CodeX: ambiente virtual de aprendizagem em programação, integrado à LLM, para auxiliar estudantes com TEA

Vitor Norton[1], **Fabrizio Honda**[1,2], **Marcela Pessoa**[1], **Fernanda Pires**[1]

[1]Escola Superior de Tecnologia – Universidade do Estado do Amazonas (EST-UEA)
ThinkTEd Lab - Pesquisa, Desenvolvimento e Inovação em tecnologias emergentes

[2]Programa de Pós-Graduação em Informática (PPGI)
Instituto de Computação – Universidade Federal do Amazonas (IComp - UFAM)

{vnla.lic18,fpires,msppessoa}@uea.edu.br,

fabrizio.honda@icomp.ufam.edu.br

Abstract. *Considering the challenges faced by students with ASD in universities, particularly in computing courses, which show high dropout and failure rates, this work proposes a Virtual Learning Environment (VLE) named CodeX. The objective is to assist students with ASD in programming content through a tool integrated with a Large Language Model (LLM), which provides a conversion agent to help and offer customizable feedback. Currently, the environment is partially implemented and in the testing phase, with results from its application with ASD students being positive, especially regarding the intelligent tutor, with suggestions to incorporate new features.*

Resumo. *Considerando os desafios de estudantes com TEA nas universidades, sobretudo em computação, que apresenta altos índices de evasão e reprovação, este trabalho propõe um Ambiente Virtual de Aprendizagem (AVA) intitulado CodeX. O objetivo é auxiliar estudantes com TEA em conteúdos de programação, por meio de uma ferramenta integrado à um Large Language Model (LLM), que disponibiliza um agente de conversão para auxiliar e fornecer feedback personalizável. Atualmente o ambiente encontra-se parcialmente implementado e em etapa de testes, cujos resultados ao aplicá-lo com estudantes com TEA foram positivos, principalmente em relação ao tutor inteligente, com sugestões para incorporar novos recursos.*

1. Introdução

O Transtorno do Espectro Autista (TEA) consiste em uma condição do neurodesenvolvimento que afeta as capacidades de socialização, comunicação e aprendizado [Klin 2006], cujas pesquisas recentes apontam um aumento consideravel de pessoas com TEA nos últimos anos, de 2 à 4 casos a cada 10000 habilitantes para 1% da população [Thapar and Rutter 2021]. Portanto, faz-se necessário a adoção de medidas para atender às necessidades desse público, entretanto, no cenário acadêmico, os estudantes com TEA enfrentam obstáculos como: adaptação ao ambiente, gerenciamento de demandas concorrentes e as competências em relação a comunicação e habilidades interpessoais [White et al. 2016, Pinder-Amaker 2014].

Além destes desafios, em se tratando de cursos superiores de computação, os estudantes com TEA também se deparam com as dificuldades características de disciplinas

DOI: 10.5753/sbie.2024.242533

de programação, como: encontrar erros no próprio código, interpretação de texto, pensamento lógico , complexidade dos conteúdos, dentre outros [Bosse 2020]. Esses são alguns dos fatores que ocasionam os altos índices de evasão e reprovação nas disciplinas [Bosse and Gerosa 2015], que não é uma realidade somente no Brasil: a pesquisa de Bennedsen e Carpensen [2019] identificou, a partir da análise de mais de 161 instituições de diversos países, uma taxa de 28% de reprovação em CS1 (*Computer Science 1*) – equivalente à disciplinas introdutórias de programação.

Uma das alternativas para contornar esse cenário pode ser a utilização de Ambientes Virtuais de Aprendizagem (AVA), cujas características de flexibilidade e acessibilidade possibilitam à estudantes de diferentes regiões terem acesso ao mesmo conteúdo [Afrouz and Crisp 2021]. Alguns destes ambientes contam com personalização de conteúdo de acordo com trilhas de aprendizagem .

Incorporados à esses ambientes, a utilização de Large Language Models (LLMs) pode ser uma abordagem capaz de otimizar a geração de conteúdo personalizado e acompanhamento individual de determinadas atividades. Esses modelos executam atividades e geram conteúdo a partir de instruções textuais denominadas *prompts*, cujos recentes avanços possibilitaram-nas realizar tarefas cada vez mais complexas, como reconhecimento de imagens e textos, conhecimento específico sobre uma área de conhecimento, etc [Ge et al. 2024, Touvron et al. 2023, Ma et al. 2023].

Portanto, considerando: (i) as características dos estudantes com TEA; (ii) as dificuldades emergentes de disciplinas de programação; (iii) a utilização de ambientes virtuais de aprendizagem; e (iv) o recente avanço dos LLMs, este trabalho propõe um AVA para auxiliar no processo de aprendizagem de conteúdos de programação para estudantes com Transtorno do Espectro Autista (TEA), integrado à um LLM: um agente de conversação que fornece *feedback* personalizado.

2. Fundamentação Teórica e Trabalhos Relacionados

O transtorno do espectro autista (TEA) é uma condição do neurodesenvolvimento caracterizado principalmente pela deficiência na socialização e no comportamento criativo [Munoz et al. 2018]. No que diz respeito ao âmbito escolar, os estudantes com TEA apresentam dificuldades como falta de apoio escolar, problemas de adaptação ao ambiente escolar, e a baixa adoção de materiais e recursos adaptados. À nível superior, em programação, esses estudantes também enfrentam os desafios característicos dessas disciplinas, que ocasionam as altas taxas de evasão e reprovação mundialmente [Bennedsen and Caspersen 2019]. Portanto, faz-se necessário uma estratégia para auxiliar o processo de aprendizagem desses estudantes, levando em contas suas características e necessidades, realçando a importância de instruções simples e bem estruturadas e a contextualização de problemas de forma explicita [Stuurman et al. 2019].

Uma dessas estratégias pode ser a utilização de Ambientes Virtuais de Aprendizagem (AVA), um ambiente *online* em que professores e estudantes estão separados por tempo e espaço [Rahayu and Wirza 2020]. Nesse ambiente, os conteúdos são disponibilizados através de uma plataforma que suporte conteúdos multimídia, e possibilita aos estudantes consumir aulas de forma síncrona e/ou assíncronas [Annansingh 2019]. Desse modo, identificou-se na literatura alguns trabalhos acadêmicos que propõem Ambientes Virtuais de Aprendizagem (AVA) para auxiliar estudantes com TEA.

A pesquisa conduzida por [Cecil et al. 2017] investiga como os ambientes virtuais de aprendizagem contribuem para a aprendizagem de estudantes com o espectro autista, em disciplinas relacionadas à ciência e engenharia. O trabalho propõe módulos de aprendizagem com tecnologias de *feedback* tátil, visando fomentar a imersão dos estudantes no ambiente. Resultados apontam que os nove participantes do estudo foram capazes de compreender os conceitos abordados nos módulos de aprendizado, enaltecendo esses ambientes como ferramenta para auxiliar estudantes com TEA.

No estudo de [Li et al. 2019], os autores investigam estratégias para facilitar o aprendizado de crianças com TEA, ao utilizar um ambiente virtual imersivo. Fundamentado no modelo de aprendizado experimental, o trabalho visa desenvolver e implementar soluções pedagógicas adaptadas às necessidades específicas de crianças com TEA como listas de tarefas, *feedback* visual e de voz. Esses recursos são projetados para apoiar as crianças ao longo de suas atividades, promovendo um engajamento efetivo e facilitando a interação com cenários de aprendizagem social. Os resultados destacam a importâncias dessas intervenções, evidenciando como elas podem auxiliar os participantes a compreenderem e interagirem efetivamente dentro dos contextos propostos.

Na pesquisa de [Nuguri et al. 2021], criou-se o sistema vSocial, uma plataforma baseada em nuvem destinada à jovens com TEA. O sistema emprega realidade virtual (VR) para criar um ambiente de sala de aula, possibilitando que as interações sociais sejam trabalhadas de forma imersiva e interativa. Dessa forma, a plataforma auxiliará os estudantes na transferência dos comportamentos e habilidades sociais do ambiente virtual para o cotidiano. Experimentos foram conduzidos com sete estudantes universitários, que avaliaram a usabilidade e o desempenho da aplicação. Os resultados foram positivos, indicando que o ambiente é propício para atividades em grupo.

Tabela 1. Comparação de características em ambientes de aprendizado virtual.

Trabalho	Programação	TEA	LLM	Personalização
Nuguri et al. 2021	-	x	-	x
Liu et al. 2019	-	x	-	-
Cecil et al. 2017	-	x	-	x
CodeX	x	x	x	x

Como evidenciado pela Tabela1, os ambientes de aprendizagem virtual apresentam diferentes características e atendem a diferentes necessidades. Apesar de todos abrangerem o público com TEA, nenhum dos trabalhos incorpora o tema de programação em seus ambientes. Desse modo, as vantagens deste trabalho, cujo ambiente denomina-se CodeX, são: (i) um ambiente virtual de aprendizagem para que estudantes com TEA possam (ii) estudar e praticar conteúdos de programação, incluindo (iii) personalização com recursos como avatares, modificações na interface, recursos sonoros e táteis que respondem de forma padronizada as interações com o usuário como a vibração para respostas certas e erradas ou quando o usuário seleciona os componentes de para montar uma resposta, e *feedback* das atividades e estudos através de um (iv) agente de conversação baseado em Large Language Model (LLM).

3. Metodologia

CodeX é um Ambiente Virtual de Aprendizagem (AVA) cujo objetivo é auxiliar estudantes com Transtorno do Espectro Autista (TEA) na aprendizagem de conteúdos de programação. O projeto visa atender às necessidades específicas desses estudantes ao oferecer funcionalidades personalizáveis, como: criação do cronograma de estudos, ajustes na interface (tipografia e papéis de parede), recursos sonoros e táteis, dentre outros. O intuito dessas opções é permitir que o usuário configure a interface do modo que mais lhe agrade, gerando identificação e imersão ao utilizar o ambiente. A metodologia adotada para o desenvolvimento da ferramenta foi o processo criativo de Pires et al. [2021], de natureza interativa-incremental de forma cíclica.

3.1. Identificar um problema

O primeiro passo para a construção do ambiente foi a identificação de um problema real de aprendizagem, que ocorreu no âmbito da disciplina "Design Instrucional", do curso de Licenciatura em Computação da Universidade do Estado do Amazonas (UEA). Nessa disciplina, um dos objetivos é conceber *softwares* educacionais para auxiliar na aprendizagem de conteúdos curriculares, cujo foco estava em acessibilidade. Portanto, em um momento de *brainstorming* com a docente da disciplina, os autores do trabalho elencaram diversos desafios de acessibilidade na universidade, tais como estudantes de computação com TEA que ficam retidos em matérias relacionadas à programação, seja por dificuldade no entendimento dos conteúdos ou interação com docentes, ou com outros estudantes[1]. Além disso, outras dificuldades identificadas podem agravar esses desafios, tais como: comunicação interpessoal, gerenciamento de tempo, *feedbacks* (visual, auditivo, sensorial e imediato), linguagem assertiva, instruções simples, respostas contextualizadas, etc.

3.2. Pesquisar

Após a identificação do problema de aprendizagem, o próximo passo consistiu na realização de pesquisas para analisar o estado da arte em relação à soluções relacionadas à programação para estudantes com TEA, as características desse público e aspectos de acessibilidade. Essa etapa ocorreu em três momentos: (i) análise da literatura científica, (ii) análise de aplicações em programação; e (iii) entrevista com psicóloga.

Em (i), consultou-se a literatura para analisar os aspectos citados anteriormente, aos quais se pôde notar poucos trabalhos relacionados a ambientes virtuais de aprendizagem para pessoas com TEA e *softwares* como ferramenta para auxiliar na aprendizagem ou treinamento para pessoas com TEA. No item (ii), realizou-se pesquisa por aplicações no mercado voltadas para a aprendizagem de programação. O objetivo foi analisar as particularidades desses ambientes, identificando pontos positivos e negativos para agregar ao trabalho. Em relação à (iii), realizou-se uma entrevista semiestruturada com uma especialista da área de psicologia para entender as necessidades e características do público com TEA. A entrevista gerou vários *insights*, principalmente sobre aspectos a serem considerações na solução, tais como a necessidade de textos e instruções simples e diretas, a padronização de comportamento da aplicação e a criação de um vínculo entre o usuário e aplicação, de forma que ele se sinta interessado e representado.

[1] A Universidade do Estado do Amazonas (UEA) conta com três cursos de Computação e em todos se registrou um aumento substancial de ingresso de estudantes com TEA nos últimos 03 anos. Isso tem suscitado muitas discussões quanto a métodos e técnicas que permitam entender e auxiliar no processo de aprendizagem desses estudantes.

3.3. Imaginar e Refletir/Discutir

Na etapa de "Imaginar", baseando-se nas pesquisas e discussões iniciais, foi possível delimitar o escopo do projeto e desenvolver os primeiros esboços da proposta – funcionamento, conteúdos disponíveis, navegabilidade, etc. Em seguida, essa "modelagem mental" foi compartilhada com os demais estudantes da disciplina e a docente responsável, que contribuíram com dicas e sugestões. Neste ponto, considerando o problema de aprendizagem identificado, as pesquisas realizadas e os *feedback* recebidos, o tipo de *software* educacional escolhido foi Ambiente Virtual de Aprendizagem (AVA). Dessa forma, a proposta seria criar um espaço para dar suporte aos estudantes com TEA e auxiliar na aprendizagem de conteúdos de programação, a partir de *feedbacks* personalizados e um agente de conversação. No momento de "Refletir/Discutir", todo o planejamento para a construção do ambiente foi elaborado: a arquitetura da informação, os requisitos, modelagem do sistema, dentre outros. Todos esses aspectos foram documentados, cujas subseções seguintes os descrevem.

3.3.1. Levantamento de requisitos

Após a escolha de AVA como *software* educacional a ser desenvolvido, pôde-se conceber a primeira versão do documento de requisitos do *software*. O objetivo foi descrever as funcionalidades do sistema, visando auxiliar no entendimento do escopo do projeto e elencar as prioridades de desenvolvimento em essencial, importante e desejável. Esse processo seguiu a elicitação dos requisitos, que resultou em dezesseis requisitos funcionais e oito não funcionais. A Tabela 2 apresenta os principais requisitos não-funcionais definidos, representando o núcleo da aplicação.

Tabela 2. Descrição dos Principais Requisitos Funcionais.

ID	Descrição
RF-01	Interação com o agente de conversação
RF-02	Suporte a conteúdo de aprendizagem de algoritmos e estrutura de dados
RF-03	Gerenciamento De cronogramas de estudo
RF-04	Personalização da interface
RF-05	Configuração dos recursos de som
RF-06	Configuração dos recursos táteis

3.3.2. Modelagem

Em seguida, iniciou-se a modelagem do sistema, dividida em (i) casos de uso; (ii) diagramas de atividade; e (iii) diagramas de sequência. Em relação à (i), inicialmente os casos de uso foram descritos de forma resumida, sendo refinados ao longo que o sistema ia sendo concebido. Após a finalização da descrição detalhada, todos os casos de uso foram diagramados, cujos principais são a interação com o tutor inteligente, o gerenciamento do cronograma de estudos e a personalização da interface (exibida na Figura 1).

Posteriormente, no item (ii) foram implementados os diagramas de atividade, com o objetivo de fornecer uma perspectiva detalhada na etapa anterior. Esses diagramas des-

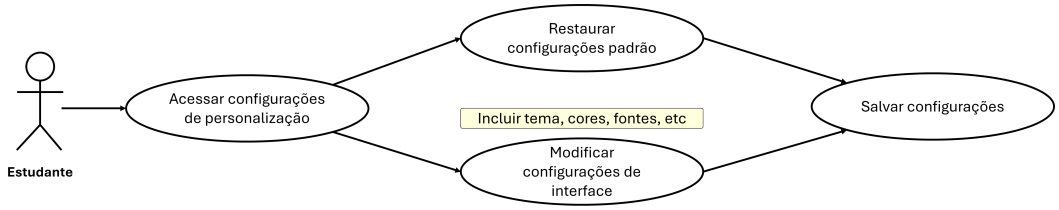


Figura 1. Caso de uso: Personalização da Interface.

crevem os fluxos de processos ou interações dentro do sistema. E em (iii) foram implementados os diagramas de sequência, com o intuito de auxiliar no entendimento de como os atores se relacionam, identificar métodos e objetos na aplicação e delinear os limites de cada sistema que compõe a plataforma. A Figura 2 ilustra o diagrama de sequência de personalização da interface.

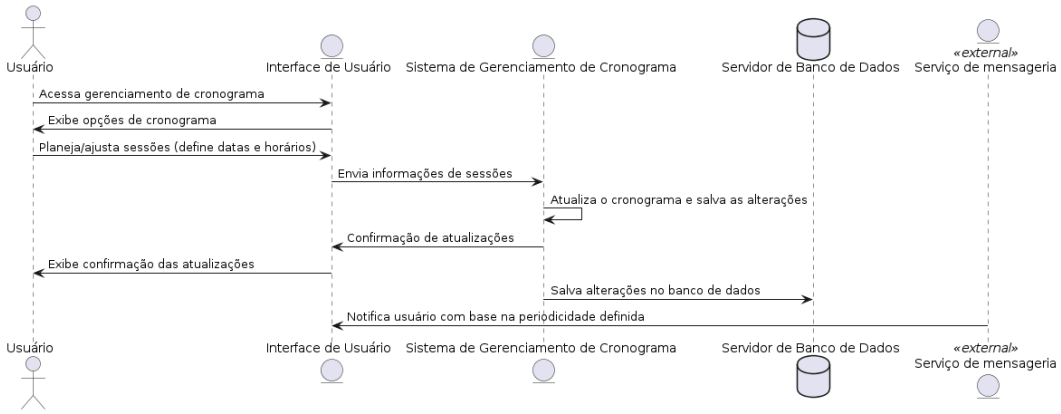


Figura 2. Diagrama de sequência: Personalização da interface.

3.3.3. Arquitetura

A partir da definição das entidades que compõem o sistema e a elaboração dos primeiros fluxos, deu-se início à composição da arquitetura do sistema. Para isso, utilizou-se o modelo C4 [Brown 2013], que consiste em descrever a arquitetura em diferente níveis de abstração: (i) contexto, (ii) contêineres, (iii) componentes e (iv) código. Portanto, o modelo de arquitetura escolhido para o desenvolvimento da aplicação foi a arquitetura em camadas. Assim, cada camada tem responsabilidade e funcionalidades específicas, possibilitando maior autonomia e isolamento das regras de negócio do sistema proposto. Essa estratégia possibilita mudanças mais bruscas nas tecnologias e dependências do sistema, sem que causem um grande impacto ou que demande tempo excessivo em manutenção e implementação de novas funcionalidades. A Figura 3 ilustra a arquitetura estruturada do ambiente CodeX.

O sistema é composto por diferentes aplicações conversam entre si por meio da *Aplication Programing Interface (API)*, tais como a aplicação móvel e a aplicação principal do CodeX, composta pelos módulos de autenticação, módulo de conversação com o tutor virtual, módulo de personalização e módulo de atividades de aprendizagem. O módulo de conversação com tutor virtual comunica-se através da API, com um sistema

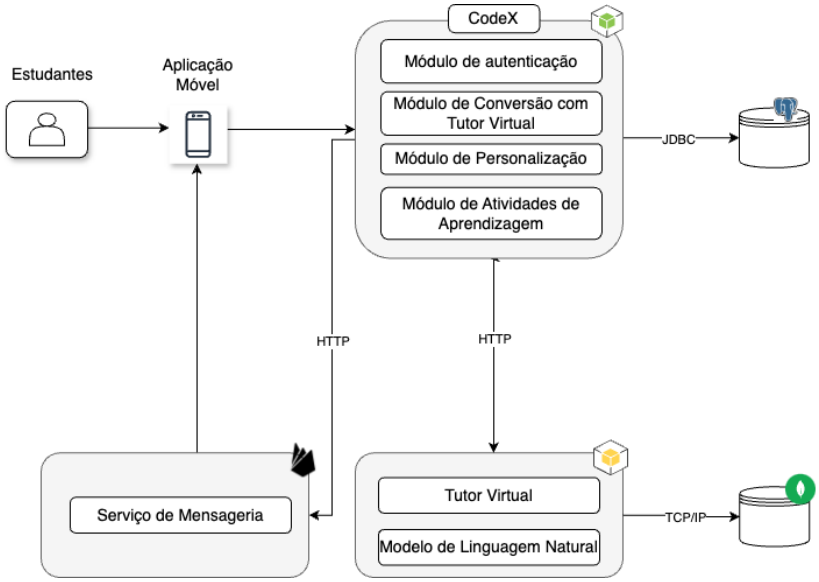


Figura 3. Arquitetura de CodeX.

dedicado para lidar com o modelo de linguagem natural em larga escala e realizar as trativas para uma conversação simples e fluída. O módulo de personalização processa e registra informações do usuário vinculadas às suas preferência e cronograma, também realizando solicitações de envio de notificação para o usuário seguindo o cronograma definido na aplicação.

3.3.4. Tecnologias e Banco de Dados

Em relação às tecnologias, foram empregadas diferentes ferramentas para atender às necessidades de funcionamento e arquitetura do Ambiente Virtual de Aprendizagem. Para implementação da aplicação móvel foi utilizado o *framework React Native*, que permite criar aplicativos móveis nativos, cujos componentes são compilados diretamente para código nativo do sistema operacional. Em relação à aplicação do lado do servidor, optou-se pelo *framework Nestjs*, que possibilita a criação de aplicações *Node.js* de forma escalável e integrada com recursos de desenvolvimento *Web*, como integração com *driver* de banco de dados, sistemas de gerenciamento de filas e *kits* de desenvolvimento de software. Para a implementação do agente de conversação foi tomada a decisão de segregar a aplicação do restante da aplicação. A escolha para divisão entre o agente de conversação integrado com o modelo de LLM e o restante da aplicação do lado do servidor ocorre devido a dois fatores: (i) a quantidade de materiais na literatura disponível para a linguagem *Python* e (ii) a disponibilidade de ferramentas abertas, cujo desenvolvimento é apoiado por uma ampla comunidade. Como recurso empregado para enviar notificações com base no cronograma de atividades do usuário, foi utilizada uma plataforma externa integrada com ambas as partes do sistema.

Baseado na arquitetura, foram desenvolvidos dois bancos de dados distintos e um sistema de armazenamento de arquivos; cada um possui otimização para as diferentes informações e necessidades. O primeiro banco de dados é relacional, o *PostgreSQL*, um Sistema de Gerenciamento de Banco de Dados (SGBD) de código aberto, responsável

por armazenar dados críticos da aplicação como informações de usuários, permissões, progresso nos cursos e cronogramas. Em paralelo, foi utilizado o banco de dados não relacional *MongoDB* para armazenar as mensagens trocadas entre o tutor virtual e os estudantes, bem como gerenciar informações não estruturadas, como os *templates* das atividades. Esse modelo de banco de dados fornece flexibilidade para lidar com grande volume de dados e menor complexidade em alterações em sua estrutura. Outro recurso utilizado foi o sistema de armazenamento de arquivos dedicada através do *MinIO*, utilizando uma versão conhecida como *self-hosted*, que permite ter uma instância da aplicação na máquina desejada.

3.4. Criação

Esta etapa está relacionada à construção de protótipos para validar as funcionalidades mapeadas na etapa anterior e *como e se* a ferramenta atende aos objetivos propostos. Neste trabalho, quatro protótipos foram concebidos: *(i)* de baixa fidelidade, *(ii)* média fidelidade, *(iii)* prova de conceito (POC), e *(iv)* protótipo de alta fidelidade.

Em (i) foram exploradas diferentes formas de interação dentro do sistema, cuja simplicidade desse protótipo permite realizar ajustes de maneira rápida, de forma que a aplicação fique mais consistente e próxima da proposta. Dessa forma, esse protótipo foi consolidado através de desenhos com o papel e caneta, ilustrando os fluxos e principais componentes. No momento (ii), utilizou-se o Figma para a prototipação de média fidelidade. O objetivo foi construir o fluxo de navegação do ambiente, concebendo as interfaces do usuário, para verificar se a proposta estava condizente e adequada ao seu escopo no que diz respeito a usabilidade e acessibilidade. Com isso, as telas foram elaboradas, possibilitando analisar a navegabilidade e, superficialmente, as funcionalidades.

Na fase (iii) foi implementada a Prova de Conceito (*Proof of Concept* - POC) do ambiente, uma versão prévia para verificar as funcionalidades do ambiente proposto. Nesta etapa, o foco foi a construção da conversação com o tutor inteligente, que envolveu: a realização de estudos de modelos de LLM adequados que oferecem recursos para manipulação de código, geração de conteúdos a partir *prompts* adequados para comunicação com estudantes com TEA e a comunicação entre o módulo responsável pelo gerenciamento do agente de conversação e com o restante da aplicação do lado do servidor. A POC foi desenvolvida conforme a arquitetura ilustrada na subseção 3.3.3, dando início à implementação dos módulos de autenticação e do tutor inteligente. A ferramenta utilizada para a implementação geral da aplicação foi o *Node.js*, enquanto a linguagem de programação *Python* aliada ao *framework FastAPI*, foi empregada no módulo do tutor inteligente. Em relação à aplicação móvel responsável pela interação com o usuário, utilizou-se o *React Native*, que possibilita a criação de aplicações móveis híbridas nativamente.

A partir da análise dos pontos de melhoria identificados com os testes via POC, pôde-se iniciar a versão mais robusta: a de (iv) alta fidelidade, que antecede o produto final. Nesta etapaas funcionalidades estão em processo de implementação a partir de análises e correções realizadas nas etapas anteriores.Deste modo, a partir da concepção desta versão, pretende-se realizar testes com estudantes para validá-la e encaminhar o desenvolvimento para o produto final.

3.5. Brincar/Testar

A etapa de "Brincar/Testar" refere-se à validação dos protótipos desenvolvidos na versão anterior. Pela característica do processo ser iterativo-incremental, após a realização dos testes, volta-se para a etapa de "Criar", elaborando um novo protótipo que incorpore os pontos de melhoria e possíveis inconsistências identificadas. No âmbito deste trabalho, à validação da aplicação ocorreu em três momentos: (i) testes de desempenho e integração, (ii) validação interna com os pares (professor/estudantes) da disciplina e (iii) validação com estudantes com TEA.

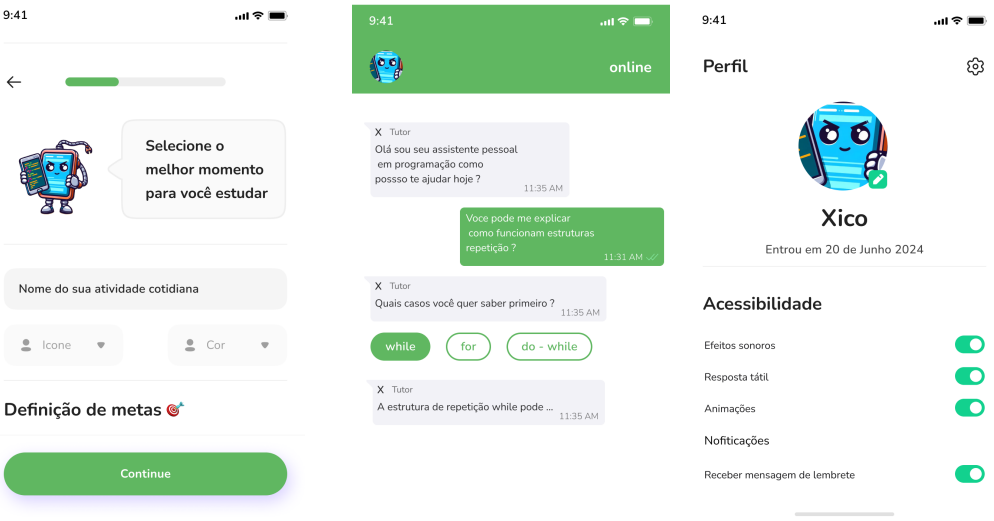
Em (i), testou-se inicialmente a viabilidade da instalação da plataforma em computadores sem unidade de processamento gráfico (GPU) e funcionando a partir da rede privada fechada (sem conexão com internet) a partir da solução GPT4All [Anand et al. 2023]. À priori, a proposta mostrou-se eficiente e acessível, porém ao escalar a aplicação e realizar os testes de desenvolvimento, foi perceptível a perda de performance, gerando problemas de usabilidade do sistema. Com isso, a estratégia adotada para solucionar o problema de performance foi a utilização de uma plataforma dedicada para hospedagem de modelos de LLM integrada à aplicação, que ocasionou em maior performance e expandindo as possibilidade de utilização do modelo no sistema.

Após realizar os testes de desenvolvimento e implantação da aplicação, iniciou-se a etapa (ii), cujos protótipos de baixa e média fidelidade, junto com a POC, foram validadas internamente pela docente da disciplina[2], um especialista em educação em computação e os demais estudantes da classe. A partir das considerações, realizou-se diversos ajustes nos protótipos, tais como: inserção de funcionalidade para personalização da interface, recurso de definição de cronograma, metas definidas pelo usuário e mapeamento de recursos sonoros e táteis. Em (iii), buscou-se realizar uma avaliação com estudantes com TEA, o público-alvo do ambiente, para verificar se o desenvolvimento da aplicação atende, de fato, às suas necessidades. Desse modo, dois estudantes da universidade, ambos com TEA nos níveis 1 e 2 de suporte, foram convidados à participar da avaliação. Foram realizadas entrevistas abertas, passando pelas oito seções da aplicação: (i) Autenticação, (ii) Percurso, (iii) Chat de conversação com tutor, (iv) Revisão, (v) Cronograma, (vi) Perfil, (vii) Loja e (viii) Personalização. Inicialmente, utilizou-se um protótipo de média fidelidade para coletar impressões e opiniões sobre as seções da aplicação. Além disso, foi aplicada a *POC*, visando extrair *insights* a partir da experiência dos testadores com o uso da aplicação e mapear possíveis inconsistências entre a aplicação móvel e a aplicação do lado do servidor. Os resultados podem ser consultados na Seção 4.

4. Resultados e discussões

Dentre os resultados obtidos, destacam-se a concepção do protótipo de alta fidelidade de CodeX, a partir das avaliações com o protótipo de média fidelidade e a *POC*. A partir das pesquisas realizadas durante o trabalho, foi possível desenvolver recursos voltados para estudantes com TEA dentro do ambiente. Esses recursos são descritos a seguir, que incluem: rotina de estudos e notificações, recursos didáticos de programação, lições de programação, tutor virtual, acessibilidade e personalização da interface. Alguns desses recursos são ilustrados na Figura 4 abaixo.

[2]A docente da disciplina trabalha com educação em computação e informática na educação, tendo como área de especialidade *Learning Design* e também tem TEA.



(a) Definição de rotinas. (b) Interação com o agente. (c) Acessibilidade e personalização.

Figura 4. Recursos disponíveis no CodeX.

Rotina de estudos e notificações: incluem a possibilidade do usuário criar sua própria rotina de estudos no ambiente, escolhendo o melhor momento do dia para suas atividades – permitindo uma progressão individual e auxiliando na manutenção de uma rotina específica. O sistema utiliza os parâmetros inseridos pelo usuário (nome da notificação, ícone e cores) para criar notificações personalizadas, que são enviadas somente nos dias escolhidos – respeitando o interesse do estudante e evitando a sobrecarga com informações excessivas.

Recursos didáticos de programação: podem ser organizados em três níveis: iniciante, intermediário e avançado – para as linguagens suportadas na plataforma. Essa estruturação permite que as atividades sejam correspondentes à cada nível, possibilitando que os estudantes exercitem gradualmente os conceitos de programação. Dessa forma, a quantidade de tarefas é menor, facilitando a abstração do conteúdo e proporcionando que os estudantes foquem-se nos aspectos fundamentais da programação.

Lições de programação: divididas em três tipos: (i) complemento de código; (ii) ordenação; e (iii) perguntas e respostas. Essas atividades foram projetadas pensando na inclusão para os estudantes com TEA, visando trazer confiança ao estudante. Além disso, considerou-se instruções passadas de forma clara e concisa, com o objetivo de evitar ambiguidade e minimizar a necessidade de memorização do código. Portanto, esse recurso possibilita que os estudantes foquem nos conceitos de forma mais simples, diminuindo o nível de tarefas e apresentando a sintaxe da linguagem escolhida. Além disso, a aplicação oferece *feedbacks* imediatos para que os estudantes compreendam seus progressos, cujo acompanhamento é registrado para monitorar a trajetória do estudante.

Tutor virtual: *chatbot* especialista em programação de computadores, criado para atender as dificuldades de comunicação frequentemente encontradas por estudantes com TEA. O agente de conversação é um LLM personalizado com interação a partir de instruções do usuário (*prompts*), cujo objetivo não é somente responder aos questionamentos, mas também direcionar a conversa para tratar os temas de forma mais objetiva e con-

sistente. Outra funcionalidade trata-se dos recursos de resposta rápida, que trazem mais objetividade na conversa e direcionam estudantes iniciantes, além de evitar que o usuário precise sempre transcrever as mensagens.

Acessibilidade e personalização da interface: inseridos para promover uma maior identificação e imersão dos estudantes com a aplicação. Esses recursos incluem: gerenciar animações, realizar ajustes nos efeitos sonoros e incorporar *feedback* tátil. Além disso, o ambiente proporciona ao usuário selecionar um avatar ou foto real para representá-lo. Em relação à personalização da interface, possibilita que os estudante com TEA criem um vínculo com o ambiente ao configurar aspectos visuais e sonoros ao seu gosto: fontes, papéis de parede e sons. Esses recursos podem ser desbloqueados com moedas, adquiridas pelos usuários ao realizar as atividades do ambiente.

Os testes realizados com as versões anteriores tinham como objetivo analisar a experiência do público-alvo (estudantes com TEA) ao utilizar a ferramenta, bem como identificar inconsistências para serem corrigidas posteriormente. Como forma de coletar essas informações, utilizou-se entrevistas abertas com os estudantes. Essa estratégia deu-se visando compreender as experiências e necessidades dos entrevistados, bem como a possibilidade de aprofundamento do tema de forma que os estudantes se sentissem livres para responder às perguntas e escolher quais assuntos abordar durante o percurso. Desse modo, dois estudantes com TEA realizaram os testes com o CodeX, acessando as oito seções da aplicação: (i) Autenticação, (ii) Percurso, (iii) Chat de conversação com tutor, (iv) Revisão, (v) Cronograma, (vi) Perfil, (vii) Loja e (viii) Personalização.

Os resultados indicaram que os participantes conseguiram interagir com a aplicação efetivamente por cada seção, contribuindo com pontos de melhoria sobre a usabilidade, bem como sugestões de novas funcionalidades para complementarem os recursos existentes no sistema. Dentre os *feedbacks* para melhoria, destacam-se: (i) Autenticação – a necessidade de incluir novas opções de linguagens de programação durante o cadastro de usuário e opção de inserção escrita sobre motivação de aprender programação; (ii) Percurso – adicionar recurso de recompensa ao progresso e comprimento de metas de atividades; (iii) Chat de conversação com tutor – poder selecionar respostas rápidas sugeridas pelo sistema, diminuir o tamanho das respostas do agente de conversação, adição de histórico de conversação em tópicos e remover a animação de escrita da resposta do tutor; (iv) Revisão – definir estratégias para mapeamento e disponibilização dos materiais de revisão de forma que possibilite utilizar diferentes métodos; (v) Cronograma – utilizar menos textos, inserir diagramas que representem melhor a atividade do usuário no sistema, adicionar imagens ou cores que possibilitem representar os *status* do usuário; (vi) Personalização – permitir que o usuário adicione um papel de parede da galeria após concluir parte do progresso da aplicação.

As validações e testes realizadas com os pares e posteriormente com o público-alvo permitiram analisar o ambiente CodeX. Com base nos resultados obtidos, percebeu-se que a ferramenta pode auxiliar estudantes com TEA em programação, principalmente em relação ao tutor inteligente, que respondeu de forma eficiente aos questionamentos de programação – de cunho teórico e prático – dos testadores. Os principais recursos da aplicação que chamaram atenção dos estudantes foram as funcionalidades voltadas para a experiência de utilização e modificação de recursos da interface. Em contrapartida, os envolvidos apontaram a necessidade de adicionar recursos complementares no processo

de interação com o tutor e melhoria entre o balanceamento entre texto e imagens. Esses pontos irão auxiliar na melhoria da proposta, que serão incorporados na versão posterior.

5. Conclusão

Os estudantes com TEA apresentam desafios no contexto acadêmico, sobretudo em programação, que apresentam dificuldades características como complexidade dos conteúdos, identificação de erros no código, interpretação de texto, etc – resultando em altas taxas de evasão e reprovação. Uma das alternativas pode ser a utilização de Ambientes Virtuais de Aprendizagem (AVAs), onde os estudantes conseguem acessar os conteúdos de forma assíncrona e estudar de acordo com suas preferências. No entanto, poucos trabalhos na literatura foram identificados na intersecção entre estudantes com TEA, AVAs e conteúdos de programação.

Portanto, este trabalho visou preencher essa lacuna através da proposta de um Ambiente Virtual de Aprendizagem (AVA) para auxiliar estudantes com TEA na aprendizagem de conteúdos de programação, denominado CodeX. Uma de suas inovações, além de ser um dos poucos artefatos na literatura nessa intersecção, é a integração com um Large Language Model (LLM), através da criação de um tutor inteligente. A partir dele, os estudantes conseguem tirar dúvidas sobre programação e interagir com um ambiente mais dinâmico e voltado para suas necessidades, recebendo *feedback* personalizado. Para a produção deste artefato, utilizou-se o processo criativo iterativo-incremental de Pires [2021]. Atualmente o ambiente encontra-se em transição da etapa de "Criar" para "Brincar/Testar": seus protótipos anteriores foram avaliados e a versão de alta fidelidade está em produção. O público-alvo, estudantes com TEA, realizaram testes com a *POC* elaborada, explorando todas as funcionalidades do sistema. Os resultados foram positivos, principalmente a respeito da interação com o tutor inteligente. Além disso, os estudantes identificaram pontos de melhoria, que foram catalogados e estão sendo considerados na implementação da versão posterior.

Portanto, como trabalhos futuros, pretende-se: (i) aperfeiçoar o ambiente virtual de aprendizagem através dos apontamentos realizados durante a etapa de testes, de forma cíclica com o público alvo, (ii) incluir gamificação na plataforma para proporcionar motivação e posterior engajamento através de elementos de jogos; (iii) adicionar recursos de aprendizagem por meio de *prompts* de geração de conteúdo; (iv) implementar recursos de notícias sobre programação; (v) empregar o ambiente virtual de aprendizagem juntamente a uma disciplina de programação da universidade e (vi) disponibilizar o ambiente virtual para o público, de forma online e gratuita. Com esses pontos, o objetivo é tornar a ferramenta cada vez mais adequada para os estudantes com TEA, considerando as necessidades desses estudantes, possibilitando a aprendizagem dos conteúdos de programação de forma dinâmica e com suporte personalizado.

Referências

Afrouz, R. and Crisp, B. R. (2021). Online education in social work, effectiveness, benefits, and challenges: A scoping review. *Australian Social Work*, 74(1):55–67.

Anand, Y., Nussbaum, Z., Duderstadt, B., Schmidt, B., and Mulyar, A. (2023). Gpt4all: Training an assistant-style chatbot with large scale data distillation from gpt-3.5-turbo. https://github.com/nomic-ai/gpt4all.

Annansingh, F. (2019). Mind the gap: Cognitive active learning in virtual learning environment perception of instructors and students. *Education and Information Technologies*, 24(6):3669–3688.

Bennedsen, J. and Caspersen, M. E. (2019). Failure rates in introductory programming: 12 years later. *ACM inroads*, 10(2):30–36.

Bosse, Y. (2020). *Padrões de dificuldades relacionadas com o aprendizado de programação*. PhD thesis, Universidade de São Paulo.

Bosse, Y. and Gerosa, M. A. (2015). Reprovações e trancamentos nas disciplinas de introdução à programação da universidade de são paulo: um estudo preliminar. In *Anais do XXIII Workshop sobre Educaçao em Computaçao*, pages 426–435. SBC.

Brown, S. (2013). Software architecture for developers. *Coding the Architecture*.

Cecil, J., Sweet-Darter, M., and Cecil-Xavier, A. (2017). Exploring the use of virtual learning environments to support science learning in autistic students. In *2017 IEEE Frontiers in Education Conference (FIE)*, pages 1–8.

Ge, Y., Hua, W., Mei, K., Tan, J., Xu, S., Li, Z., Zhang, Y., et al. (2024). Openagi: When llm meets domain experts. *Advances in Neural Information Processing Systems*, 36.

Klin, A. (2006). Autismo e síndrome de asperger: uma visão geral. *Brazilian Journal of Psychiatry*, 28:s3–s11.

Li, C., Ip, H. H. S., and Ma, P. K. (2019). A design framework of virtual reality enabled experiential learning for children with autism spectrum disorder. In *Blended Learning: Educational Innovation for Personalized Learning: 12th International Conference, ICBL 2019, Hradec Kralove, Czech Republic, July 2–4, 2019, Proceedings 12*, pages 93–102. Springer.

Ma, X., Fang, G., and Wang, X. (2023). Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

Munoz, R., Villarroel, R., Barcelos, T. S., Riquelme, F., Quezada, A., and Bustos-Valenzuela, P. (2018). Developing computational thinking skills in adolescents with autism spectrum disorder through digital game programming. *IEEE Access*, 6:63880–63889.

Nuguri, S. S., Calyam, P., Oruche, R., Gulhane, A., Valluripally, S., Stichter, J., and He, Z. (2021). vsocial: a cloud-based system for social virtual reality learning environment applications in special education. *Multimedia Tools and Applications*, 80:16827–16856.

Pinder-Amaker, S. (2014). Identifying the unmet needs of college students on the autism spectrum. *Harvard review of psychiatry*, 22(2):125–137.

Pires, F. G. d. S. et al. (2021). Thinkted lab, um caso de aprendizagem criativa em computação no nível superior.

Rahayu, R. P. and Wirza, Y. (2020). Teachers' perception of online learning during pandemic covid-19. *Jurnal penelitian pendidikan*, 20(3):392–406.

Stuurman, S., Passier, H. J., Geven, F., and Barendsen, E. (2019). Autism: Implications for inclusive education with respect to software engineering. In *Proceedings of the 8th Computer Science Education Research Conference*, pages 15–25.

Thapar, A. and Rutter, M. (2021). Genetic advances in autism. *Journal of autism and developmental disorders*, 51:4321–4332.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

White, S. W., Elias, R., Salinas, C. E., Capriola, N., Conner, C. M., Asselin, S. B., Miyazaki, Y., Mazefsky, C. A., Howlin, P., and Getzel, E. E. (2016). Students with autism spectrum disorder in college: Results from a preliminary mixed methods needs analysis. *Research in developmental disabilities*, 56:29–40.