Escola Politécnica da Universidade de São Paulo

Thalles Andrade Estrela Batista

Projeto de um Sistema de Tutoria Inteligente para o Ensino de Física

São Paulo

2021

Thalles Andrade Estrela Batista

Projeto de um Sistema de Tutoria Inteligente para o Ensino de Física

Trabalho de conclusão de curso apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do diploma de Engenheiro Mecatrônico.

Orientador: Prof. Dr. Marcos Pereira Barreto

São Paulo - SP

2021

AGRADECIMENTOS

Gostaria de agradecer aos meus pais por terem me possibilitado chegar até aqui, me educando com valores sólidos e possibilitando cada passo. Nada disso seria possível sem vocês. Também à minha irmã, pelo apoio e pela motivação dada, ser um bom irmão mais velho é uma forte motivação. A minha família tornou tudo muito mais leve e fácil, em todos os sentidos, desde sempre.

Agradeço também à Dany, minha namorada, por todos esses anos de amor, companheirismo e grande compreensão. Não poderia querer uma parceira melhor do que você.

Agradeço aos meus demais familiares e amigos, felizmente são muitos e bons. Parte desta pequena vitória se deve a vocês, de alguma forma.

Também agradeço aos professores do PMR, felizmente a grande maioria de vocês são pessoas simpáticas, grandes pesquisadores e educadores dedicados. Nem sempre é fácil, mas foi bom contar com vocês. Um especial agradecimento ao Prof. Dr. Marcos Ribeiro Pereira Barretto, meu orientador e um grande especialista em empreendedorismo, obrigado por aceitar me orientar e pelo apoio dado neste projeto. E, claro, agradecimentos ao Casio pelas vezes que salvou a minha matrícula e me orientou quando eu estava totalmente perdido (o que aconteceu um número razoável de vezes).

E por último, e mais importante, agradeço à Deus pela vida e por toda a sorte que sempre pareceu me seguir e guiar em todos os lugares que fui.

Thalles Andrade Estrela Batista

RESUMO

Não restam dúvidas de que a educação é um setor de extrema importância, sobretudo no mundo contemporâneo, em que a automação de tarefas braçais e a capacitação constante são tendências que parecem irreversíveis. Mesmo assim, milhões de pessoas ainda vivem à margem do sistema educacional, problema que gera muitos debates e pesquisas nos mais diversos campos. A tecnologia é apontada como uma forte aliada no combate a este problema e os desenvolvimentos de *hardware,* como chips especializados, e *software,* como novas e mais eficientes técnicas de Inteligência Artificial e Aprendizado de Máquina, potencializam cada vez mais a capacidade dos computadores de compreender o meio em que estão inseridos. Uma tendência que tem despontado em artigos e levantamentos mais recentes, é o emprego de técnicas de Inteligência Artificial em sistemas voltados ao ensino, de modo a torná-lo adaptável ao ritmo do estudante, provendo o nível de desafio adequado ao mesmo. Estes sistemas podem ser chamados de Sistema de Tutoria Inteligente. O presente trabalho explicita em detalhes o desenvolvimento de um sistema deste tipo voltado ao ensino de física básica de Ensino Médio, com o uso de estratégias pedagógicas adequadas e permitindo a interação com um professor humano, que poderia usar a aplicação como uma ferramenta de ensino complementar. Foi dado grande foco ao uso de tecnologias de mercado e ao uso de boas arquiteturas de *software* e padrões de projeto.

Palavras chave: Sistemas de Tutoria Inteligente, Educação, Ferramentas Educacionais

ABSTRACT

There is no doubt that education is an extremely important sector, especially in the contemporary world, where the automation of manual tasks and constant training are trends that seem irreversible. Even so, millions of people still live outside the educational system, a problem that generates many debates and research in the most diverse fields. Technology is seen as a strong ally in combating this problem and hardware developments, such as specialized chips, and software, as new and more efficient Artificial Intelligence and Machine Learning techniques, increasingly enhance the ability of computers to understand the environment in which they are inserted. A trend that has emerged in more recent articles and surveys is the use of Artificial Intelligence techniques in systems aimed at teaching, in order to make it adaptable to the student's pace, providing the appropriate level of challenge for him. These systems can be called Intelligent Tutoring Systems. The present work explains in detail the development of such a system aimed at teaching basic physics in high school, with the use of adequate pedagogical strategies and allowing interaction with a human teacher, who could use the application as a complementary teaching tool. Great focus was given to the use of current technologies and the use of correct software architectures and design patterns.

Keywords: Intelligent Tutoring Systems, Education, Educational Tools

LISTA DE FIGURAS

SUMÁRIO

1. INTRODUÇÃO

A educação é um setor primordial na sociedade contemporânea, sendo indissociável do pensamento democrático vigente na maioria dos países do mundo moderno que já atingiram determinado grau de desenvolvimento. O próprio termo cunhado por Milton Santos para descrever o meio no qual os seres humanos se encontram nos dias atuais, o meio Técnico-Científico-Informacional (Maia, L., 2012), evidencia o papel central que o conhecimento tem na forma como vivemos atualmente.

Mesmo com toda a importância dada à instrução no mundo moderno, muitas pessoas ainda vivem à margem da educação, sobretudo nos países menos desenvolvidos, onde a falta de estrutura e recursos pela maior parte das pessoas, atrapalha o processo educacional e a visão do papel da educação em si. Conscientes dessa situação, as Nações Unidas colocaram na Agenda de Desenvolvimento Sustentável para 2030 (*2030 Agenda for Sustainable Development*) a Meta de Desenvolvimento Sustentável 4, que consiste em assegurar condições inclusivas e igualitárias de educação por toda a vida para todos (United Nations, 2019).

A tecnologia surge então como uma forte aliada contra o problema da desigualdade na educação. Hoje dispomos de facilidades que até 10 anos atrás eram consideradas parte da ficção, como serviços de *delivery* e *streaming*, transporte compartilhado, bancos sem agências, com funcionamento 100% digital, equipamentos sempre conectados, como as *smartTVs* e o próprio tablet - vale lembrar que o lançamento do primeiro iPad só foi ocorrer em 2010.

Tendências atuais apontam para um forte desenvolvimento das capacidades que as tecnologias modernas têm de interagir e compreender o meio em que estão inseridas. Novas e mais eficientes técnicas de IA e *machine learning,* potencializadas pela popularização dos *chips* especialistas, tratamento de *big data*, barateamento de infraestruturas de computação em nuvem e conectividade prometida pelo 5G são alguns dos desenvolvimentos nessa direção.

Em (Pedró *et al.,* 2019), são apresentadas tendências de incorporação da Inteligência Artificial (IA) (outro dos campos que ganharam grande visibilidade durante a última década) na educação. No trabalho, a promoção da personalização do estudo é apontada como uma

forte tendência do uso de IA na instrução, com o desenvolvimento de plataformas que se adaptam ao estudante, fornecendo melhor acompanhamento e o nível de dificuldade adequado à cada um. Por outro lado, a falta de melhorias significativas no processo educativo e a falha em prover ferramentas tecnológicas que sejam usadas confortavelmente pelos professores são citados como pontos de fragilidade da maioria das aplicações.

1.1. Motivações e Objetivos

Na educação, o potencial das tecnologias digitais deve ser explorado para promover a inclusão e personalização do processo, com o desenvolvimento de ferramentas que permitam o aprendizado efetivo, individual e independente de pessoas sem acesso à instrução de outro modo. Enquanto que, nas instituições formais de ensino, podem promover uma maior interação com o conteúdo por parte do aluno, ao mesmo tempo em que auxiliam as atividades do dia-a-dia dos próprios professores, como a elaboração e correção de atividades e o acompanhamento dos alunos de modo mais próximo.

Este trabalho consiste no desenvolvimento de um Sistema de Tutoria Inteligente que se adapte ao estudante, com a aplicação de técnicas de modelagem comportamental e pedagógicas, de modo a promover melhoras sensíveis nos seus resultados. Ao mesmo tempo, o sistema é projetado para empoderar o professor com dados mais granulares a respeito dos alunos, permitindo um acompanhamento individual, além de permitir a alteração do fluxo dos conteúdos vistos e atividades realizadas. Assim, o objetivo é empregar a tecnologia como uma ferramenta útil tanto no aprendizado independente, quanto nas instituições de ensino, tendo foco não somente no aluno, mas também no professor e seu cotidiano de trabalho.

1.2. Estrutura do Trabalho

O texto é organizado da seguinte forma: a Seção 2 apresenta a revisão bibliográfica feita acerca de Sistemas de Tutoria Inteligentes e estratégias pedagógicas, a Seção 3 expõe a arquitetura geral do sistema e explica brevemente alguns conceitos importantes. A Seção 4 mostra detalhes técnicos, do *front-end* e do *back-end*. A Seção 5 detalha os módulos do sistema, suas funcionalidades e detalhes da implementação, a Seção 6 apresenta os resultados obtidos e, finalmente, a Seção 6 conclui o trabalho.

2. SISTEMAS DE TUTORIA INTELIGENTES

Esta seção apresenta alguns dos principais temas, técnicas e ferramentas no campo dos Sistemas de Tutoria Inteligentes, tendo em vista a aplicação proposta. Assim, este conteúdo não foi produzido com o objetivo de ser uma revisão sistemática da literatura e sumarizar os desenvolvimentos feitos e tendências de cada linha de pesquisa, já que tal tarefa demandaria muito tempo e esforço, além de trazer muitos conteúdos que sairiam do escopo do trabalho.

Para uma melhor compreensão, esta seção foi dividida nas seguintes subseções: a 2.1 mostra brevemente a definição de um Sistema de Tutoria Inteligente, enquanto 2.2 trata da arquitetura do mesmo. Em 2.3 são apresentadas algumas das estratégias pedagógicas utilizadas. Por fim, a seção 2.4 traz técnicas utilizadas para lidar com o componente emocional no aprendizado, sobretudo no que tange a motivação do aluno em aprender.

2.1. Introdução

O termo Sistemas de Tutoria Inteligentes (*Intelligent Tutoring Systems*, ITS) é mais antigo que costuma se pensar e foi citado pela primeira vez em 1982 de forma a distinguir seu funcionamento do sistemas de instrução auxiliados por computador (*computer-aided instruction systems*, CAIS) (Olesea, C. 2019).

Existem várias definições para os ITS, por exemplo, (Ramesh, Rao, 2012) define que "Um sistema de tutoria inteligente (ITS) é qualquer sistema computacional que provenha instrução customizada ou *feedback* aos estudantes, sem a intervenção de seres humanos, enquanto executam uma tarefa". Uma definição mais genérica, que sintetiza o conceito de ITS de forma precisa, é dada em (Olesea, C. 2019) como o uso de "técnicas de Inteligência Artificial para traçar as necessidades do estudante e responder adequadamente".

O primeiro ITS que se tem notícia, no sentido aqui empregado do termo, é o sistema SCHOLAR de Carbonell, feito 1970 para comunicar informações sobre a geografia da América do Sul (Olesea, C. 2019). Desde então muitos outros têm sido desenvolvidos e empregados nas mais diversas áreas, desde o ensino de habilidades básicas, como a leitura (Pedró *et al.,* 2019), até o ensino da modelagem de sistemas dinâmicos (Wetzel *et al.,* 2017).

Algumas das maiores vantagens dos ITS são:

- **Disponibilidade:** os ITS podem ser utilizados quando não há a possibilidade do ensino feito por outro agente humano (Pedró *et al.,* 2019), condição que pode ser observada em regiões pobres ou por pessoas de renda mais baixa em geral, quando a simples contratação de um professor ou o ingresso em uma instituição de ensino são inviáveis economicamente. Além disso, a praticidade de tais sistemas permite que sejam utilizados em adição às aulas presenciais em diversos contextos, como forma de estudo complementar;
- **Detalhamento:** esses sistemas podem modelar o comportamento e a assimilação do usuário, permitindo que sejam observados sinais mais sutis acerca da assimilação do conhecimento do que é feito usualmente por testes tradicionais (Beyyoudh *et al.,* 2019; Pedró *et al.,* 2019; Wetzel *et al.,* 2017; Hooshyar et al., 2019);
- **Adaptabilidade:** a partir dos modelos do usuário, os ITS podem adaptar tanto o conteúdo exibido, quanto a forma de exibir e a abordagem utilizada, podendo ainda haver a intervenção de um professor humano neste processo (Beyyoudh *et al.*, 2019; Hooshyat *et al.,* 2019). Assim, os ITS podem se adequar a alunos com maior ou menor facilidade com determinado tema, permitindo que o ensino chegue a todos os estudantes de forma mais homogênea (Pedró *et al.,* 2019; Beyyoudh *et al.*, 2019);
- **Aspecto emocional:** ao se utilizar os meios digitais, o engajamento dos estudantes deve aumentar (Beyyoudh *et al.*, 2019). O que só aumenta à medida em que novas tecnologias e conceitos são incorporados ao sistema, como agentes afetivos, processamento de voz, uso de games e toda a ideia de gamificação (Beyyoudh *et al.*, 2019; Johnson, Lester, 2016).

Algumas dax maiores questões ao se desenvolver esse tipo de sistema são a seleção de estratégias pedagógicas, o desenvolvimento de problemas, a modelagem e compreensão do comportamento e aprendizado do usuário, a recomendação de conteúdos, o momento certo e a forma de dar feedback (Olesea, C. 2019).

Alguns autores propõem os ITS como substitutos aos professores humanos, o que pode ser vantajoso quando os mesmos estão inacessíveis, enquanto outros defendem que os professores humanos são insubstituíveis, assim, esses sistemas deveriam ser utilizados de

forma integrada ou complementar ao ensino feitos pelos mesmos (Pedró *et al.,* 2019; Beyyoudh *et al.*, 2019). Outro ponto em favor dos professores humanos é a questão da credibilidade: professores humanos impõe muito mais confiança e respeito aos olhos dos alunos do que um programa de computador (Johnson, Lester, 2016). Assim, muitos têm proposto sistemas utilizados em parceria com professores humanos.

2.2. Arquitetura de um ITS

A nível lógico, um ITS pode ser construído a partir de quatro módulos (Ramesh, Rao, 2012; Beyyoudh, 2019), conforme a Figura 1. Abaixo uma breve descrição de cada um.

- **Interface com o Usuário:** este módulo compreende toda a parte do sistema que faz a comunicação diretamente com o usuário, recebendo informações de mesmo e exibindo respostas adequadas na tela do dispositivo utilizado;
- **Módulo do Estudante:** compreende toda a representação do usuário no sistema, contendo informações a respeito do mesmo, como dados pessoais, preferências, histórico de desempenho, objetivos e estado do conhecimento;
- **Módulo Especialista:** contém e organiza todo o conteúdo a ser transmitido ao estudante;
- **Módulo de Tutoria:** interage com todos os demais módulos, orquestrando o conteúdo dos mesmos para apresentar o conteúdo ao estudante da melhor forma possível, para isso, selecionando o conteúdo adequado, as atividades adequadas, a abordagem adequada, as respostas adequadas às ações do usuário, etc.

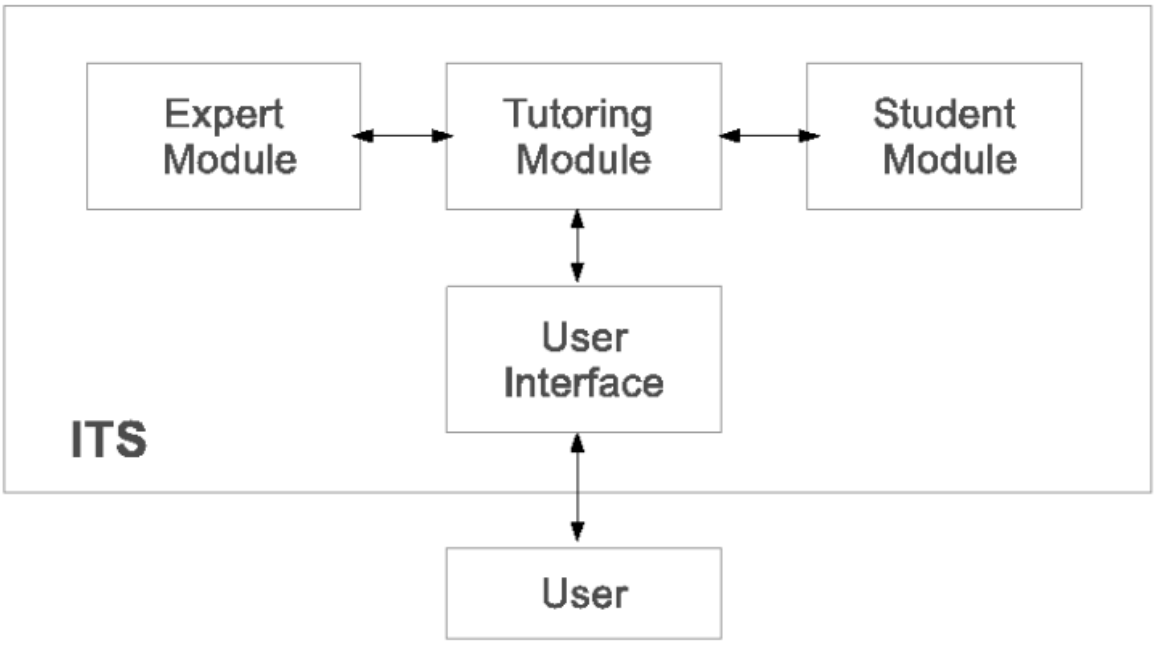

Figura 1: Arquitetura lógica de um ITS, extraído de (Ramesh, Rao, 2012).

Já a nível de implementação, (Wetzel *et al.,* 2017) oferece detalhes de um ITS a ser utilizado como suporte para aulas de modelagem de sistemas dinâmicos[1] para alunos pré-universitários, permitindo que sejam construídos modelos gráficos como resposta para um determinado problema. A solução do aluno é comparada à solução do autor do exercício (*example-tracing*) à cada passo de construção (*step-based*). O programa é implementado como um sistema Web, onde um código Javascript é executado no navegador do usuário, que se comunica com o servidor (Apache Server), este sendo responsável por armazenar os exercícios produzidos e recursos complementares, pela comunicação com o serviço de fórum (phpBB) e com a base de dados relacional (MySQL), além de conter o código Javascript a ser baixado. A base de dados relacional contém informações sobre o estudante o log produzido pelas interações do mesmo com o software. A Figura 2 contém uma representação desta arquitetura.

[1] Os sistemas dinâmicos modelados no Dragoon diferem daqueles comumente tratados em engenharia mecatrônica, tratando, por exemplo, da modelagem de crescimento populacional de uma espécie e as variáveis que podem interferir

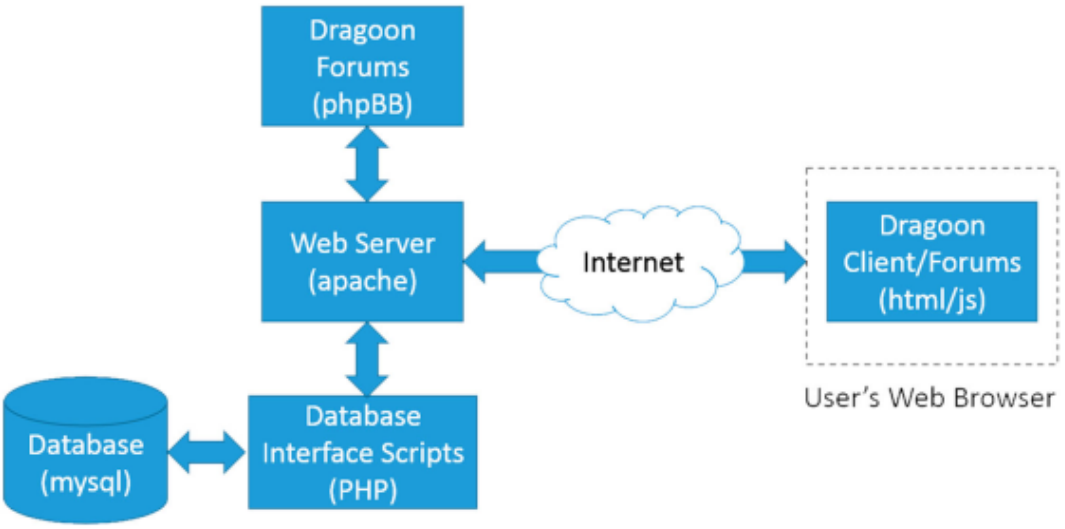

Figura 2: Arquitetura do ITS Dragoon, extraído de (Wetzel *et al.*, 2017).

Comparando a implementação do Dragoon com a arquitetura lógica citada anteriormente, pode-se ver que o navegador executa as funções determinadas pelo módulo de Interface com o Usuário, já a base de dados MySQL contém as informações utilizadas no Módulo do Estudante, enquanto o Web Server, que possui papel central, orquestra todos os recursos a partir das interações com o usuário, implementando então o Módulo de Tutoria, além de conter os exercícios, logo, também implementa o Módulo Especialista.

Em (Beyyoudh *et al.*, 2019) é implementado um ITS com a arquitetura lógica descrita anteriormente com o objetivo de implementar um sistema baseado no uso de games para atividades e avaliações, objetivando uma maior motivação e engajamento, além de extrair informações sobre todo o processo de solução do problema empregado pelo aluno e não somente a solução final. O sistema gera uma sequência de atividades baseada nos conhecimentos do usuário e nos objetivos de aprendizado. Esse sistema é projetado para ser utilizado junto a um acompanhamento por um professor humano, que pode alterar a sequência de atividades, caso julgue necessário, e, com os dados coletados, fazer um acompanhamento mais detalhado e individual do estudante, intervindo caso necessário.

A seguir, maiores detalhes sobre os módulos de um software de ITS.

2.2.1. Módulo do Estudante

O Módulo do Estudante é a representação do mesmo no sistema, contendo todas as informações referentes ao mesmo. É a partir desse módulo que o Módulo de Tutoria tomará as decisões acerca de qual conteúdo apresentar, quais atividades serão feitas e como cada uma destas será personalizada.

Em (Wetzel *et al.,* 2017), o modelo do estudante do software Dragoon é aberto para que o usuário possa acompanhar o seu progresso e desempenho, de modo a mantê-lo motivado. Os professores podem acompanhar o desempenho dos seus alunos em dashboards que contém, entre outras informações, os erros dos seus alunos em cada etapa de cada exercícios e o desempenho geral de cada estudante individualmente, além da agregação desses dados, permitindo uma visão geral da turma como um todo.

(Beyyoudh *et al.*, 2019) divide o Módulo do Estudante em 5 sub-módulos (Figura 3), cada um agrupando determinado tipo de informação.

- **Dados do Estudante:** informações gerais, como nome e idade;
- **Conhecimento do Estudante:** informações sobre os conceitos adquiridos pelo estudante, seu nível de aprendizado e até mesmo cursos feitos e experiência profissionais;
- **Características do Estudante:** informações a respeito de abordagens e técnicas preferidas pelo estudante e o seu estilo de aprendizagem, podendo ser traçadas por técnicas específicas de reconhecimento;
- **Interação com o Sistema:** informações acerca da interação do usuário com o sistema, a partir das quais serão tiradas conclusões a respeito do estado do estudante em si;
- **Estado do Estudante:** informações sobre a situação do usuário, como seu histórico de aprendizagem e os objetivos de aprendizado.

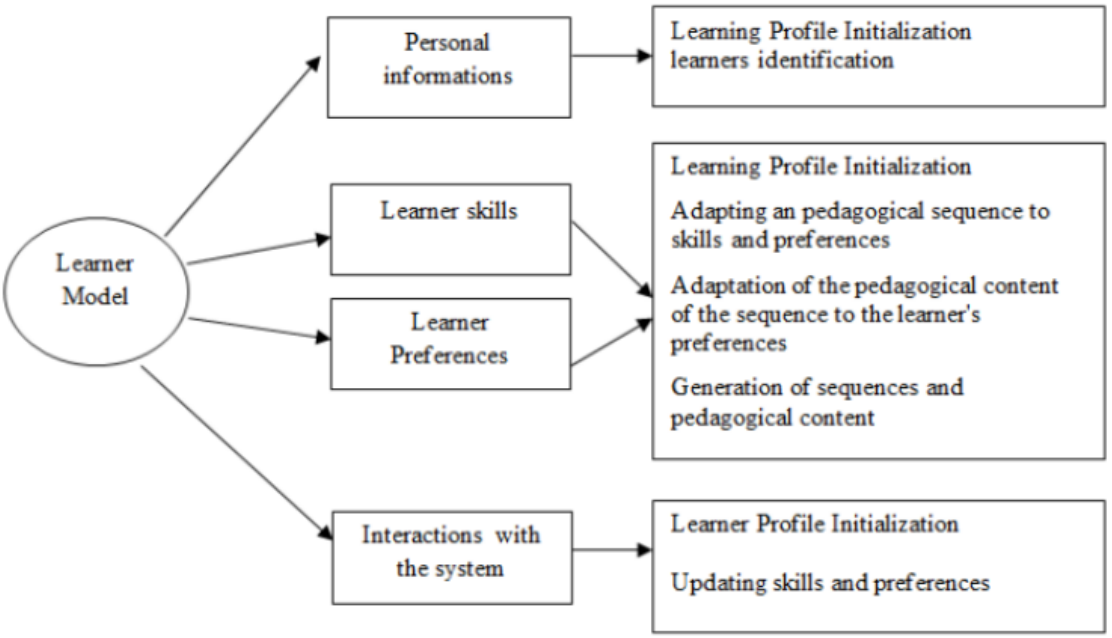

Figura 3: Submódulos do Módulo do Estudante, extraído de (Beyyoudh *et al.*, 2019).

Alternativamente à construção de um modelo de estudante fechado, após cuidadosa análise das informações significativas, para então poder fazer inferências previsões, em (Hooshyar *et al.,* 2019) é feita uma revisão sistemática da literatura (*systematic literature review*, SLR) acerca da modelagem do usuário a partir de dados coletados em sistemas baseados em jogos educativos, ou seja, o modelo não é primeiramente definido e depois populado com os dados, mas sim construído por técnicas de machine learning a partir dos dados disponíveis. Nesse trabalho são mostradas as principais áreas de pesquisa nesse tema e as principais dificuldades desse tipo de abordagem. Além disso, são apresentadas as técnicas mais utilizadas para construir o modelo de usuário a partir dos dados coletados, podendo ser utilizadas técnicas de clustering caso os estados ainda não estejam definidos. As técnicas mais empregadas para este fim foram as *Markov Logic Networks* e os *Hidden Markov Models*, ambas técnicas de aprendizado supervisionado e que permitem que estados de dimensões desconhecidas sejam tratados. As principais vantagens de construir o modelo do estudante desta forma, ainda segundo (Hooshyar *et al.,* 2019), são a ausência da necessidade de conhecimento específico para a construção de um bom modelo de estudante e a possibilidade de insights que não seriam obtidos caso a modelagem tradicional fosse aplicada.

2.2.2. Módulo Especialista

Esse módulo contém todos os conceitos e fatos sobre os tópicos a serem abordados, podendo ser organizados em mapas ou grafos como os Mapas Conceituais e Mapas Conceituais Difusos (Ramesh, Rao, 2012).

Uma possível forma de se implementar este módulo pela separação do mesmo em dois submódulos é citada (Ramesh, Rao, 2012), sendo eles:

- **Módulo Pedagógico:** contém o material a ser ensinado;
- **Módulo Gerador de Problemas:** gera um grande número de exercícios baseando-se em alguns exemplos.

A implementação do Módulo Especialista em (Beyyoudh *et al.*, 2019) também é feita pela separação do mesmo em dois submódulos:

- **Esquema do Domínio:** contém um representação esquemática da Base de Conhecimentos;
- **Base de Conhecimentos:** instâncias individuais do que é apresentado no Esquema do Domínio.

A implementação do Esquema do Domínio é feita pela organização nos seguintes níveis hierárquicos: objetivos de aprendizado, noções, conceitos, atividades e conteúdo. A Figura 4 demonstra esse tipo de organização.

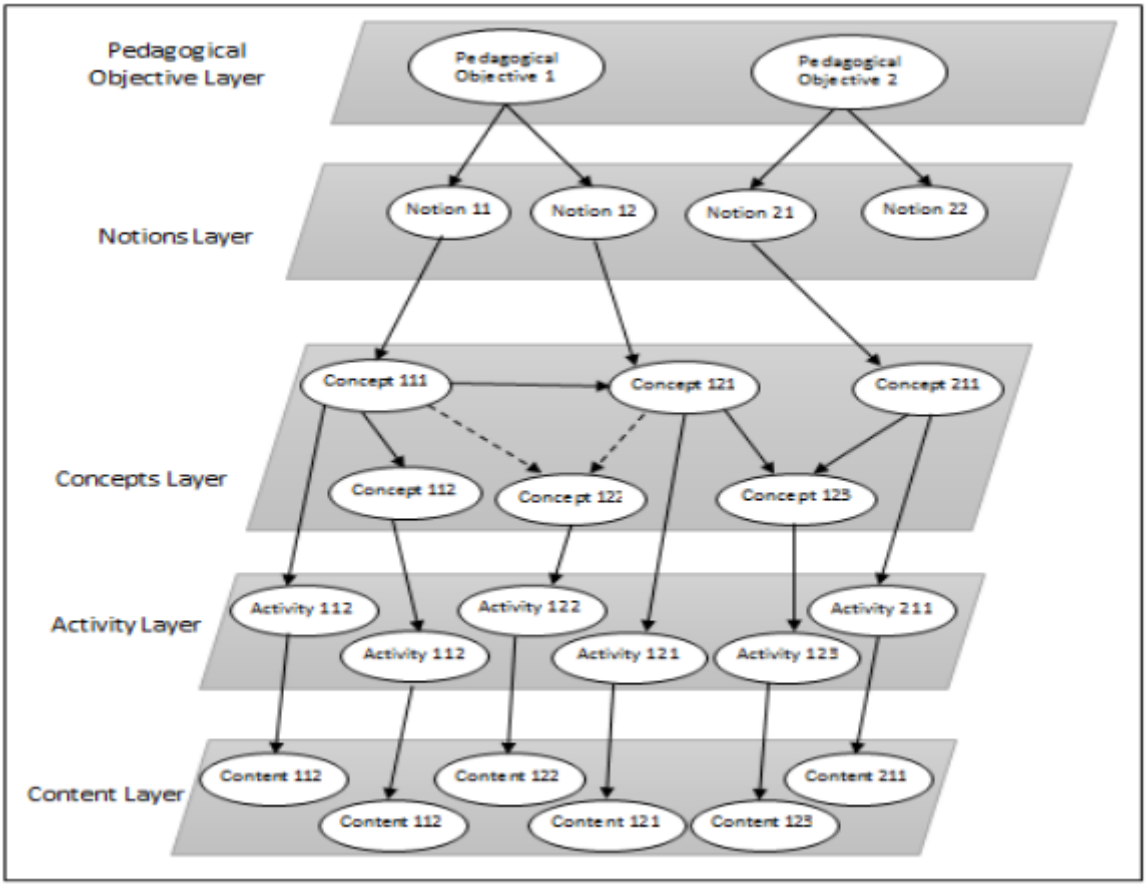

Figura 4: Implementação do Esquema do Domínio, extraído de (Beyyoudh *et al.*, 2019).

A organização do conteúdo também pode ser feita com o uso de ontologias. Conforme (Rybina et al., 2017), essa tarefa é tradicionalmente feita por ontologias de domínio, que visam a modelagem de todo um domínio do conhecimento, porém, nos últimos anos, tem crescido o uso de ontologias aplicadas, feitas para um problema ou aplicação mais específicas, o que impede o seu reuso, porém facilita a tarefa de criá-las. No mesmo trabalho, é apresentado um modelo geral de ontologia para a modelagem genérica de um curso ou disciplina.

2.2.3. Módulo de Tutoria

Este é o módulo que interage com todos os demais para arquitetar a interação com o usuário, ou seja, como o objetivo de aprendizagem será cumprido para determinado estudante (Ramesh, Rao, 2012). Para essa missão, faz-se o uso de diversos recursos, como dicas, explicações conceituais, avaliações, exposição de resultados, etc. Nesse planejamento podem ser levados em conta o histórico do usuário, seus objetivos e até as suas emoções (Ramesh,

Rao, 2012; Beyyoudh, 2019), traçando as atividades e abordagens mais adequadas ao momento em que o estudante se encontra.

Em muitos casos de uso, a interação com o usuário pode ser abstraída em etapas ou passos, por exemplo, respostas a um questionário. O ITS Dragoon (Wetzel *et al.,* 2017) é baseado nos passos da construção do modelo de sistemas dinâmicos (step-based), nele são implementados dois loops, um interno e outro externo, para desempenhar as seguintes funções:

- **Loop interno:** executado uma vez a cada passo, analisa o que foi feito pelo estudante nesse passo e, seguindo a Política Pedagógica, decide que tipo de *feedback* dar;
- **Loop externo:** executado no final de cada tarefa, analisa o desempenho geral do estudante e recomenda alguma atividade para o mesmo, objetivando aumentar a motivação.

Além disso, o Dragoon é baseado e exemplos (*example-tracing* ITS), o que significa que inicialmente um problema é construído a partir de um modelo assumido como certo, então a solução dada pelos estudantes vai ser comparada a este exemplo a cada passo. Segundo (Ramesh, Rao, 2012), ITS baseados em exemplos traçam um grafo acíclico que representa os múltiplos modos de se resolver um dado problema, o que faz surgir a necessidade de limitar o número de escolhas possíveis para o usuário

Em (Wetzel *et al.,* 2017), também é apresentado o conceito de Política Pedagógica, cuja utilidade é decidir o que fazer para cada ação do usuário, sobretudo os erros, já que, em geral, acertos não requerem tratamento especial. O Dragoon conta com uma tabela onde constam as ações a serem tomadas para cada evento, de acordo com o modo selecionado, que pode ser: sem *feedback* (especialmente úteis para provas), *feedback* atrasado, *feedback* imediato, treinamento.

Já em (Johnson, Lester, 2016), é feito uma revisão da literatura referente ao uso de Agentes Pedagógicos, que são atores do processo de aprendizado colocados no mesmo ambiente que o estudante para enriquecer a experiência, aumentando a interação e a troca de emoções entre o estudante e o ITS. Mais detalhes sobre esse tipo de abordagem serão dados na Seção 2.4.

2.3. Estratégias Pedagógicas

As Estratégias Pedagógicas são utilizadas pelo Módulo de Tutoria para que a interação com cada estudante em particular atinja os melhores resultados na transmissão do conteúdo, a seguir, algumas estratégias que podem ser utilizadas.

2.3.1. Personal Learning Pathway

Em (Olesea, C. 2019) é discutida a técnica de *Personal Learning Pathway* (PLP), definida como a sequência de passos intermediários, composta por atividades teóricas e exercícios, para se atingir um determinado objetivo ou habilidade, sendo adaptada a cada usuário individualmente. Nesse trabalho, é abordada também a questão da retenção do conteúdo, modelada pela curva de esquecimento de Ebbinghaus, desenvolvida pelo psicólogo alemão Hermann Ebbinghaus em 1885, e que relaciona a retenção do conteúdo com o tempo, conforme pode ser visto na Figura 5.

Figura 5: Curva de Esquecimento de Ebbinghaus, extraído de (Olesea, C. 2019).

Ainda segundo (Olesea, C. 2019), muitas pesquisas e modelos foram desenvolvidas objetivando evitar o esquecimento, indicando que, ao nos testarmos, aumentamos o tempo de retenção do conteúdo e o passamos gradualmente para a memória de longo prazo. Assim, o trabalho implementou um ITS para alunos de educação infantil reforçarem os conhecimentos adquiridos nas aulas de geometria, aplicando exercícios de modo espaçado, que poderiam ser

repetidos conforme os erros, e que estimulariam o estudante de modos distintos, baseando-se na hipótese de que desta forma o conteúdo seria retido por mais tempo.

(Beyyoudh *et al.*, 2019) implementa o PLP como uma sequência de atividades construída com base no estado de conhecimento do usuário e os objetivos de aprendizado, conforme o Esquema do Domínio (vide Seção 2.2.2). O PLP é acessível pelo professor a todo momento, que pode intervir na sequência das atividades sempre que julgar necessário.

2.3.2. Uso de Games

Uma das várias formas de abordar o conteúdo que podem ser utilizadas em ITS são os *games*, ou, como definido em (Beyyoudh *et al.*, 2019), "*games* sérios", que podem aqui ser definidos como jogos, de quaisquer gêneros, sendo usados para fins educativos. Aqui eles serão chamados simplesmente de *games* ou jogos. O uso de jogos na educação traz os seguintes benefícios (Beyyoudh, 2019; Hooshyar *et al.,* 2019):

- **Motivação:** vários testes indicam que os jogos podem aumentar o estado de motivação dos estudantes, principalmente pelos *feedbacks* dados;
- **Adaptação aos diferentes ritmos:** nos jogos, os estudantes podem prosseguir ao seu próprio passo;
- **Aprendizado por tentativa e erro:** nos *games,* os estudantes podem fazer as suas próprias hipóteses e testá-las, aprendendo no processo;
- **Interação entre estudantes:** algumas plataformas estimulam a interação e cooperação entre estudantes.

(Beyyoudh *et al.*, 2019) defende o uso de jogos nas avaliações, apontando que, em comparação ao método de avaliação tradicional, baseado em provas, os jogos aumentam o interesse e a motivação do aluno, que fica desestimulado ao estudar apenas para realizar exames, além de conter dados mais detalhados acerca da interação do usuário, permitindo ir além de simplesmente o resultado final e identificar os passos intermediários que levaram à solução, possibilitando uma compreensão muito melhor do raciocínio do mesmo.

A autonomia do estudante foi apontada como um ponto favorável à adoção de *games* na educação, já que permite que o estudante prossiga ao seu próprio ritmo. Porém este ponto

também representa um dos maiores desafios, já que o estudante pode ficar repetindo tarefas ou passar pelas mesmos sem um entendimento adequado, ou ainda, ele pode ficar bloqueado em algum ponto, sem conseguir prosseguir. Todos esses cenários fazem com que o conteúdo não seja transmitido satisfatoriamente, além de levar à consequente frustração por parte do usuário.

Em (Beyyoudh *et al.*, 2019), essas situações são tratadas pela intervenção dos professores no processo de aprendizagem, os quais podem auxiliar os alunos em situações de bloqueio e avaliar o seu aprendizado por outros meios. Todavia, em situações onde não há um professor físico, uma alternativa foi proposta em (Hooshyar *et al.,* 2019), que consiste em adaptar a dificuldade dos jogos dinamicamente (*dynamic difficulty adaptation*), onde são utilizadas técnicas de Geração de Conteúdo Procedural (*procedural content generation*, PCG), onde o conteúdo do jogo, em certa medida, é gerado automaticamente, adaptando-se ao usuário e o seu desempenho, sem a necessidade de ser programado previamente. O PCG baseia-se em técnicas de machine learning, sobretudo algoritmos genéticos e Redes Neurais.

2.3.3. Avaliações

As avaliações são parte crucial do processo de aprendizado, pois somente sabendo o estado de aprendizado de um aluno é que se pode decidir os próximos conceitos e atividades a serem executadas. Em (Beyyoudh *et al.*, 2019), as avaliações são classificadas em implícitas, onde são coletados dados da interação do usuário, que não necessariamente está consciente da avaliação, e explícitas, onde o usuário passa por testes avaliativos mais claros. Adicionalmente, o trabalho classifica as avaliações quanto à finalidade, podendo ser de três tipos:

- **Avaliação Diagnóstica:** o objetivo deste tipo é coletar dados para definir o estado do estudante antes de iniciar algum processo de aprendizagem;
- **Avaliação Formativa:** é tipo das avaliações feitas durante o processo de aprendizagem em si, de modo a adaptar as abordagens e conteúdos melhor, além de identificar as falhas no processo de aprendizado;
- **Avaliação Sumativa:** feita ao final do processo de aprendizagem para verificar se o estudante assimilou satisfatoriamente os principais conceitos.

2.4. Componente Afetivo

O aspecto emocional é muito importante durante o aprendizado. Para que todo o processo de transmissão de conhecimento atinja os resultados desejados é muito importante que o estudante esteja engajado e motivado com o aprendizado, o que foi o objetivo de diversos artigos (Beyyoudh, 2019; Johnson, Lester, 2016; Wetzel *et al.,* 2017)

Uma das formas de melhorar o aspecto emocional dos ITS é o uso de Agentes Pedagógicos, amplamente discutidos em (Johnson, Lester, 2016). Nesse trabalho, são apontadas várias potencialidades do uso dos mesmos, dentre as quais:

- **Demonstrações interativas:** os agentes podem demonstrar como utilizar determinada ferramenta ou como determinado processo é feito, dentro de um ambiente virtual;
- **Expressar e provocar emoções e empatia:** a demonstração e sentimentos por agentes animados pode ser feita pelos agentes, o que pode ajudar na transmissão das emoções desejadas aos usuários;
- **Dar *feedbacks* não-verbais:** ao, por exemplo, balançar a cabeça, o Agente Pedagógico dá um *feedback* ao estudante de uma forma diferente de uma caixa de texto. Esse exemplo mostra que esses agentes aumentam as possibilidades de interação com o aluno.

Tornar a interação mais adaptativa ao usuário: um agente pode responder a várias ações tomadas pelo usuário, tornando o ambiente mais responsivo ao mesmo. O artigo também cita alguns dos papéis que os Agentes Pedagógicos podem desempenhar num ambiente de ITS, sendo eles:

- **Virtual Coach:** representa o professor, ou tutor;
- **Agente Ensinável:** são agentes projetados para serem ensinados, fazendo uso do efeito da explicação do conteúdo para alguém afim de que o próprio estudante torne o pensamento mais claro e, consequentemente, fixe melhor o conhecimento;
- **Companheiros de Aprendizado:** simulam outro estudante que acompanha o usuário durante o aprendizado;
- **Atores Virtuais**: são agentes utilizados em simulações.

O trabalho também apresenta algumas conclusões interessantes acerca do uso de Agentes Pedagógicos em ITS, como o fato de produzirem melhores resultados quando aplicados à pessoas mais jovens, o que provavelmente se deve em parte ao maior apego à tecnologia, como também ao maior nível de maturidade e auto-motivação de pessoas mais experientes. Também, foi observado uma maior eficácia quando os agentes se comunicavam por voz invés de por texto e com uma comunicação educada porém não excessivamente forma, o que indica o grande potencial que as tecnologias de Linguagem Natural apresentam quando integradas à abordagem baseada em Agentes Pedagógicos.

3. ARQUITETURA GERAL DO PROJETO

O desenvolvimento do projeto é fortemente baseado na arquitetura tradicional de um ITS, como descrito na seção 2.2. Há, no entanto, o acréscimo de um novo módulo, responsável pelas funções relacionadas ao professor humano, tal componente é chamado de Módulo do Professor. A arquitetura completa do sistema pode ser vista na Figura 6.

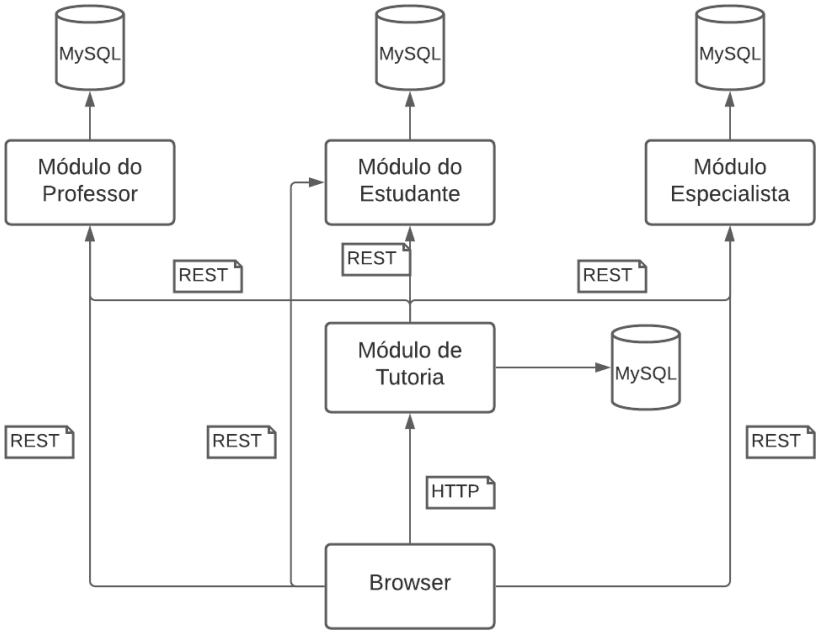

Figura 6: Arquitetura geral do sistema (autoria própria).

Houve, neste projeto, um grande foco para que ele fosse feito de modo a estar o mais próximo possível de um produto lançável no mercado, o que implica em utilizar ferramentas e *frameworks* adequados (melhores descritos na Seção 4), além de padrões de projeto e boas práticas. Neste contexto, uma arquitetura de projeto que tem sido cada vez mais utilizada em empresas de tecnologia é a chamada Arquitetura de Microsserviços. Desta forma, o ITS desenvolvido busca implementar os módulos Especialista, do Estudante e do Professor como RESTful *webservices*, que encapsulam determinadas funcionalidades.

Vale ressaltar que o desenvolvimento de uma boa arquitetura de *software*, baseada ou não em microsserviços, e a construção de RESTful *webservices*, dependem de muito conhecimento teórico e experiência prática, sendo feitas adequadamente apenas por

desenvolvedores seniores competentes. O sistema desenvolvido para este trabalho não tem a ambição de seguir à risca os padrões citados e muito menos impor algum modelo de arquitetura. A intenção do autor foi apenas de construir um projeto que siga aproximadamente as ideias por trás destes padrões.

As subseções 3.1 e 3.2 definem, respectivamente, os conceitos de arquitetura de microsserviços e REST. O detalhamento é dado apenas em nível suficiente para a compreensão deste trabalho, pois trata-se de conceitos complexos, cuja discussão foge do escopo deste texto. A subseção 3.3. mostra os principais casos de uso da aplicação.

3.1. A Arquitetura de Microsserviços

A ideia básica por trás da arquitetura de microsserviços se origina do padrão SOA (*service-oriented architecture*) e consiste da quebra de uma aplicação grande e complexa em componentes melhores que provém determinadas funcionalidades (Redhat).

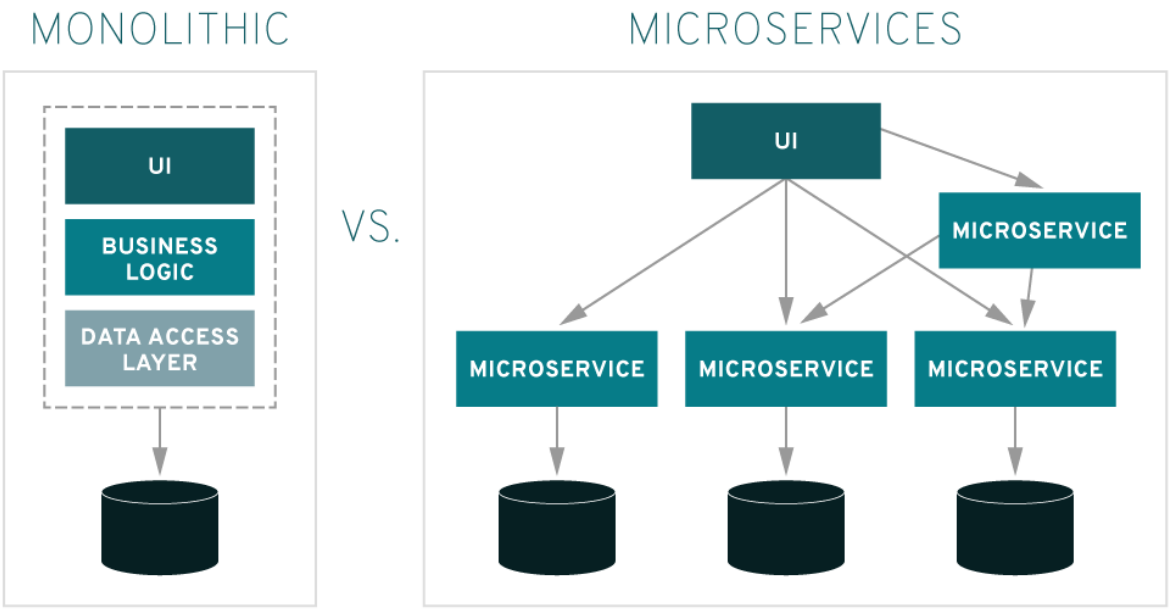

Figura 7: Comparação de uma arquitetura de microsserviços e um monolito, extraído de (RedHat).

Tradicionalmente, todo o código de uma aplicação era contido em uma única unidade (monolito), ou seja, num mesmo projeto coexistiam os códigos-fonte de diversas funções.

Essa abordagem gera um código bastante acoplado, o que traz as seguintes desvantagens principais (Redhat):

- Propagação de erros entre funções;
- Dificuldade na compreensão e, consequentemente, na manutenção;
- Baixa resiliência, pois um erro gerado em uma função gera falha na aplicação inteira;
- Se uma funcionalidade é mais exigida do que as demais, gerando um gargalo no desempenho, deve-se implementar mais servidores da aplicação inteira.

De forma resumida, monolitos tendem a não escalar muito bem, o que tem tornado comum a prática de desmembrar a aplicação principal em microsserviços, procedimento chamado de "quebra de monolito". Não apenas pensando na escalabilidade da aplicação, as diferenças fundamentais entre os códigos das unidades lógicas do ITS (módulos), tornaram evidente que se tratavam de aplicações distintas, o que motivou mais ainda a decomposição do mesmo em microsserviços.

3.2. RESTful Webservices

REST, acrônimo para *Representational State Transfer*, é um modelo de arquitetura para sistemas distribuídos criado no ano 2000 por Roy Fielding, um dos criadores do protocolo HTTP (IBM Cloud Education). Um RESTful *webservice* é um serviço web que implementa as ideias deste modelo.

Os conceitos REST se baseiam no uso do protocolo HTTP e possuem grande flexibilidade, os principais são (IBM Cloud Education):

- **Recursos:** são as entidades gerenciadas por uma aplicação, por exemplo, o conteúdo gerenciado pelo Módulo Especialista;
- **Identificadores de recursos:** são as URI's utilizadas para identificar determinadas entidades, permitindo assim, que um consumidor da aplicação possa identificar para a mesma o recurso que deseja consumir;
- **Representação de recursos:** os recursos em si não podem ser transferidos entre aplicações, por isso, o REST fala da transferência e manipulação de representações do

estado de recursos. Por exemplo: um arquivo JSON que contém as informações básicas de um estudante, como nome e email.

O REST também possui alguns princípios, como (IBM Cloud Education):

- Arquitetura cliente-servidor;
- Comunicação *stateless* (não são armazenadas informações da sessão do usuário);
- Interface uniforme entre cliente e servidor, com o uso de recursos identificáveis (por URI's), cuja manipulação é feita através de representações de estado com o uso da semântica dos verbos e códigos HTTP.

3.3. Casos de Uso

O sistema desenvolvido busca servir a dois atores principais: estudantes e professores. O estudante é aquele que utilizará o sistema como um facilitador de aprendizado enquanto o professor o utiliza como uma ferramenta pedagógica, podendo criar turmas de alunos para o aprendizado de determinado tema e acompanhar o progresso dos alunos, bem como alterar o fluxo de aprendizado, caso julgue necessário. Vale ressaltar que um aluno não precisa estar vinculado à uma turma para utilizar a plataforma. O diagrama de casos de uso da Figura 8 ilustra as principais funcionalidades do sistema.

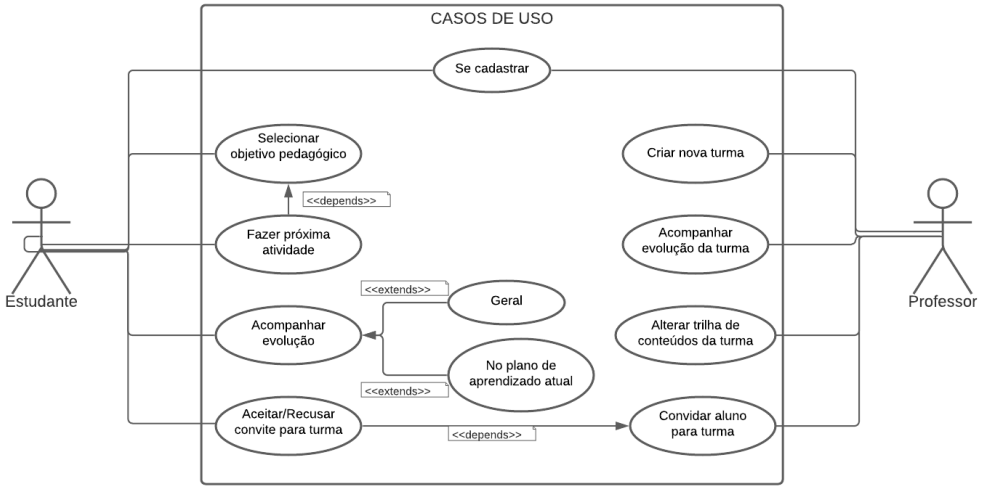

Figura 8: Diagrama de casos de uso do sistema (autoria própria).

4. DETALHES TÉCNICOS GERAIS

Esta seção apresenta as principais considerações técnicas gerais, do ponto de vista de desenvolvimento de *software*, acerca do projeto executado. Para uma melhor compreensão, esta seção foi subdividida em *back-end* (seção 4.1) e *front-end* (seção 4.2).

4.1. Back-end

O *back-end* da aplicação compreende a totalidade dos Módulos Especialista, do Estudante e do Professor, que são RESTful *webservices*, e parte do Módulo de Tutoria. Para cada RESTful *webservice* foi também desenvolvida uma biblioteca (API, *Application Programming Interface*) correspondente, para permitir o uso do mesmo.

A linguagem utilizada foi o Java, devido à maior familiaridade do autor, na versão 11, embora sejam apenas utilizados recursos presentes a partir da versão 9, como os módulos, streams e lambdas. Para facilitar o gerenciamento de dependências do projeto, a execução de testes e o build (geração do projeto final) foi utilizado o Apache Maven. Cada módulo possui seu banco de dados MySQL próprio, acessado somente por ele.

Todos os módulos foram desenvolvidos dentro do Spring Framework, talvez o mais completo, poderoso, robusto e utilizado do mundo Java. A razão para utilizar um framework tão grande para o projeto foi obter o amplo suporte das funcionalidades do mesmo, bem como deixar o projeto mais preparado para um ambiente de uso real. Os componentes do Spring utilizados foram:

- **Spring Core:** fornece todas as funcionalidades básicas, como um container de injeção de dependência;
- **Spring Boot:** facilita muito a configuração do projeto como um todo e a execução em si, pois dispensa a necessidade de gerar um arquivo .WAR e fazer o *deploy* em um servidor. Com o Spring Boot, é gerado um arquivo .JAR, executado normalmente, que contém internamente um servidor (Apache Tomcat, por padrão);
- **Spring Web:** contém o Spring MVC e provê várias funcionalidades básicas e facilidades para se trabalhar na web, como classes que facilitam a manipulação de requisição e resposta e um *entry point* robusto. Há módulos do Spring mais adequados

para aplicações maiores, que permitem o trabalho assíncrono e reativo, porém a complexidade maior das mesmas levou a sua não adoção neste momento do projeto.

- **Spring Data:** é um *umbrella* de projetos do Spring para o acesso à fontes de dados, abstraídas dentro da aplicação como repositórios. Neste projeto foi utilizado Spring Data JPA, que utiliza o Hibernate como implementação da especificação JPA (*Java persistence API*) para acessar bases de dados relacionais.
- **Spring Security:** módulo spring para a parte de autenticação e segurança.

Um projeto Java é dividido em pacotes que contém as suas classes (algo muito semelhante a diretórios simples). Neste projeto, foi seguida a estrutura de pacotes mais comum para projetos *web*, mais especificamente:

- **Model:** classes referentes ao modelo;
- **Repository:** classes que fazem acesso ao banco de dados, fazendo a interface entre o modelo e o banco. Repository é o padrão vindo do Spring Data, em outros projetos é comum que o uso de DAO's (*Data Access Objects*);
- **Controller:** classes responsáveis por gerir o fluxo, deste o recebimento de requisições do cliente, passando pelos processamentos intermediários, até a devolução da resposta;
- **Service:** muitos defendem que as classes controllers devem ter a única responsabilidade de gerir o fluxo, não devendo fazer o acesso às bases de dados ou processamentos relacionados às regras de negócio. Assim, as classes de serviço deveriam ser utilizadas pelos controller para desempenhar tais papéis;
- **Dto:** acrônimo para *Data Transfer Object* (chamados por alguns de VO, *Value Object*), são classes próprias para a transferência de dados. Assim, dados entrando ou saindo da aplicação são colocados em DTO's, o que também auxilia a não expor o modelo diretamente e sim apenas o necessário e de modo controlado;
- **Config:** contém classes de configuração.

4.2. Front-end

O Módulo de Tutoria também é o responsável por fornecer o *front-end* da aplicação, sendo uma aplicação MVC clássica que recebe uma requisição do cliente (*browser*) e devolve uma página como resposta. Vale pontuar que o autor do projeto, no momento da sua implementação, possuía um conhecimento bastante limitado das tecnologias de *front-end*, o que impediu a construção de uma arquitetura de *software* mais bem acabada e diminuiu as possibilidades criativas da exibição.

A estrutura de pacotes é, então, semelhante ao *back-end*, acrescida apenas de um diretório com os arquivos HTML com Javascript. No entanto, num projeto de *front-end* completo seriam acrescidos diretórios referentes ao código e libs Javascript e ao CSS. Também, o Módulo de Tutoria não possui um modelo propriamente dito, pois as entidades manipuladas são gerenciadas pelos *webservices*.

Foi utilizado o *stack* tradicional de *front-end*, composto por HTML, CSS e Javascript, bem como as seguintes ferramentas para facilitar o desenvolvimento:

- **Thymeleaf:** faz a renderização *server-side* da página HTML, muitas vezes complementada com o Javascript posteriormente. Possui boa integração com o Spring;
- **Bootstrap:** utilizado para facilitar a estilização da página e para permitir a construção de páginas responsivas;
- **VueJS:** para facilitar o desenvolvimento Javascript;
- **Axios:** para fazer requisições aos *webservices* pelo Javascript.

O Módulo de Tutoria possui apenas uma base de dados que é utilizada para fazer o cadastro e registro de usuários. Essa base, que também é MySQL, possui apenas uma tabela contendo informações relacionadas ao registro, como username e senha, além do tipo deste usuário (STUDENT ou TEACHER) e o id no módulo correspondente.

5. DETALHAMENTO DOS MÓDULOS

Nesta seção serão melhor expostas as ideias, o projeto e a implementação por trás de cada um dos módulos da aplicação. Como citado anteriormente, a aplicação segue a ideia da arquitetura de microsserviços e é dividida em 4 módulos, que seguem a estrutura descrita na Seção 4, assim, o que diferencia realmente cada módulo é o modelo de cada um e alguma funcionalidade especial, geralmente contida no pacote de *services*. Desta forma, esta seção detalha as responsabilidades de cada módulo, seus modelos e funcionalidades especiais. O Módulo do Estudante é tratado na subseção 5.1, enquanto o Módulo do Professor é tema da subseção 5.2, a subseção 5.3 se dedica ao Módulo Especialista e, finalmente, o Módulo de Tutoria é melhor exposto na subseção 5.4.

O repositório do projeto é público e pode ser acessado em: https://github.com/ThallesBatista/tutoring-system

5.1. Módulo do Estudante

Este módulo possui as seguintes responsabilidades (relacionadas ao estudante):

- Gerenciar informações pessoais;
- Armazenar as interações com o sistema;
- Inferir e gerenciar as preferências;
- Gerenciar o estado de conhecimento;
- Gerenciar os planos de aprendizado;

As principais classes envolvidas na modelagem do estudante podem ser vistas na Figura 9.

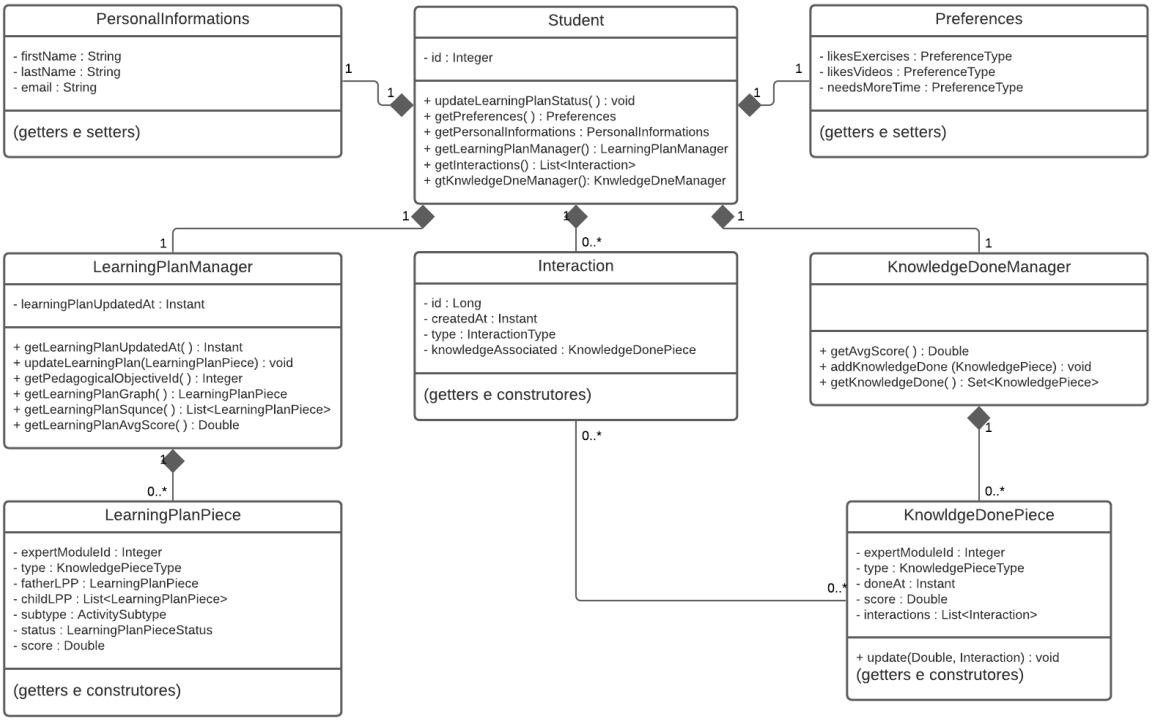

Figura 9: Diagrama de classes (simplificado) do modelo do módulo do estudante (autoria própria).

Um estudante possui seu id, um conjunto de Invitation's (classes que representam convites para turmas) e também o conjunto dos Ids das classes das quais faz parte. Através da composição, podemos desmembrá-lo em classes menores, com responsabilidades bem definidas.

5.1.1. PersonalInformations

Classe que agrupa as informações pessoais, como nome e e-mail.

5.1.2. Preferences

Classe que agrupa as preferências do usuário, como se o mesmo gosta ou não de exercícios ou se possui dificuldade, precisando de mais tempo para aprender um tema. Talvez "preferências" seja uma nomenclatura inadequada, dada em estágios iniciais do projeto, uma melhor denominação talvez seja "características".

Uma preferência pode assumir 3 valores: YES, NO, NEUTRAL_OR_UNDETECTED.

Sempre que um cliente da API requisita as informações de preferências do estudante, é acionado o PreferencesUpdater, que é o serviço responsável por atualizar as preferências do estudante a partir das interações do mesmo com o sistema. Como o Módulo de Tutoria adapta o plano de aprendizagem a partir das preferências do usuário, o PreferencesUpdater possui papel crítico na eficiência do sistema.

O mecanismo de inferência utilizado ainda é bastante simplificado, utilizando somas ponderadas. Por exemplo[2], para atualizar a preferência "likesVideos", o PreferencesUpdater segue os seguintes passos:

1. Busca as últimas *n* interações dos tipos "WATCHED_OPTIONAL_VIDEO" (tipo A), "SKIPPED_OPTIONAL_VIDEO" (tipo B);
2. Faz-se uma soma ponderada do tipo $a*x + b*y$, onde a e b são o número de ocorrências de interações dos tipos A e B, respectivamente. E x e y, são inteiros. Neste exemplo, supõe-se que
    - $a = 10$
    - $b = 7$
    - $x = 2$
    - $y = -3$

    Assim, dá-se maior importância ao evento de pular um vídeo do que ao evento de assistir. A soma neste caso é igual à -1.

3. Baseado no valor da soma e nos limitadores t e u, atualiza-se a preferência com o valor correspondente. Seja d o valor da soma, faz-se então:
    - $t =< d$: YES
    - $u =< d < t$: NO
    - $d < u$: NEUTRAL_OR_UNDETECTED

Supondo $t = 3$ e $u = -2$, o valor da preferência "likesExercises" seria atualizado para "NEUTRAL_OR_UNDETECTED".

[2] O exemplo foi um pouco simplificado em relação ao procedimento real do sistema para facilitar a compreensão, embora o comportamento fundamental seja exatamente como exposto.

Os tipos das interações, os pesos de cada uma, os delimitadores utilizados e o número de interações consideradas, são parâmetros que variam para cada preferência a ser atualizada.

5.1.3. Interactions

Lista de interações feitas pelo estudante, como responder um exercício opcional, responder uma atividade, tirar um score muito baixo em uma atividade, etc. A classe Interaction representa uma interação com o sistema.

5.1.4. KnowledgeDoneManager

Classe utilizada para gerenciar o estado de conhecimento do usuário, contendo o conjunto das KnowledgeDonePiece's feitas pelo usuário, através do qual pode retornar informações como o que o usuário já aprendeu e qual o seu desempenho como um todo ou em um assunto particular. KnowledgeDonePiece é a classe utilizada para representar uma unidade de conhecimento, contendo sua correspondência no Módulo Especialista, quando foi feita e com qual desempenho. Contém também as interações associadas.

Além do estado de conhecimento, o KnowledgeDoneManager também tem a função de histórico do conhecimento do usuário. Assim, quando um estudante adiciona um plano de estudos, o KnowledgeDoneManager é acionado para verificar se alguma parte do mesmo já foi feita anteriormente.

5.1.5. LearningPlanManager

Gerencia os planos de aprendizado do estudante, sendo responsável por os adicionar atualizar, remover, entre outras funções. LearningPlan é a classe que representa um plano de aprendizado e contém um grafo de LearningPlanPieces.

LearningPlanPiece é a classe utilizada para representar uma unidade do plano de aprendizagem, contendo informações como sua correspondência no Módulo Especialista, qual o seu status (DONE, TO_DO ou BLOCKED) e o desempenho. Também representa um nó no grafo do esquema do domínio, podendo possuir elementos filhos, elementos dependentes e um elemento-mãe.

5.2. Módulo do Professor

Este módulo possui as seguintes responsabilidades (relacionadas ao professor):

- Gerenciar informações pessoais;
- Criar e gerenciar turmas e planos de aprendizado.

As principais classes envolvidas no modelo podem ser vistas na Figura 10.

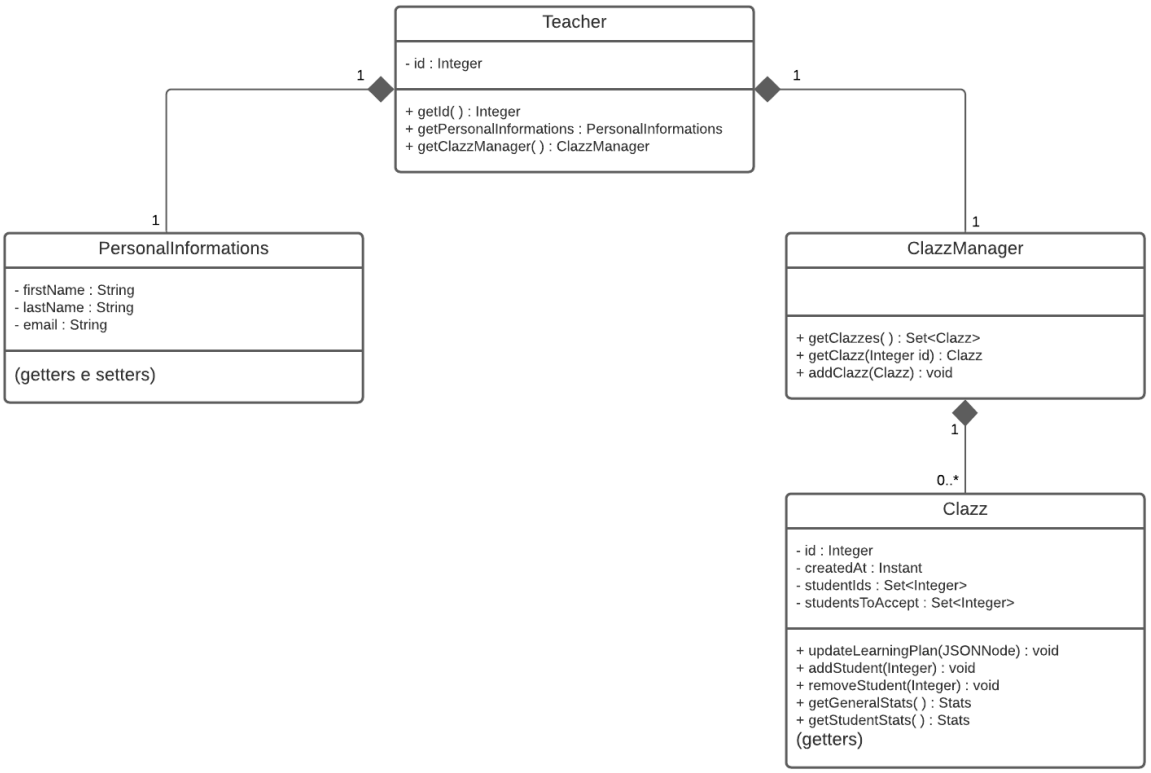

Figura 10: Diagrama de classes (simplificado) do modelo do módulo do professor (autoria própria).

Um professor possui o seu id, além de ser composto por classes menores, com responsabilidades bem definidas.

5.2.1. PersonalInformations

Classe que agrupa as informações pessoais, como nome e e-mail.

5.2.2. ClazzManager

Gerencia as classes daquele professor e provê funcionalidades como adição e exclusão de classes. Clazz é a classe que representa uma turma ou classe. Ela contém um nome, o conjunto dos ids do estudantes que fazem parte da mesma também um conjunto de planos de estudo.

ClazzLearningPlan representa um plano de estudo no Módulo do Professor e possui, além do plano de estudos, informações gerais como o score geral, o número de estudantes bloqueados, etc.

ClazzLearningPlanPiece é a classe utilizada para representar uma unidade do plano de aprendizagem, contendo informações como sua correspondência no Módulo Especialista e também informações sobre a aplicabilidade da mesma. Como o professor pode alterar o plano de estudos para seus alunos individualmente, algumas *pieces* podem ser aplicadas para todos os alunos e outras apenas para alguns.

5.3. Módulo Especialista

Este módulo possui a responsabilidade de organizar e disponibilizar todo o conteúdo pedagógico da aplicação. Para tanto, o conteúdo segue uma divisão esquemática, semelhante ao exposto na seção 2.2.2, mas com adaptações. O esquema do domínio deste módulo é ilustrado, de maneira simplificada, na Figura 11.

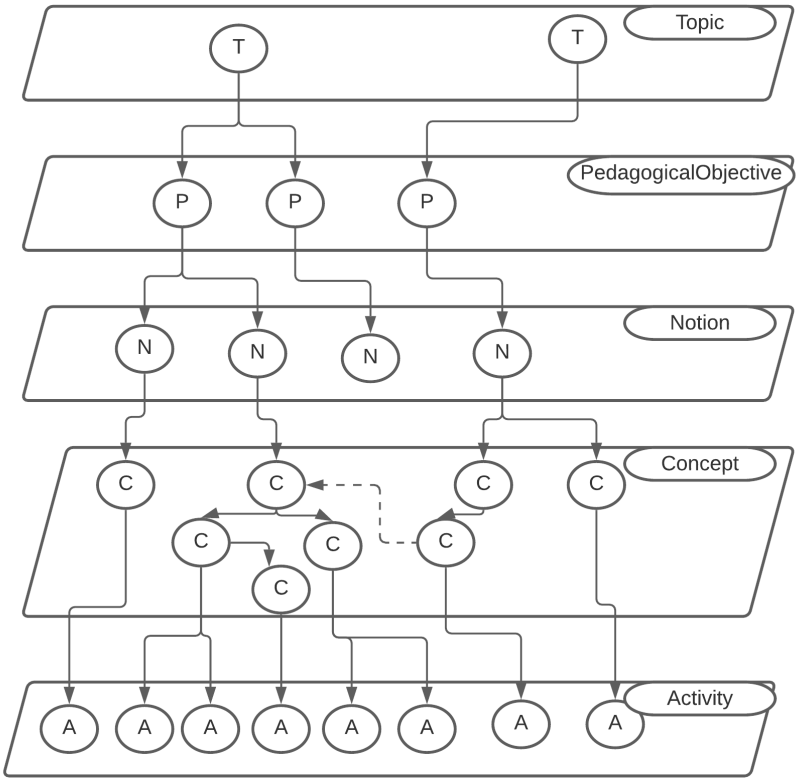

Figura 11: Esquema do domínio empregado (autoria própria).

Abaixo, seguem mais detalhes acerca dos níveis hierárquicos do esquema do domínio. São dados exemplos reais da aplicação, que, vale lembrar, tem o conteúdo destinado ao ensino de física básica a nível de Ensino Médio.

- **Topic:** compreende um tópico ou uma área ampla, que pode conter vários PedagogicalObjectives. Exemplos: “MECÂNICA”, “ELETROMAGNETISMO”;
- **PedagogicalObjective:** representa o que seria um capítulo inteiro de um Topic, contendo várias Notions. Exemplos: “Movimento Retilíneo Uniformemente Variado”, “Potencial Elétrico”;
- **Notion:** simboliza uma noção ampla, dentro daquele PedagogicalObjective, ou capítulo, contendo vários Concepts. Exemplos: “Queda Livre”, “Energia Potencial Elétrica”;
- **Concept:** a unidade mais complexa do esquema do domínio e que permite maior flexibilidade, seu objetivo é representar um conceito, direta ou indiretamente

subordinado a uma Notion. Exemplos: “Aceleração gravitacional”, “Diferença de Potencial”. Pode ter as seguintes relações:

    - Unidade mãe: pode ser filha direta de uma Notion ou de outro Concept, neste caso, é um subconceito. De toda forma, pode ter apenas uma unidade mãe;
    - Pode ter conceitos-filhos ordenados;
    - Pode ter dependência de outros conceitos (dependências não possuem uma ordem definida);
    - Pode ou não ter uma lista ordenada de atividades;
- **Activity:** representa uma atividade, possuindo tipo, subtipo e Content associado;
- **Content:** representa um conteúdo do tipo correspondente ao da atividade.

Os tipos da atividade informam o tipo de conteúdo da mesma, podendo ser:

- TEXT;
- INTERACTIVE;
- VIDEO;
- DIAGNOSTIC_EVALUATION
- FORMATIVE_EVALUATION;
- SUMMATIVE_EVALUATION;

Os diferentes tipos de avaliações e as suas utilidades seguem o que foi exposto na Seção 2.3.3. Assim, avaliações diagnósticas são compostas por exercícios voltados a inferir o estado de conhecimento atual do estudante, avaliações formativas são exercícios para auxiliar no aprendizado e avaliações sumativas são como provas, aplicadas ao fim de cada noção. Inicialmente, todos os exercícios são questões de múltipla escolha, mas posteriormente podem ser utilizados também outros meios de avaliação.

Uma atividade possui também um subtipo, que informa o papel daquela atividade dentro da sequência planejada. Os subtipos são:

- ESSENCIAL;
- COMPLEMENT;
- REINFORCEMENT;
- CHALLENGE.

O esquema do domínio foi projetado tendo em vista o compromisso entre a flexibilidade de representação do mesmo e a complexidade do modelo. Importante também é que, com as restrições descritas, o grafo do esquema do domínio pode ser “achatado” de apenas um modo, garantindo que a ordem de atividades proposta inicialmente seja sempre a mesma ao se partir de determinado nó. A implementação do modelo (Figura 12) foi simples, sendo a ordem dos elementos filhos garantida por um atributo “order”. Houve também cuidado em não permitir a inserção de ciclos.

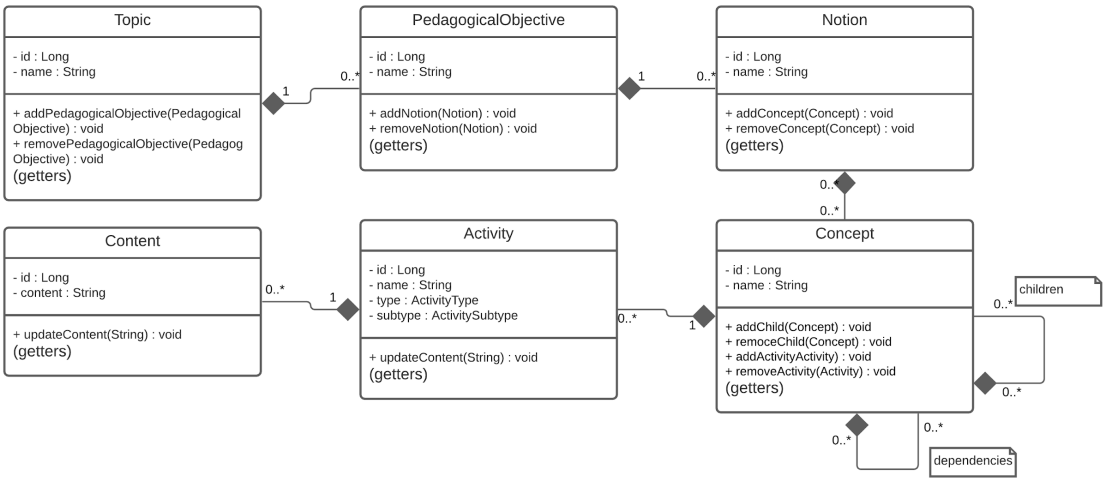

Figura 12: Diagrama de classes (simplificado) do modelo do módulo especialista (autoria própria).

Uma questão importante na implementação do Módulo Especialista é o formato de armazenamento do conteúdo. Embora o método mais utilizado seja com o uso de um texto com *tags* especiais que passa por algum interpretador antes de ser exibido, neste trabalho foi adotado um formato JSON, por simplicidade. Assim, na classe Content, há um atributo “content”, do tipo String, que contém o JSON com o conteúdo a ser exibido. Este JSON é interpretado por um código Javascript presente na página de exibição de conteúdos.

5.4. Módulo de Tutoria

Possui as seguintes responsabilidades:

- Fornecer acesso ao *front-end* da aplicação;
- Adaptar o conteúdo ao estudante.

O Módulo de Tutoria é uma aplicação web que segue o padrão MVC (*Model-View-Controller*). A única entidade gerenciada por este módulo é o user, que contém informações de login, senha, papel (professor ou estudante) e o id no módulo correspondente. O acesso ao usuário é feito apenas para autenticação e autorização, sendo feito totalmente pelo Spring Security. Este módulo utiliza as bibliotecas para uso dos demais módulos. Maiores detalhes sobre a parte de *front-end* podem ser vistos na Seção 4.2.

É o Módulo de Tutoria que gera o plano de aprendizagem do estudante. Para tanto, ele interage com o Módulo Especialista para obter o grafo do esquema do domínio e com o Módulo do Estudante para obter as preferências (características) do mesmo. Então, a partir das características do estudante, exclui certas atividades e gera o grafo de LearningPlanPiece's a ser passado ao Módulo do Estudante.

Para exemplificar este processo, supõe-se um estudante com as seguintes características:

- *likesExercises:* YES;
- *needsMoreTime:* YES;
- *likesVideos:* NO.

Neste caso, o algoritmo de geração do plano de aprendizagem retiraria vídeos complementares, colocaria atividades e exercícios de reforço e retiraria atividades mais desafiadoras, para evitar frustrar o aluno.

O plano de aprendizagem é gerado sempre que um novo objetivo pedagógico for escolhido pelo aluno ou se o mesmo ingressar na turma de algum professor. Podendo ser gerado novamente caso o Módulo do Estudante atualize as preferências do estudante em questão.

Também com o objetivo de personalizar o processo de aprendizagem ao estudante, este módulo adapta os *feedbacks* dos exercícios à situação do mesmo. Assim, um estudante com mais dificuldades ou que, por algum motivo, esteja errando muitos exercícios, recebe maiores auxílios para resolver exercícios do que um estudante que esteja com mais facilidade naquele momento.

6. RESULTADOS

Esta seção apresenta os resultados obtidos de fato com o projeto, cuja estrutura foi detalhada nas seções anteriores. Para tanto, esta seção é subdividida nos casos de uso de cada ator, conforme a seção 3.3. A subseção 6.1 expõe as funcionalidades comuns, enquanto a 6.2 apresenta os resultados referentes aos casos de uso do Estudante e a 6.3 apresenta os do Professor. Ao final, a subseção 6.4 relata brevemente os testes feitos.

6.1. Funcionalidades comuns

As únicas funcionalidades comuns são aquelas referentes ao cadastro, *login* e *logout* do usuário no sistema. Um usuário não autenticado tem acesso restrito apenas à páginas *home* (Figura 13) e à de conteúdos, embora o mesmo não possa selecionar um objetivo de aprendizagem (Figura 14). Posteriormente, seriam adicionadas mais páginas abertas, como as políticas da plataforma, "quem somos", etc.

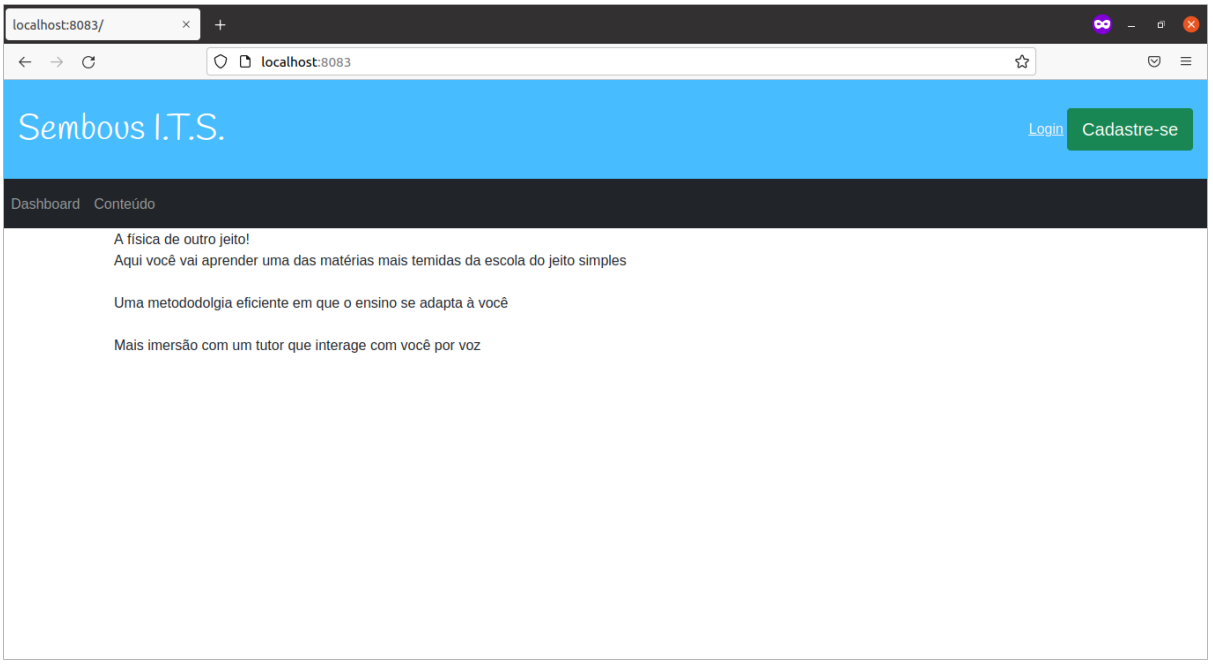

Figura 13: Página inicial da aplicação para usuário não autenticado (autoria própria).

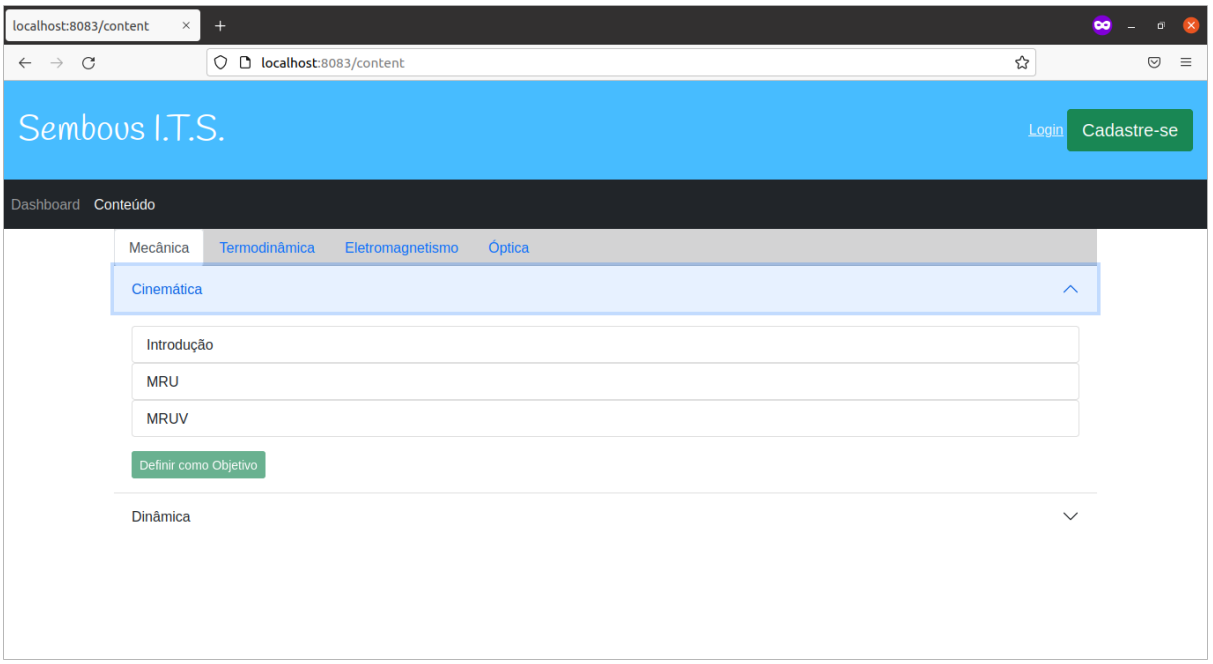

Figura 14: Página de conteúdos para usuário não autenticado (autoria própria).

O cadastro no sistema pode ser feito clicando-se no botão “Cadastre-se”, redirecionando o usuário para o formulário de cadastro, onde o mesmo pode colocar algumas informações básicas e também selecionar se quer ingressar na plataforma como professor ou estudante (Figura 15). Vale ressaltar que a plataforma ainda não faz a verificação de e-mail.

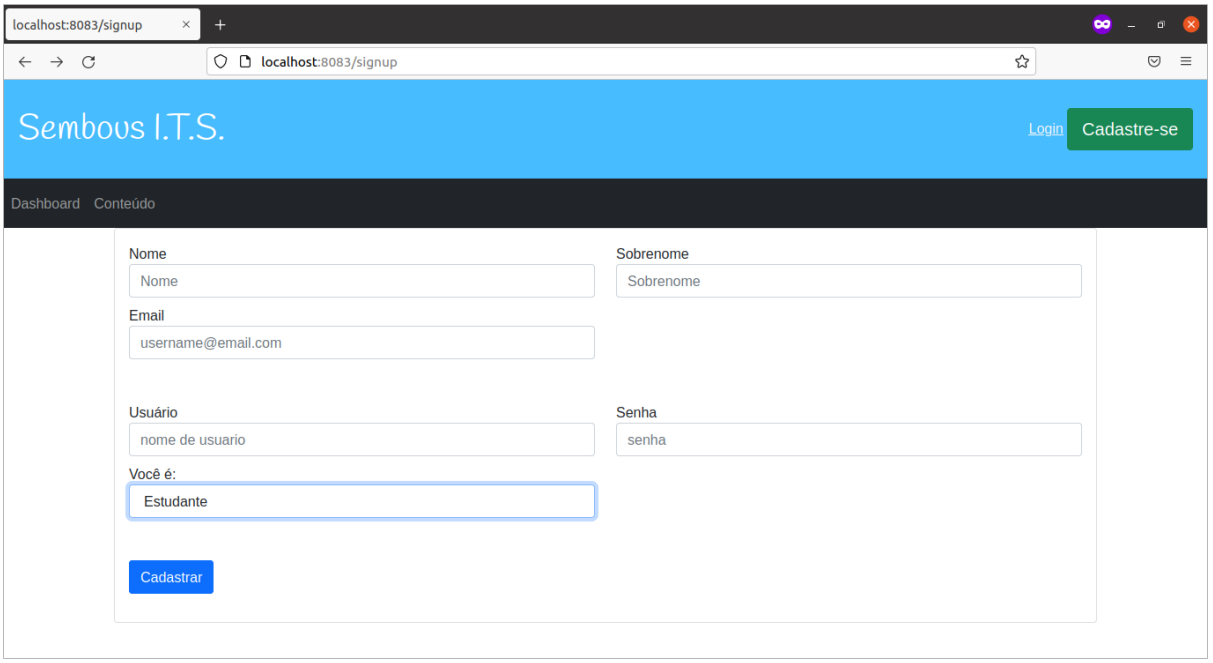

Figura 15: Formulário de cadastro na plataforma (autoria própria).

Para um usuário já cadastrado, basta clicar no botão “login” para ser redirecionado para um formulário onde é inserido o usuário e a senha cadastradas e o usuário se autentica no sistema, sendo ele imediatamente redirecionado ao seu Dashboard. Já o *logout* é feito pelo usuário já autenticado, bastando clicar no botão “logout”.

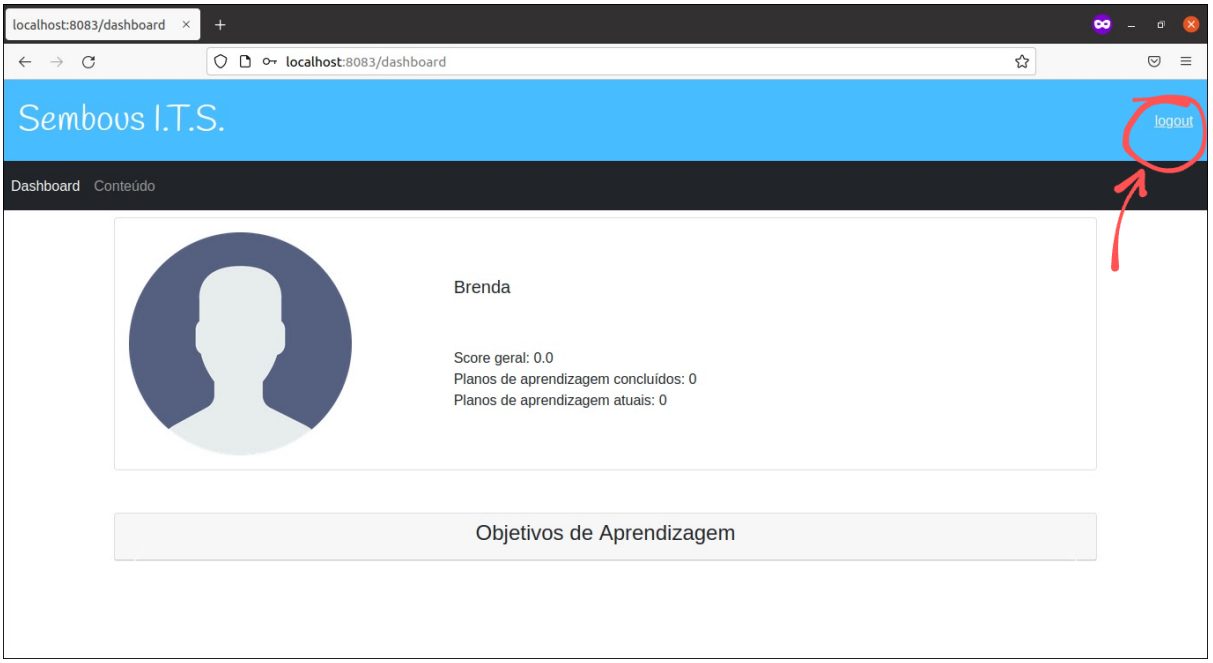

Figura 16: Página inicial da aplicação para um usuário autenticado, com foco no “logout” (autoria própria).

6.2. Estudante

Para o Estudante, o sistema fornece as seguintes funcionalidades:

- Selecionar um objetivo pedagógico;
- Acompanhar a própria evolução;
- Fazer a próxima atividade de um plano de estudos;
- Ingressar em uma turma.

6.2.1. Selecionar objetivo pedagógico

Para selecionar um novo objetivo pedagógico, o que consequentemente adiciona um novo plano de estudos ao estudante, o usuário precisa acessar a página de conteúdos, o que

pode ser feito pelo menu de navegação ou pela própria barra de endereços. A página de conteúdos (Figura 17) permite a seleção de um novo plano de estudos baseado em um objetivo de aprendizado. Nesta página pode-se conferir os objetivos e os tópicos (noções) de cada um. O sistema impede a adição de um plano de estudos múltiplas vezes, salvo o caso em que o plano de estudos faz parte de alguma turma. Ao adicionar um novo plano de estudo, o Módulo do Estudante também verifica se aquele plano já foi iniciado em algum momento e então atualiza com as informações passadas, para que o progresso do estudante não seja perdido.

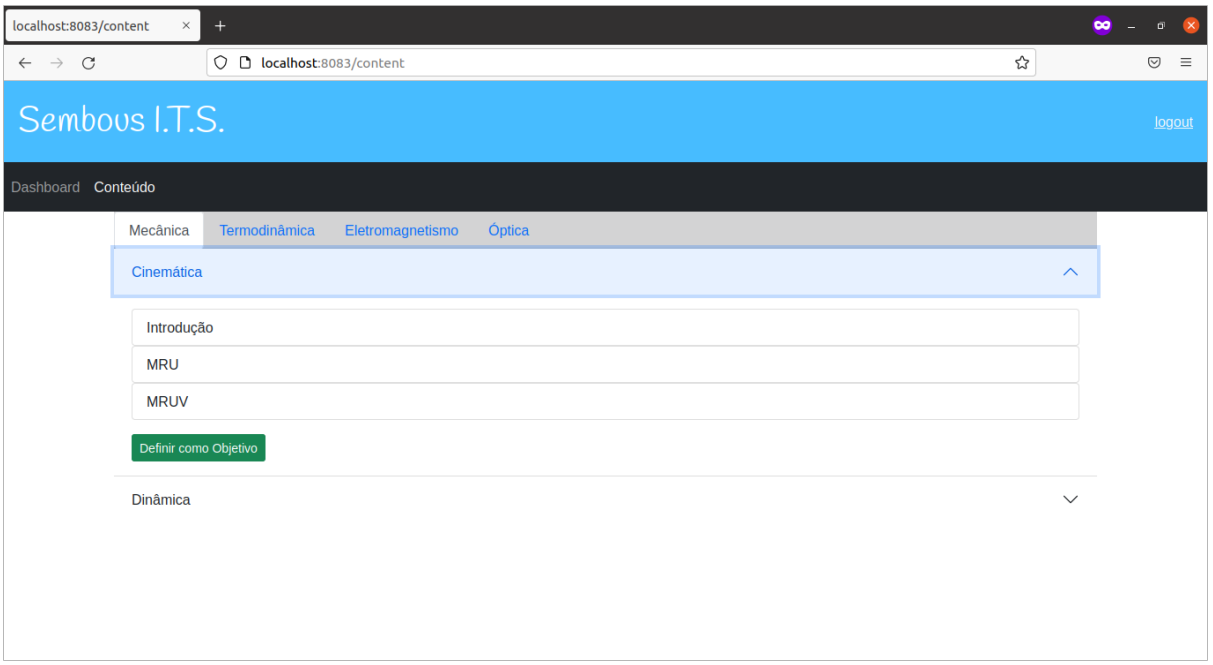

Figura 17: Página de conteúdos para estudante autenticado (autoria própria).

6.2.2. Acompanhar a própria evolução

Ao se autenticar no sistema, o estudante é redirecionado ao seu Dashboard, que provê muitas funcionalidades, como excluir de planos de estudo, visualizar as próprias estatísticas gerais de desempenho, responder um convite para uma turma e também acompanhar a própria evolução nos diferentes planos de estudo seguidos e nas turmas de que se faz parte.

No Dashboard podem ser visualizados os diferentes planos de estudo, relacionados ou não a uma turma, bastando clicar em “DETALHES”, conforme mostra as Figuras 18 e 19.

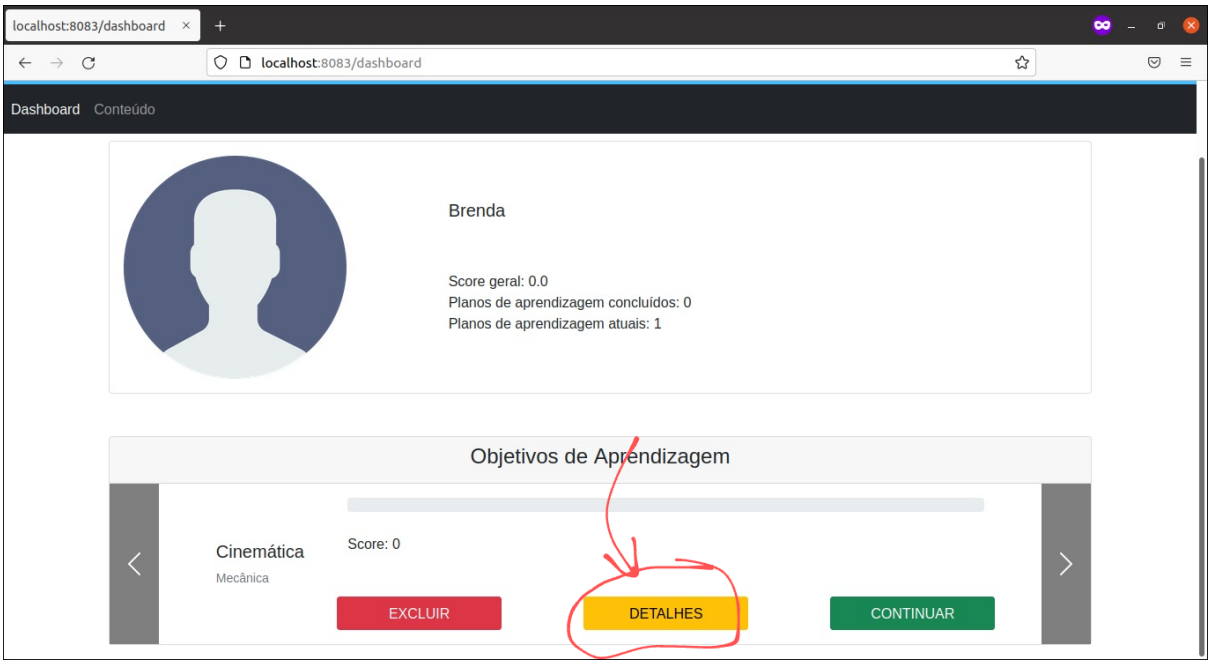

Figura 18: Dashboard do estudante com ênfase no botão de detalhes do plano de estudos (autoria própria).

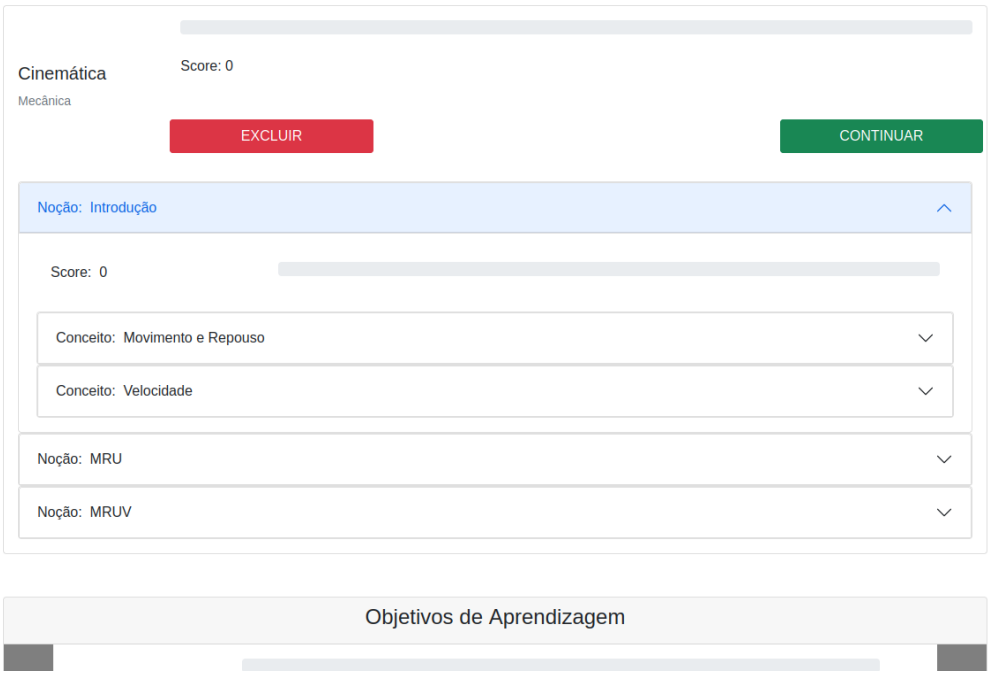

Figura 19: Dashboard do estudante com detalhes do plano de estudos (autoria própria).

6.2.3. Fazer a próxima atividade de um plano de estudos

No Dashboard é possível selecionar a opção “CONTINUAR” de um plano de estudos (Figura 20), redirecionando o estudante para a página com o conteúdo desta atividade, conforme mostra a Figura 21.

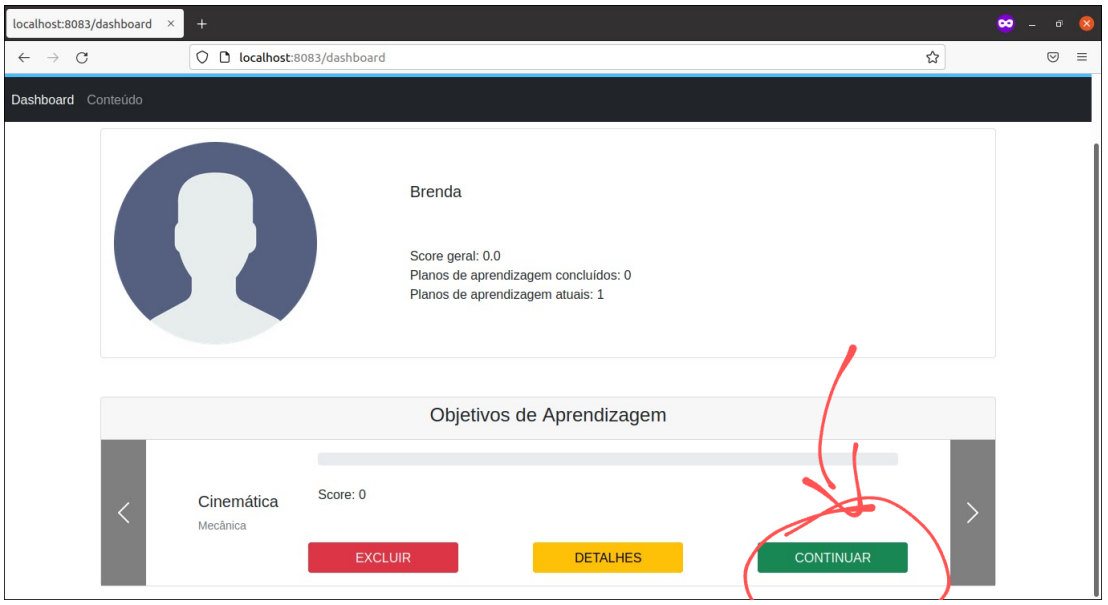

Figura 20: Dashboard do estudante com ênfase no modo de continuar o plano de estudos (autoria própria).

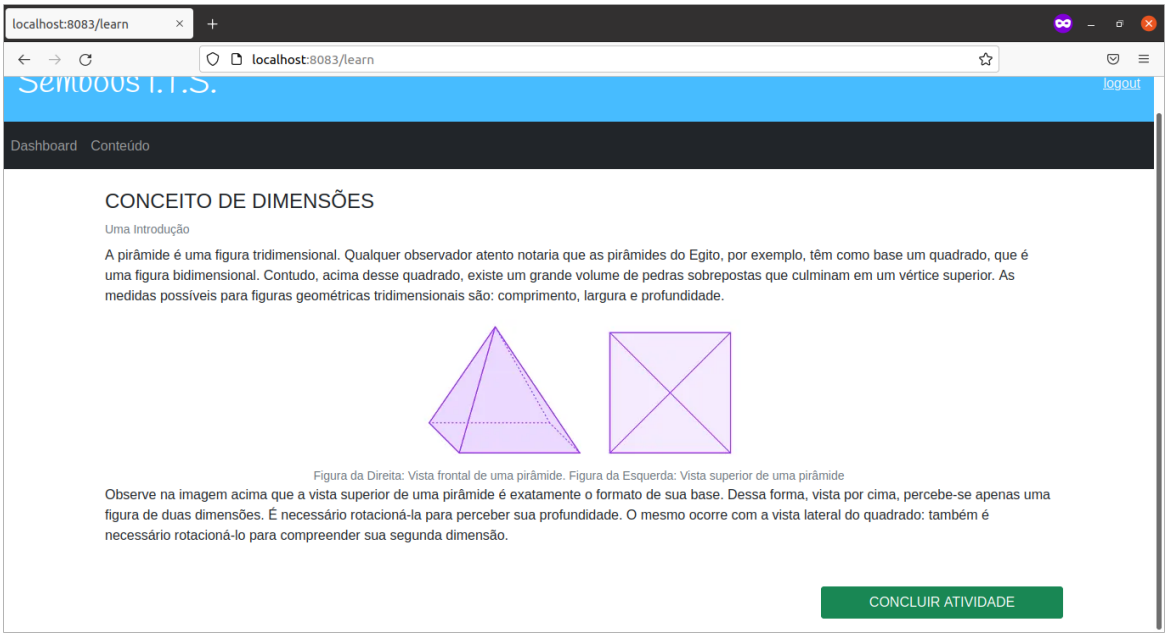

Figura 21: Exemplo de conteúdo de atividade (autoria própria).

A página onde são feitas as atividades renderiza o conteúdo de um Json vindo do Módulo Especialista, que pode conter informações textuais, imagens, vídeos, equações e exercícios. Num primeiro momento, a plataforma possui apenas exercício de múltipla escolha, porém não está inerentemente limitada a este tipo de exercício.

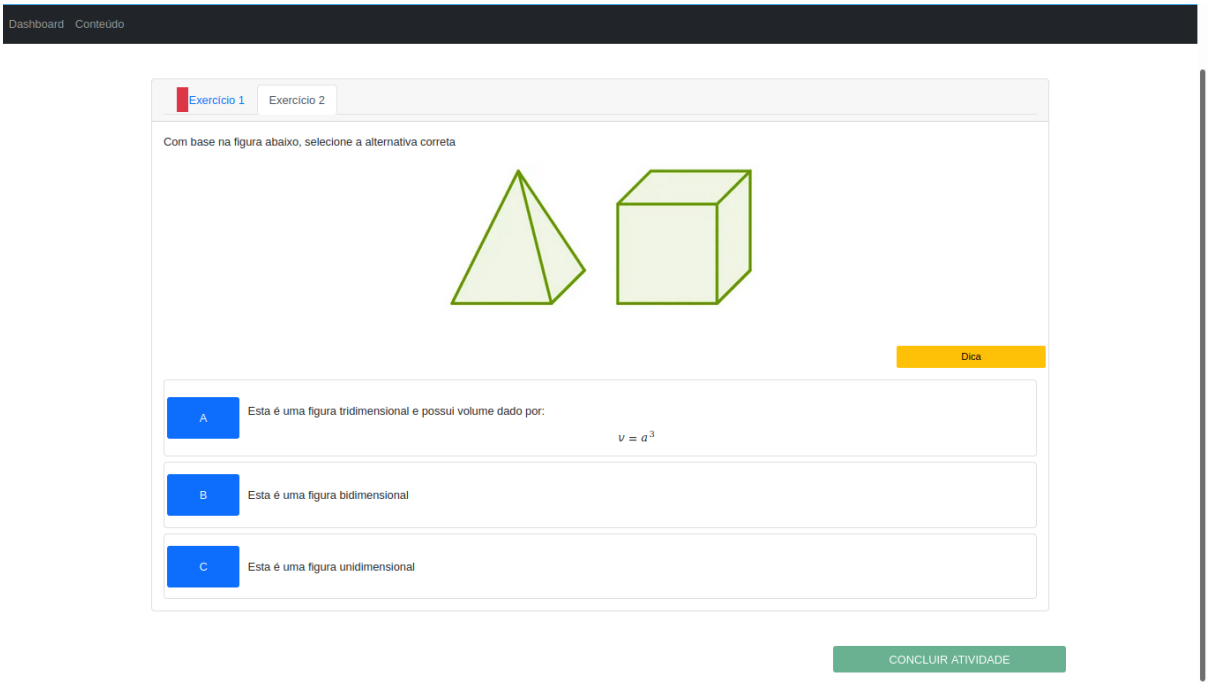

Figura 22: Exercício (autoria própria).

Após fazer um exercício, é mostrada a solução do mesmo, bem como comentários em cada uma das alternativas. O Json com o conteúdo pode ainda conter dicas fáceis, médias e difíceis para os exercícios e seleção de quais dicas exibir é feita a partir da inferência do Módulo do Estudante acerca das necessidades do mesmo.

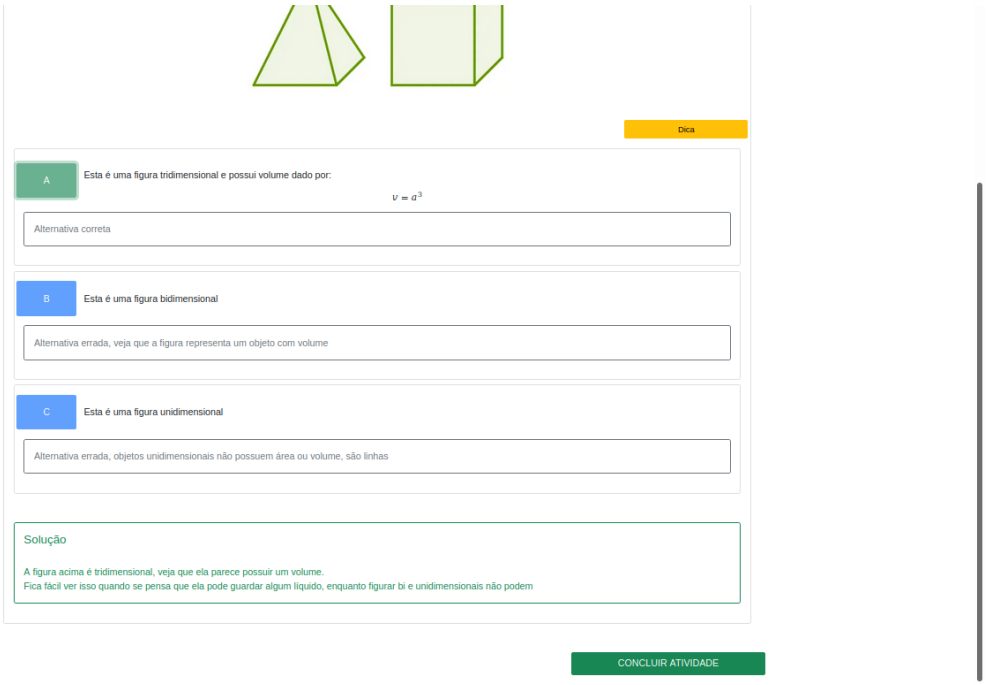

Figura 23: Exercício respondido (autoria própria).

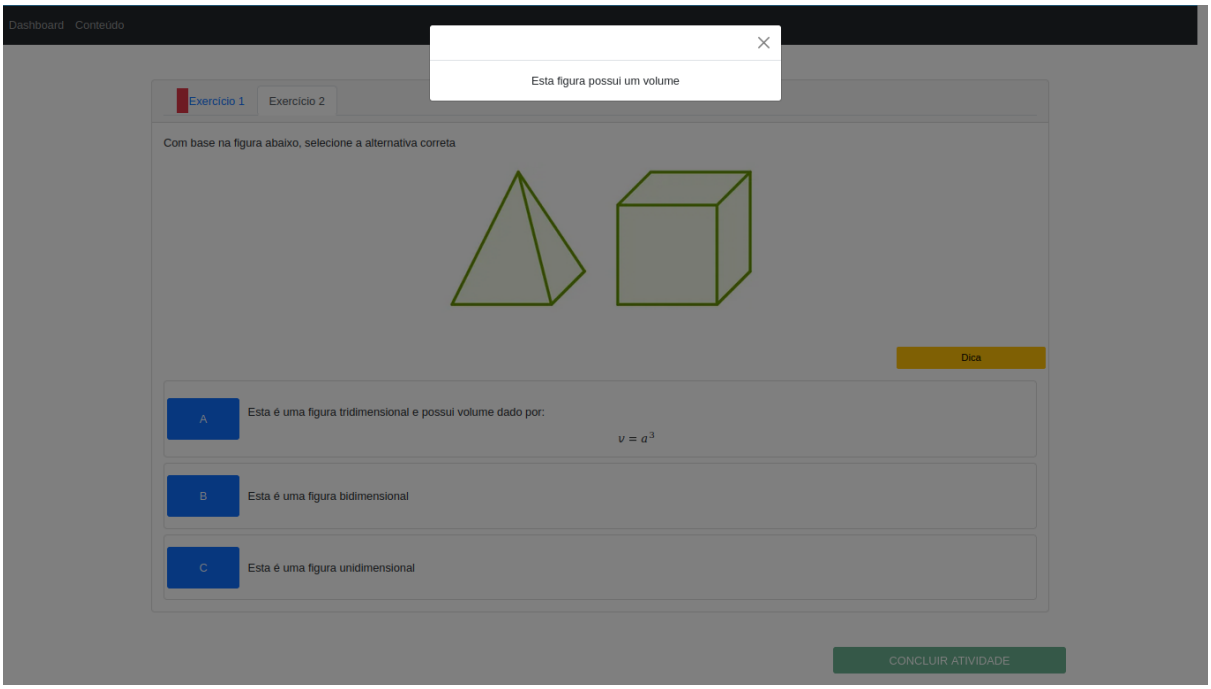

Figura 24: Dica de exercício (autoria própria).

6.2.4. Ingressar em uma turma

Para que um estudante ingresse em uma turma, primeiramente é necessário que o professor o convide para a mesma, então basta aceitar o convite e as informações da turma serão exibidas e os planos de estudo poderão ser seguidos.

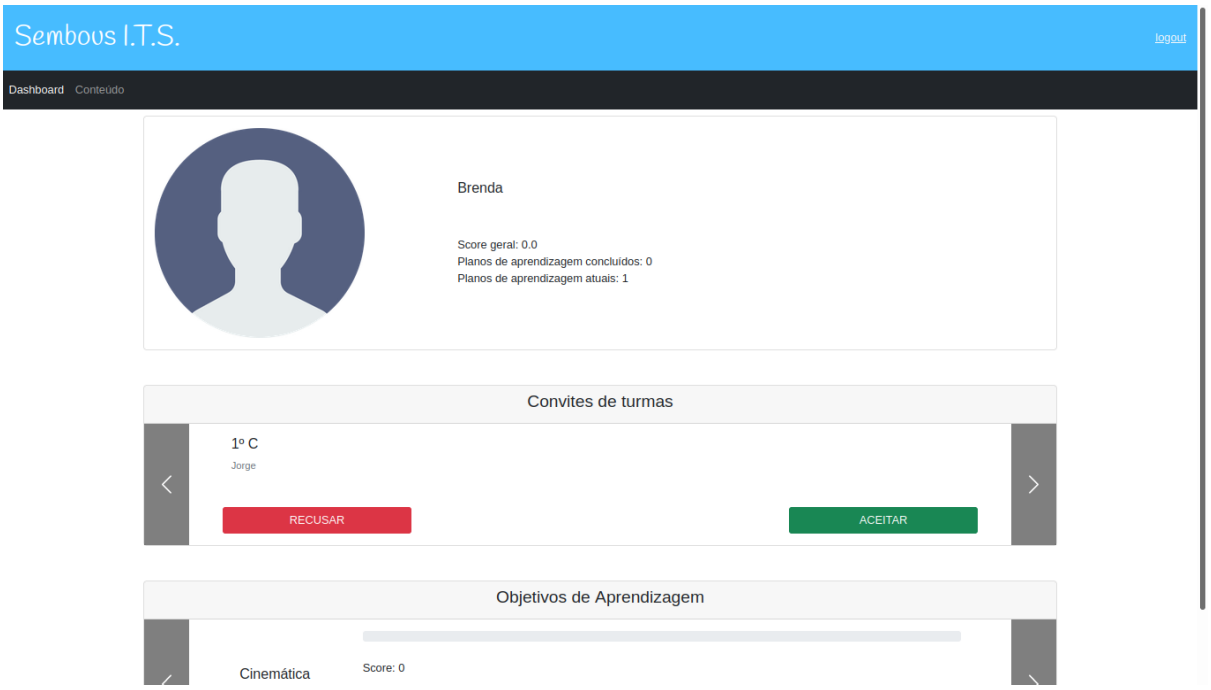

Figura 25: Dashboard do estudante com convite (autoria própria).

6.3. Professor

Para o professor, o sistema fornece as seguintes funcionalidades:

- Criar uma nova turma;
- Acompanhar a evolução das turmas;
- Convidar estudantes para uma turma;
- Selecionar planos de estudo para uma turma;
- Alterar os planos de estudo das turmas;

6.3.1. Criar uma nova turma

A criação de uma nova turma é feita pelo Dashboard do professor, clicando-se no botão “NOVA TURMA” que redireciona para um formulário de adição de nova turma. No momento é necessário apenas informar o nome da turma.

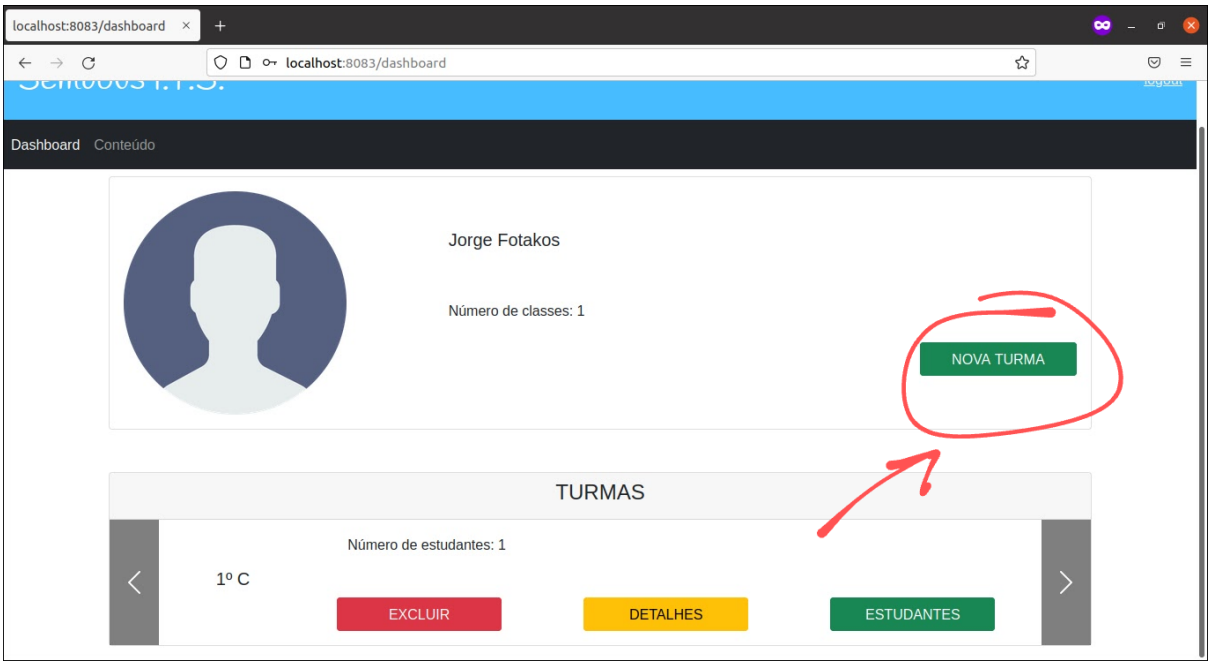

Figura 26: Dashboard do professor com ênfase na criação de nova turma (autoria própria).

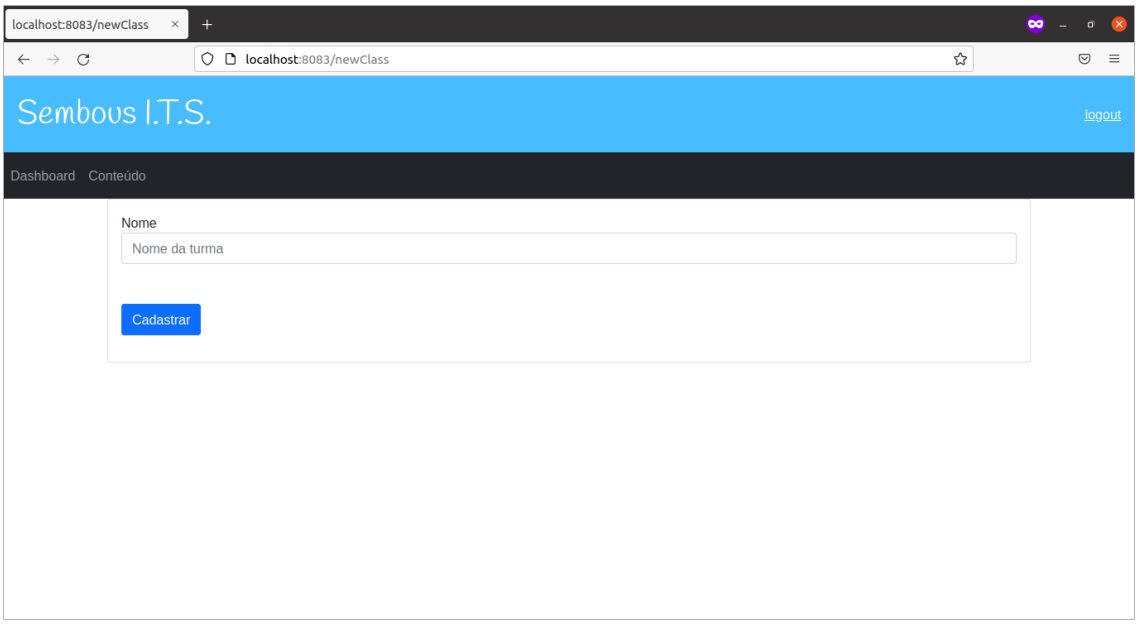

Figura 27: Formulário de adição de nova turma (autoria própria).

6.3.2. Acompanhar a evolução das turmas

No seu Dashboard, o professor pode selecionar uma turma para acompanhar, sendo redirecionado para a página da mesma (Figura 28). Nela, ele pode ver diversas informações da turma, como as estatísticas gerais, os alunos cadastrados e informações gerais de cada plano de estudo.

Figura 28: Página da turma (autoria própria).

6.3.3. Convidar estudantes para a turma

Na página da turma, o professor pode convidar um estudante clicando no botão "ADICIONAR ESTUDANTE", então será aberto um modal pedindo o endereço de e-mail cadastrado na plataforma pelo estudante, como mostra a Figura 29. Ao enviar o convite o mesmo será exibido no Dashboard do estudante.

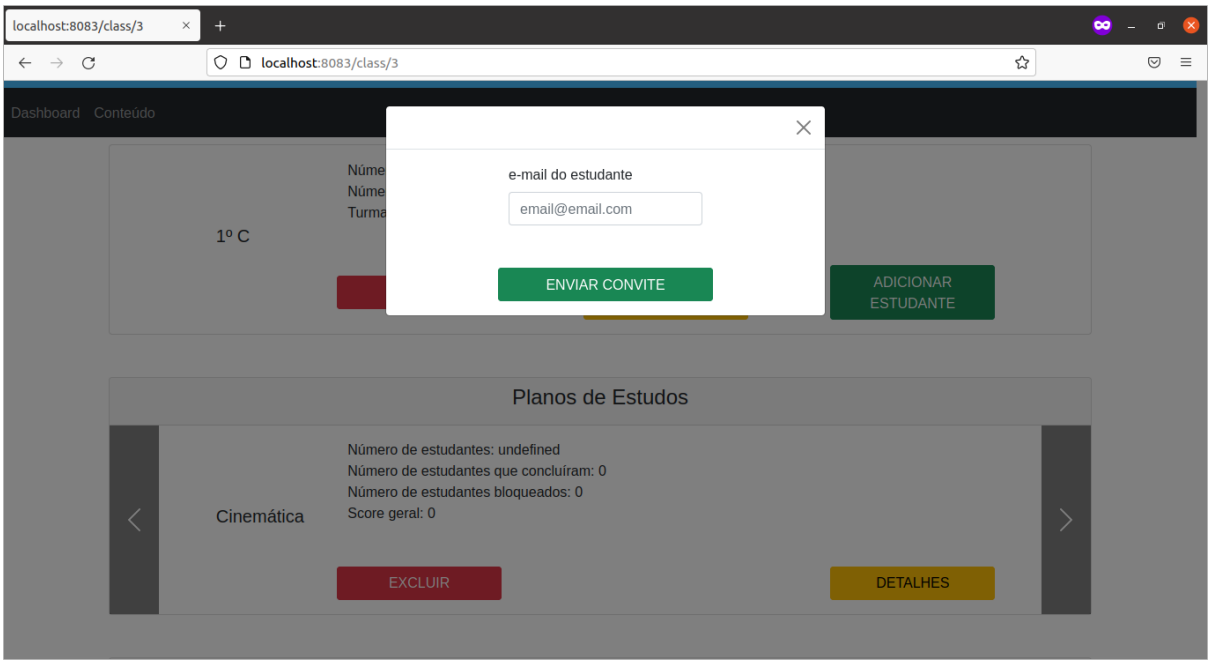

Figura 29: Modal para envio de convite para estudante (autoria própria).

6.3.4. Selecionar planos de estudo para a turma

Uma turma pode ter múltiplos planos de estudo, que serão exibidos para todos estudantes cadastrados na mesma. Para adicionar um plano, basta que o professor clique sobre o botão “ADICIONAR PLANO DE ESTUDO” na página da turma, que redireciona o usuário para uma tela semelhante a tela de conteúdos utilizada pelo estudante para selecionar um plano de estudos (Figura 31).

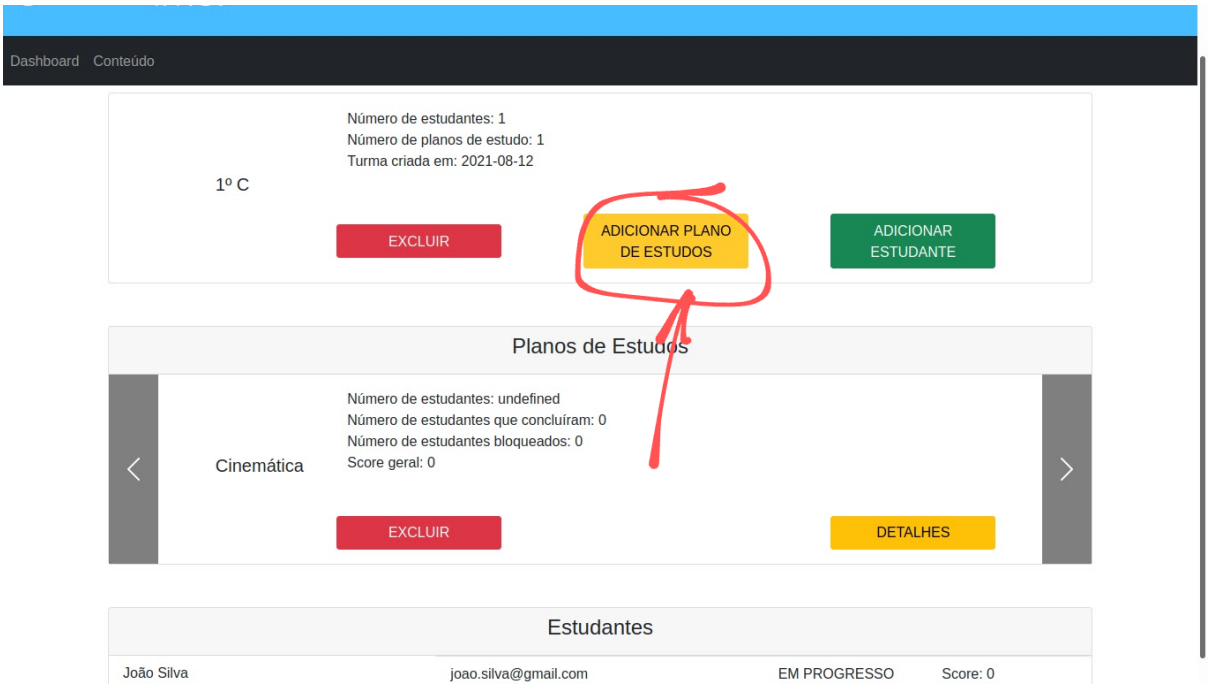

Figura 30: Página da turma com ênfase na adição de novo plano de estudos (autoria própria).

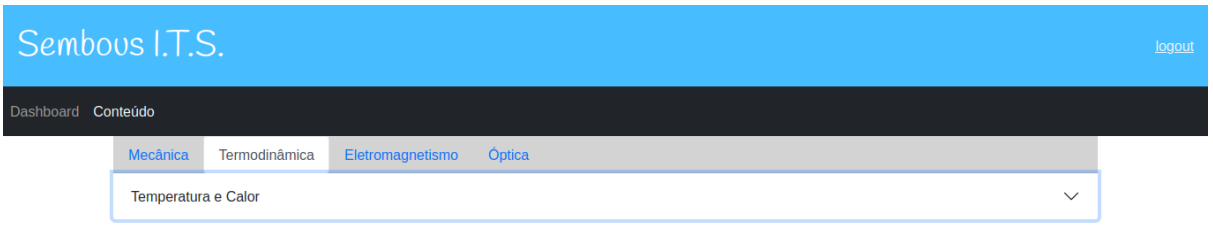

Figura 31: Página de seleção de plano de estudos (autoria própria).

Na página da turma, o professor pode visualizar as informações gerais de cada plano de estudo adicionado, bem como excluir os mesmos ou ser redirecionado para a página específica de cada plano.

6.3.5. Alterar os planos de estudo das turmas

Na página da turma, ao clicar no botão “DETALHES” de um plano de estudo, o professor é redirecionado à página específica daquele plano de estudos (Figura 33), que contém informações gerais daquele plano, informações do progresso dos estudantes e também permite a alteração do plano de estudos.

A alteração do plano de estudos se faz através da remoção ou inclusão de tópicos do Módulo Especialista, ou seja, noções, conceitos e atividades. As alterações no plano de estudos podem ser válidas para toda a turma ou apenas para um seleto grupo de estudantes, assim o professor pode traçar um plano de estudos geral, porém pode também dar acompanhamento personalizado aos estudantes.

6.4. Testes

Para testar efetivamente a aplicação foi utilizado o Ngrok para criar túneis com um endereço público para o servidor local, permitindo que familiares e amigos do aluno pudessem testar a plataforma. Além de avaliar a experiência do usuário, os testes serviram para prover um ambiente menos controlado e próximo do real. De modo geral, a aplicação se saiu muito bem, não apresentando bugs, embora as questões de performance não possam ainda ser avaliadas desta forma.

Embora altamente recomendados, não foram utilizados testes automatizados neste projeto, nem mesmo os mais simples testes unitários e de integração. O motivo foi a limitação de tempo, fazendo o autor priorizar a implementação das funções básicas aos testes, não sobrando tempo para desenvolver os mesmos.

7. CONCLUSÕES

O presente trabalho detalha o projeto e a construção de Sistema de Tutoria Inteligente (ITS) voltado ao ensino de física possuindo três objetivos principais: oferecer um ensino que se adapta às necessidades individuais dos estudantes, permitir que professores utilizem o sistema como ferramenta didática complementar e contribuir, de alguma forma, com a inclusão de pessoas menos favorecidas no processo educacional.

Após uma breve explicação da arquitetura tradicional de um ITS e de algumas estratégias pedagógicas, a implementação foi sendo explicitada, a começar pela arquitetura geral. Foi então evidenciada a intenção de que o projeto ficasse o mais próximo possível de um produto lançável no mercado, levando à adoção de um arquitetura de microsserviços RESTful com uso de ferramentas e *frameworks* de mercado. Após expor alguns detalhes do *back-end* e do *front-end* da aplicação, os módulos do projeto foram melhor descritos, com a responsabilidade, arquitetura e outros detalhes da implementação sendo explorados. Finalmente, foram expostos os resultados obtidos, ou seja, o funcionamento da plataforma e o modo como os casos de uso propostos inicialmente foram atendidos e testados.

O uso de uma arquitetura descentralizada, inspirada na muito popular arquitetura de microsserviços, mostrou-se muito útil ao permitir a melhor escalabilidade do código e separação das responsabilidades, o que fez com que raramente bugs ou comportamentos inesperados fossem observados e também permitiu que facilmente novas funcionalidades fossem adicionadas. Porém, a escrita de bibliotecas para se comunicar com as API's REST aumentou consideravelmente o volume de código escrito, sobretudo de classes DTO (*Data Transfer Object*).

Numa plataforma web como a que foi desenvolvida neste trabalho, pode-se dizer que sempre existirão pontos de melhoria e funcionalidades a serem acrescidas, o que justifica o fato de serem empregadas dezenas de desenvolvedores experientes para suprir estas demandas de forma competitiva. Naturalmente, existem muitos pontos de melhoria e limitações do projeto atual que merecem ser mencionados, alguns são mais complexos e exigiriam sofisticações no modelo, enquanto outras são bastante simples e não foram implementadas por limitações no tempo.

Dentre as melhorias mais complexas encontram-se os mecanismos de inferência utilizados para detectar as características do estudante e adaptar o plano de aprendizagem e a exibição do conteúdo para o usuário, que ainda são bastante rudimentares. Futuramente, o emprego de técnicas de IA mais sofisticadas, com o possível uso de Machine Learning, parece um caminho promissor para uma melhor adaptação ao usuário. Outra limitação do projeto é o *front-end*. Como foi citado anteriormente, o aluno não possuía competência técnica suficiente para fazer um bom projeto de *front-end*, desta forma, o mesmo ainda encontra-se com poucas funcionalidades, pouco responsivo e uma com uma arquitetura de código mal projetada.

A performance do sistema foi deixada como um ponto de melhoria futura, pois como a plataforma encontra-se apenas em ambiente de desenvolvimento, os gargalos de desempenho não são atingidos. Mesmo assim, durante o trabalho foram notados alguns pontos que poderiam melhorar a performance, como um melhor planejamento das *queries* no banco de dados, evitando sucessivas transações e minimizando o número de *joins* entre as tabelas, o uso de cache para consultas no banco de dados e no retorno de métodos e uma diminuição do número de chamadas entre os módulos. A descentralização do sistema em módulos independentes faz com que o módulo central (Módulo de Tutoria), precise constantemente se comunicar com os demais, desta maneira, um melhor planejamento dos dados transportados nas requisições entre módulos poderia diminuir o número total de chamadas feitas.

A segurança também é um ponto complexo e que precisaria ser tratado antes da plataforma ir para o ambiente de produção. Existem várias possíveis falhas de segurança e seria recomendado consultar um especialista, porém uma vulnerabilidade imediata vem do fato de os Módulos do Estudante, do Professor e Especialista ainda não possuírem qualquer sistema de autenticação. Como são API's REST, isso seria facilmente corrigido pela inserção de um esquema de autenticação utilizando JWT (*JSON Web Token*) pelo próprio Spring Security, porém foi deixado para um momento posterior, mais pela prioridade de outras funcionalidades do que por dificuldade técnica.

Há ainda algumas limitações pequenas e simples, porém numerosas, referentes a fluxos básicos e códigos simples que foram deixados para depois que as funcionalidades centrais e mais complexas fossem implementadas. Alguns exemplos seriam a alteração das informações básicas cadastradas, a verificação do e-mail, a exibição de mais informações nos

dashboards, tanto do professor como do estudante, melhoria da navegação e o uso de testes automatizados.

Mesmo com essas limitações e melhorias em vista, considera-se que o objetivo básico do projeto tenha sido atingido, pois foi obtida uma aplicação web robusta, flexível e escalável, que atende os requisitos básicos propostos de adaptabilidade ao estudante e possibilidade de uso integrado com a sala de aula por parte dos professores.

REFERÊNCIAS

Maia, L. O CONCEITO DE MEIO TÉCNICO-CIENTÍFICO-INFORMACIONAL EM MILTON SANTOS E A NÃO-VISÃO DA LUTA DE CLASSES. Ateliê Geográfico, 2012. Disponível em: <https://dx.doi.org/10.5216/ag.v6i4.15642>

Special edition: progress towards the Sustainable Development Goals. United Nations, 2019. Disponível em: <https://undocs.org/E/2019/68>

Pedró, F; Subosa, M; Rivas, A; Valverde, P. Artificial intelligence in education: challenges and opportunities for sustainable development. The United Nations Educational, Scientific and Cultural Organization (UNESCO), 2019. Disponível em: <https://unesdoc.unesco.org/ark:/48223/pf0000366994>

Olesea, C. The concept of personal learning pathway for Intelligent Tutoring System GeoMe. COMPUTER SCIENCE JOURNAL OF MOLDOVA, v.: 27, p.: 355-367, 2019.

Ramesh, V.M; Rao, N.J. Tutoring and Expert Modules of Intelligent Tutoring Systems. 2012 IEEE FOURTH INTERNATIONAL CONFERENCE ON TECHNOLOGY FOR EDUCATION, p.: 251-252, 2012.

Beyyoudh, M.; Khalidi, I. M.; Bennani, S.. Towards a New Generation of Intelligent Tutoring Systems. International Journal of Emerging Technologies in Learning (iJET), 2019. Disponível em: <https://dx.doi.org/10.3991/ijet.v14i14.10664>.

Wetzel, J. *et al*. The design and development of the dragoon intelligent tutoring system for model construction: lessons learned. Interactive Learning Environments, 2017.

Hooshyar, D.; Yousefi, M.; Lim, H.. A systematic review of data-driven approaches in player modeling of educational games. Artificial Intelligence Review, 2019.

Rybina, G. V. et al.. Using ontological approach in tutoring integrated expert systems. SECOND RUSSIA AND PACIFIC CONFERENCE ON COMPUTER TECHNOLOGY AND APPLICATIONS (RPC 2017), 2017.

Johnson, W. L.; Lester, J. C.. Face-to-Face Interaction with Pedagogical Agents, Twenty Years Later. International Journal of Artificial Intelligence in Education, 2016. Disponível em: <https://dx.doi.org/10.1007/s40593-015-0065-9>

O que são microsserviços?. RedHat. Disponível em: <https://www.redhat.com/pt-br/topics/microservices/what-are-microservices>. Acesso em: 13 de jul. de 2021.

IBM Cloud Education. REST APIs. IBM. 6 de Abril de 2021. Disponível em: <https://www.ibm.com/cloud/learn/rest-apis>. Acesso em: 13 de jul. de 2021.