece um método que permite a criação das configurações de uma linguagem customizada chamada *setLanguageConfiguration* que recebe como argumento o nome da linguagem e um objeto contendo as configurações. Como o Pytuguês possui uma estrutura muito similar ao Python, foi utilizada a configuração do Python disponível no repositório do próprio editor como base.

A partir das configurações do Python, foram adicionadas as traduções do Pytuguês e do Tugalinhas. Também foram adicionadas as regras para as palavras-chave exclusivas do Pytuguês como a estrutura mostrada na Figura 16 que não tem uma estrutura equivalente no Python. Assim o destaque de sintaxe pode ser aplicado corretamente. Essa configuração possibilita, também a utilização da sugestão para completar palavras, conforme a Figura 17.

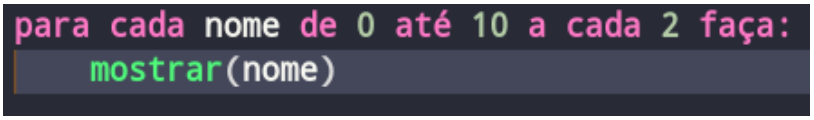

Figura 16 – Exemplo de estrutura exclusiva do Pytuguês

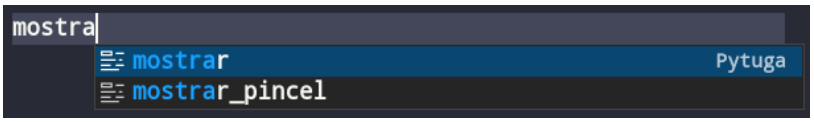

Figura 17 – Sugestão de palavra do Pytuguês no Monaco Editor

Com a configuração da linguagem definida, basta inicializar o *Monaco Editor* passando como argumento essas configurações. Após a inicialização do editor é possível definir teclas de atalho para determinadas ações, para este trabalho foi adicionado um atalho para executar o código apertando as teclas *Ctrl* e *Enter*.

3.2.6 HTML Canvas

Para exibir os desenhos gerados pelo *Turtle Graphics* utilizou-se um elemento HTML Canvas onde é exibido o SVG. Esses desenhos são recebidos pelo componente através do evento DOM emitido ao final da execução do código.

O elemento SVG gerado possui os valores constantes como *view box* que define a área do SVG exibida. Porém alguns desenhos podem extrapolar esses limites e não são exibidos completamente. Para resolver esse problema foi utilizado a biblioteca *panzoom* que permite adicionar funcionalidades de visualização ao SVG alterando dinamicamente a *view box* do elemento. Assim, o usuário pode alterar a visualização através de zoom ou alterando a posição do desenho utilizando o mouse ou com o toque em uma tela sensível ao toque.

3.2.7 Funcionalidades adicionais

Com o intuito de melhorar a usabilidade da aplicação foram adicionadas algumas funcionalidades baseadas nas aplicações analisadas.

Algo comum nos principais ambientes de desenvolvimento é a possibilidade de customizar a interface conforme o usuário desejar. Baseado nisso, foi adicionada a possibilidade de redimensionar os principais componentes da ferramenta. Essa funcionalidade foi implementada utilizando uma biblioteca chamada *Split.js*, que adiciona uma área de clique e arraste entre as fronteiras dos componentes.

No Pytuga Web o usuário consegue, conforme mostra a Figura 18, ocultar o console para visualizar melhor os desenhos, ou ocultar o desenho deixando mais espaço para o console (Figura 19). Fazendo com que a ferramenta se adapte conforme as necessidades do usuário.

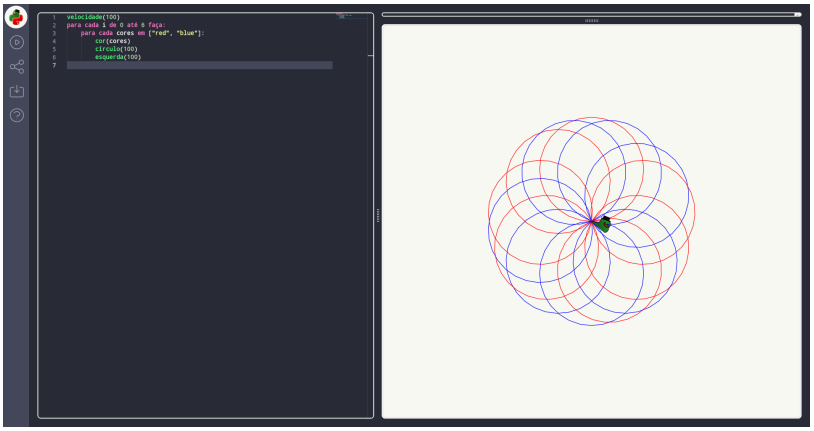

Figura 18 – Modo desenho

Como o Pytuga Web não salva qualquer tipo de dados do usuário, foi adicionada a possibilidade de baixar o código presente no editor de código para o seu armazenamento

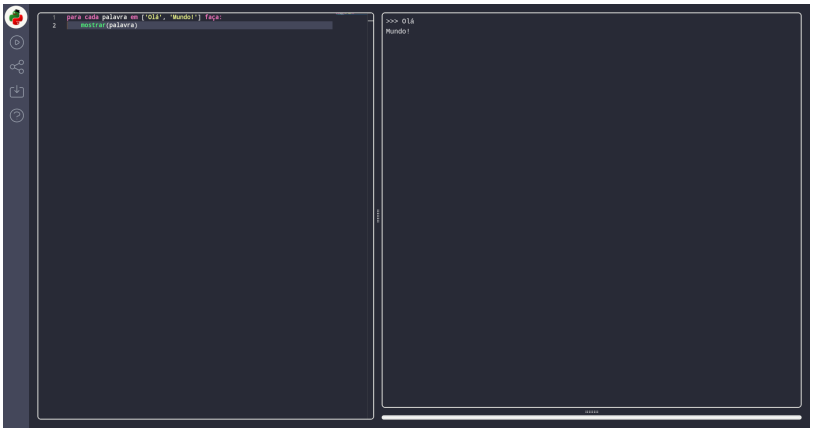

Figura 19 – Modo console

local. O arquivo é baixado no formato *.pytg*, que pode ser carregado na versão nativa do Pytuga.

Outra função adicionada ao projeto foi a possibilidade de compartilhar o código através de um link. Nesta funcionalidade o código é codificado no formato Base64 e então adicionado como um parâmetro à URL. Desta forma, quando o Pytuga web é acessado, é feita a verificação se o parâmetro está presente na URL, caso sim, ocorre a decodificação do valor do parâmetro e o resultado é inserido no editor de código, exibindo assim o código compartilhado.

3.3 Implantação

Como toda execução do Pytuga web está localizada no lado do cliente através do carregamento de arquivos estáticos. A implantação pode ser realizado através da hospedagem desses arquivos em algum serviço de hospedagem de arquivos. Como o repositório do projeto se encontra hospedado no *Github*, optou-se por utilizar o *Github Pages* para a implantação.

Então para fazer a implantação basta realizar o *build* do projeto seguindo as instruções do README do repositório deste trabalho, e armazenar os arquivos gerados na *branch* chamada gh-pages que é utilizada pelo *Github Pages* para servir os arquivos. Após esse processo já é possível acessar e usar o Pytuga Web através do link gerado pelo *Github Pages*.

4 Resultados

Este capítulo apresenta as conclusões e trabalhos futuros propostos. Ele apresenta a ferramenta resultante do desenvolvimento e examina esses resultados.

4.1 Pytuga Web

Ao final do desenvolvimento da ferramenta proposta neste trabalho é possível responder a questão de pesquisa *"É possível desenvolver uma ferramenta de ensino de programação capaz de executar Pytuguês e que seja acessada pelo navegador?"*.

Sim, é possível, o Pytuga Web é capaz de auxiliar professores e estudantes no estudo de programação, principalmente no início dos estudos, sendo capaz de executar código Pytuguês.

A ferramenta desenvolvida visa atender as demandas de inciantes a programação, seja dentro de uma sala de aula ou de forma autodidata. Ela se destaca das demais alternativas, pois facilita o estudante na transição para linguagens e ferramentas industriais, visto que possibilita, também a execução de código Python.

A facilidade no acesso a ferramenta é outro fator de destaque, para utilizar o Pytuga Web é preciso possuir acesso a internet e acessar a ferramenta através do navegador. Desta forma, não é necessário nenhum tipo de instalação ou cadastro. Também é possível utilizar a ferramenta através de dispositivos móveis, porém é necessário melhorar a usabilidade da interface nesses dispositivos, já que o desenvolvimento da ferramenta foi focado no uso em computadores.

Ao acessar o Pytuga Web pela primeira vez o usuário verá a tela da Figura 20. A interface é composta pelos três componentes principais: editor de código, console, e tela onde são exibidos os desenhos. A esquerda se encontra o menu de ações com as opções de executar, compartilhar ou baixar o código, há também a opção "sobre" que exibe as informações da ferramenta.

Com o objetivo de facilitar a compreensão do usuário sobre o funcionamento da ferramenta, no primeiro carregamento da página o editor é preenchido com um código exemplo que ao ser executado imprime uma mensagem de boas vindas, e exibe um desenho criado com Tugalinhas o nome da ferramenta conforme a Figura 20.

E após esse carregamento inicial da aplicação o usuário já pode iniciar seus estudos e experimentos com a linguagem Pytuguês ou até mesmo Python. Podendo criar desenhos como o apresentado na Figura 21 utilizando código, enquanto aprende os conceitos básicos

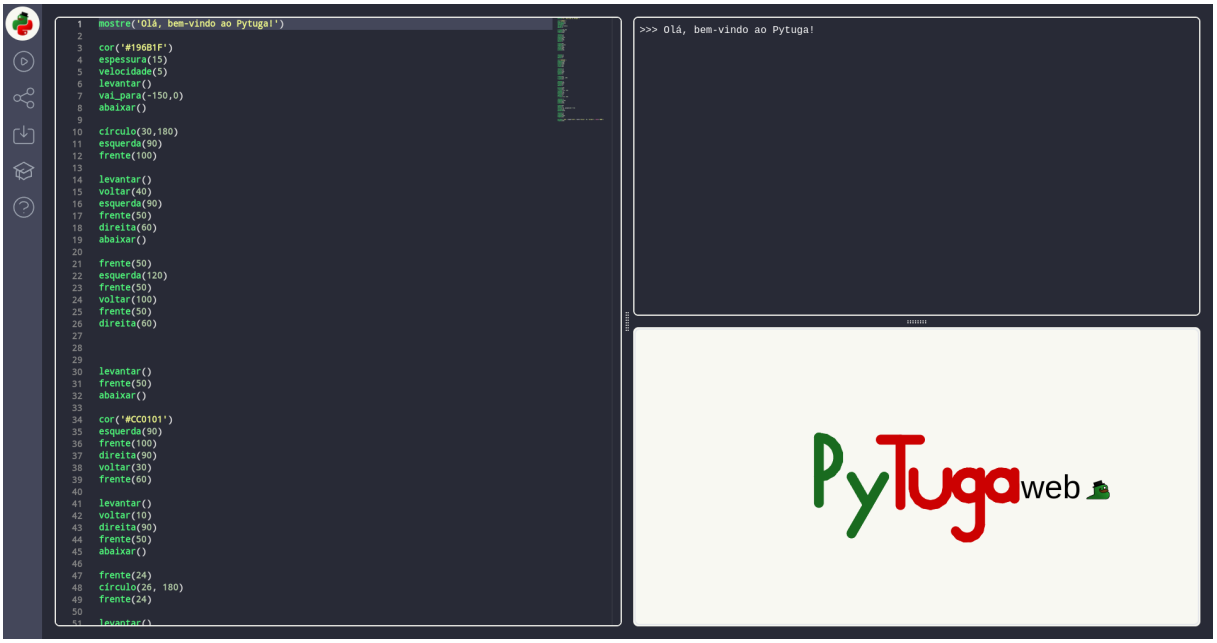

Figura 20 – Tela principal do Pytuga Web

da programação e como funcionam os algoritmos.

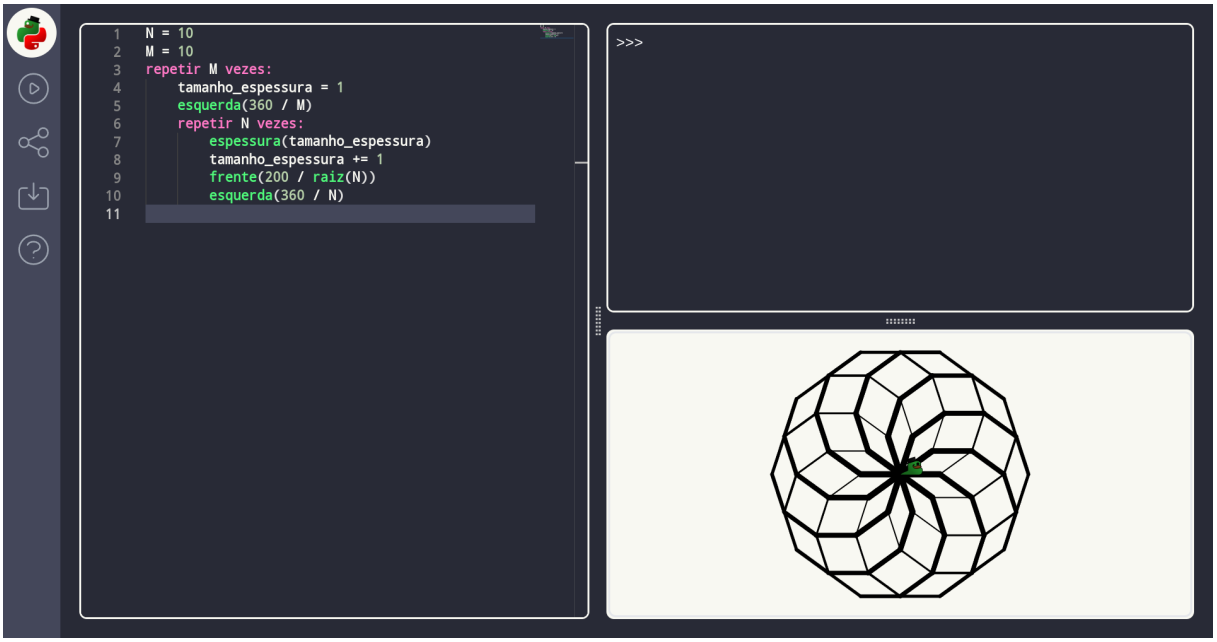

Figura 21 – Exemplo de desenho criado no Pytuga Web

Além da ferramenta desenvolvida, este trabalho também tem como resultado o pacote do Turtle Graphics adaptado para navegador, que pode ser facilmente integrado a outros projetos. E também a contribuição feita ao projeto Transpyler tornando-o compatível com as versões mais recentes do Python, e assim pode ser utilizado não só pelo Pytuga Web como pelo Pytuga nativo.

A seguir os links para os resultados deste trabalho:

- Pytuga Web: <https://andrewlucasgs.github.io/pytuga-web/>
- Código fonte do Pytuga Web: <https://github.com/andrewlucasgs/pytuga-web>
- Código fonte do Turtle Graphics adaptado: <https://github.com/andrewlucasgs/turtle-for-pytuga-web>
- Contribuição feita ao Transpyler: <https://github.com/Transpyler/transpyler/pull/21>

5 Considerações finais

Com a construção deste trabalho, foi possível aplicar e testar técnicas e metodologias aprendidas durante a graduação. Reforçando a importância do estudo da teoria para construção de um produto de qualidade.

Estudar programação é uma fase pela qual todos desenvolvedores passam, e o desenvolvimento de ferramentas e metodologias que facilitem esse processo é extremamente essencial. E a diversidade de soluções presentes, colabora para que cada pessoa encontre e use a sua ferramenta ideal.

A expectativa é a ferramenta desenvolvida neste trabalho venha a ser utilizada por professores para introduzir conceitos e estruturas iniciais da programação, e que seja simples e intuitiva para que os alunos se preocupem apenas em aplicar esses conceitos.

O autor deste trabalho pretende realizar a evolução deste projeto, para torna-lo ainda mais maduro, com novas funcionalidades e contribuindo com a manutenção e evolução da linguagem Pytuguês, pois ela possui um grande potencial para ser utilizada no ensino de programação, em especial, para crianças.

5.1 Trabalhos futuros

Nesta seção são apresentadas algumas melhorias que podem ser aplicadas à ferramenta, mas que estão fora do escopo deste trabalho. Essa melhorias tem como objetivo tornar a ferramenta cada vez mais completa e flexível, para que possa auxiliar mais pessoas no ensino de programação. Assim, as seguintes possíveis melhorias foram levantadas:

- Melhorar a usabilidade da interface em ambiente *mobile*
- Executar o Pyodide utilizando Web Worker para evitar travamentos da tread principal em operações mais pesadas utilizando Python.
- Adicionar suporte a outras bibliotecas gráficas, como exibir gráficos plotados usando Matplotlib do Python, ou alguma biblioteca de jogos.
- Implementar sistema para exibir dicas de implementação à medida que o aluno digita o código, isso pode ser feito utilizando a funcionalidade do Monaco Editor de inserir elemento HTML incorporado ao editor.
- Implementar sistema inteligente de *hot-reload*, possibilitando ao aluno realizar uma alteração no programa e visualizar o resultado sem a necessidade de executar o programa do início.

Referências

ANTÔNIO, C. *PORTUGOL (Português Estruturado) E VisuAlg Como TUDO COMEÇOU*. 2017. Disponível em: <https://visualg3.com.br/portugol-portugues-estruturado-e-visualg-como-tudo-comecou/>. Citado na página 24.

BASTHON. 2021. Disponível em: <https://basthon.fr/about.html>. Citado 2 vezes nas páginas 30 e 34.

BENNEDSEN, J.; CASPERSEN, M. Failure rates in introductory programming. *SIGCSE Bulletin*, v. 39, p. 32–36, 06 2007. Citado na página 23.

FOUNDATION, P. S. *Turtle - turtle graphics*. 2021. Disponível em: <https://docs.python.org/3/library/turtle.html>. Citado na página 26.

JÚNIOR, M. Experiências positivas para o ensino de algoritmos; how to teach algorithms. In: . [S.l.: s.n.], 2004. Citado na página 19.

KNIBERG, H. *Kanban and Scrum - Making the Most of Both*. [S.l.]: Lulu.com, 2010. ISBN 0557138329. Citado na página 41.

MDN. 2021. Disponível em: <https://developer.mozilla.org/pt-BR/docs/WebAssembly/Concepts>. Citado na página 34.

MDN. 2022. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/API/Web_Workers_API>. Citado na página 34.

OVERFLOW, S. *2021 Developer Survey*. 2021. Disponível em: <https://insights.stackoverflow.com/survey/2021#section-most-popular-technologies-integrated-development-environment>. Citado na página 38.

PEARS, A. et al. A survey of literature on the teaching of introductory programming. *SIGCSE Bull.*, Association for Computing Machinery, New York, NY, USA, v. 39, n. 4, p. 204–223, dez. 2007. ISSN 0097-8418. Disponível em: <https://doi.org/10.1145/1345375.1345441>. Citado na página 23.

PEP427. 2022. Disponível em: <https://peps.python.org/pep-0427/#canonical-specification>. Citado na página 44.

PYODIDE. *pyodide/pyodide*. Zenodo, 2021. Disponível em: <https://doi.org/10.5281/zenodo.5135072>. Citado na página 34.

PYODIDE. 2022. Disponível em: <https://pyodide.org/en/stable/development/building-from-sources.html>. Citado na página 34.

PYPL. 2022. Disponível em: <https://pypl.github.io/PYPL.html>. Citado na página 33.

PYTUGUêS. 2015. Disponível em: <https://pythonhosted.org/pytuga/>. Citado 3 vezes nas páginas 19, 25 e 27.

REPLIT. *The collaborative browser based IDE*. 2021. Disponível em: <https://replit.com/>. Citado na página 29.

RUBY, I.; KRSMANOVIC, B. Does learning a programming language require learning english? a comparative analysis between english and programming languages. In: . [S.l.: s.n.], 2017. Citado na página 19.

SCRATCH. 2021. Disponível em: <https://scratch.mit.edu/>. Citado na página 28.

SILVA, G. et al. Impact of calango language in an introductory computer programming course. In: *2020 IEEE Frontiers in Education Conference (FIE)*. [S.l.: s.n.], 2020. p. 1–9. Citado na página 24.

SOMMERVILLE, I. *Software Engineering*. 9. ed. Harlow, England: Addison-Wesley, 2010. ISBN 978-0-13-703515-1. Citado na página 41.