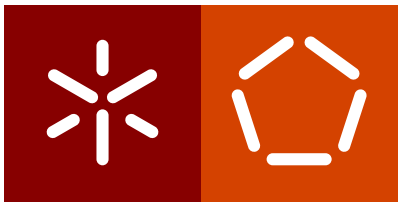
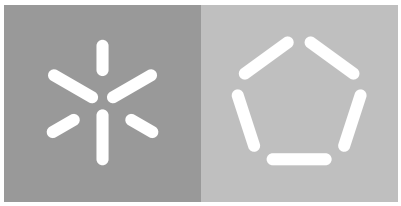


Universidade do Minho
Escola de Engenharia
Departamento de Informática

João António Cidade Vieira

Hypatiamat: Especificação e Implementação de um Componente para Gestão de Trabalhos de Casa

Dezembro 2021



Universidade do Minho
Escola de Engenharia
Departamento de Informática

João António Cidade Vieira

Hypatiamat: Especificação e Implementação de um Componente para Gestão de Trabalhos de Casa

Dissertação de Mestrado
Mestrado Integrado em Engenharia Informática

Dissertação orientada por
José Carlos Leite Ramalho
Ricardo Manuel Neves Pinto

Dezembro 2021

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

João Vieira

RESUMO

O Hypatiamat é um projeto que pretende mapear as condições de (in)sucesso e assim, contribuir para a promoção do sucesso escolar dos alunos do Ensino Básico, na disciplina de Matemática.

O desempenho escolar na Matemática é uma preocupação crescente junto da comunidade educativa: estamos todos preocupados com o elevado insucesso escolar e o correspondente abandono escolar precoce. Sabemos que a literatura educacional aponta a promoção do sucesso na Matemática e a utilização corrente das TIC na sala de aula como dois grandes desafios que a educação enfrenta na atualidade. Respondendo a estes desafios, a presente investigação organiza-se em torno da seguinte questão: Como podem as novas tecnologias, nomeadamente aplicações hipermédia utilizadas nos IWB (*interactive whiteboards*), contribuir para a promoção do sucesso escolar na Matemática[Hyp20]?

Para responder a estes desafios foi criado o Projeto Hypatiamat cuja parte mais visível é a sua Plataforma *web* com inúmeras aplicações hipermédia e jogos sérios centrados nos conteúdos de Matemática do 1.º ao 9.º ano. Com algumas aplicações ainda em fase de testes, mas com a garantia da qualidade e correção dos conteúdos, esta Plataforma utilizada por milhares de alunos e professores, de norte a sul do país, pretende contribuir para a promoção do sucesso neste domínio de conhecimento.

Nesta dissertação, foi especificado e implementado um novo componente que permite a um professor criar, monitorizar e avaliar trabalhos de casa (TPC), para posteriormente serem realizados pelos respetivos alunos.

Palavras-Chave: Hypatiamat; TPC; Estatísticas; Resoluções; Exercícios.

ABSTRACT

Hypatiamat is a project which maps the conditions of success/failure and contributes to the promotion of academic success for students of basic schooling, in the subject of mathematics. Students performance in mathematics is a growing concern in the education community: we are concerned with the high academic failure and correspondent school dropout by students. It is known that educational literature points in the promotion of success in mathematics and the use of information technology in the classroom as the two main challenges that education faces in the current days. In order to give answer to these challenges, the present investigation organizes itself around the next question: How can modern technology, like hypermedia applications used on IWB (interactive whiteboards), contribute to the promotion of academic success in mathematics[Hyp20]?

In order to respond to these challenges, the Hypatiamat Project was created, whose main highlight is a *web* Platform that offers numerous hypermedia applications and games centered on the mathematics subject from the 1st to the 9th scholar year. With some applications still in test phase, but with the quality guarantee and correction of its contents, this Platform used by thousands of students and teachers from the north to the south of the country, intends to contribute to the promotion of success in this knowledge domain.

In this dissertation, a new component was specified and implemented for the Platform, which allows a teacher to create, monitor and evaluate homework, which is to be made by its respective students.

Keywords: Hypatiamat; Homework; Statistics; Resolutions; Exercises.

CONTEÚDO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE EXEMPLOS

GLOSSÁRIO

Application Programming Interface Interface ou protocolo de comunicação entre um cliente e um servidor. xiii

back-end Parte de um sistema de *software*, onde os dados são armazenados ou processados. xi, 16, 21, 66, 67

Command-Line Interface Uma interface de linha de comandos é um meio de interação entre o utilizador e um programa de computador, em que o programa interpreta comandos emitidos pelo utilizador. xiii

CRUD CRUD (*Create, Read, Update, Delete*) são as quatro operações básicas utilizadas em bases de dados. 16, 38

Data Store Repositório de armazenamento persistente e de gestão de dados. 20

flag Valor que atua como sinal de controlo numa função ou processo. 12, 14, 28, 29, 45, 47

framework Estrutura de suporte pela qual se pode construir um sistema. 16, 19

front-end Parte de um sistema de *software* ou *website*, que é diretamente vista e manipulada por um utilizador. 16, 21, 66, 67

***headless* CMS** Sistema de gestão de conteúdo de *back-end* criado como repositório de conteúdo acessível por meio de uma API. 16

JSON Formato compacto, de padrão aberto e independente, utilizado em troca de dados simples e rápida entre sistemas. 17

Material Design Sistema adaptável de guias, componentes e ferramentas padrão que suportam as melhores práticas de desenho de interfaces. 19

MySQL Sistema de gestão de base de dados relacional, que utiliza a linguagem SQL como interface. 10, 15, 16

plugin Programa de menor dimensão, que torna mais rápido ou adiciona mais funcionalidades a outro de maior dimensão. 16, 31, 41, 43, 47

Single-page Application Aplicação *web* que interage com o utilizador dinamicamente, alterando a mesma página *web* com dados diferentes do servidor. xiii, 19

Sistema de Gestão de Base de Dados Sistema de software responsável pela gestão de uma ou mais bases de dados. 10

view layer Camada de *software* relativa às interfaces. 19

workflows Fluxo de Trabalho é uma sequência de passos necessários para se automatizar processos de negócio. 2

LISTA DE ACRÓNIMOS

API Application Programming Interface, Glossário: Application Programming Interface xi, 2, 4, 16, 21, 30, 31, 37, 38, 41, 43–46, 66–70

BD Base de Dados vii, 9–11, 13, 14, 16, 21, 27–31, 34, 35, 37, 39, 43–45

CLI Command-Line Interface, Glossário: *Command-Line Interface* 66

E-R Entidade-Relação 29, 30, 32

JWT Json Web Token x, 17–20, 41–43, 47, 67, 68

PIICIE Plano Integrado e Inovador de Combate ao Insucesso Escolar 1

SGBD Sistema de Gestão de Bases de Dados 10

SPA Single-page Application, Glossário: *Single-page Application* 19, 46

TIC Tecnologias da Informação e Comunicação iii, 2, 5

TPC Trabalho para Casa iii, vii, 2–4, 8–10, 13, 15, 17, 19–24, 26–31, 34, 37–41, 43–47, 51–61, 63–66, 68–70

UI User Interface 19

UX User Experience 19

1

INTRODUÇÃO

O primeiro capítulo desta dissertação, tem como objetivo apresentar toda a contextualização e motivação inerente ao tema do projeto proposto. Para além disto serão explicitados os objetivos e metodologias do trabalho, assim como a sua respetiva calendarização.

A última secção do capítulo fará um resumo da estrutura e conteúdos presentes neste documento.

1.1 CONTEXTO

É notório que nos dias de hoje, a sociedade enfrente cada vez mais desafios, aliados a uma constante globalização e desenvolvimento tecnológico em aceleração. Torna-se assim necessário formar os jovens adequadamente nas mais variadas valências, nomeadamente no contexto do ensino da matemática, que é tão importante para o seu desenvolvimento.

Neste sentido, a escola assume um papel fundamental, na medida em que é um local privilegiado para aliar a matemática ao desenvolvimento tecnológico que presenciamos diariamente, preparando assim os alunos para o futuro. Para além disto, cabe também à escola possibilitar um ensino adequado da matemática, de forma a que os alunos, desde cedo, ganhem gosto por esta componente do currículo que assume uma grande importância[Hor20]. Um professor ao ensinar matemática através das novas tecnologias, estará a fomentar no aluno um maior interesse na aquisição de conhecimentos, permitindo também a consolidação dos vários conceitos de uma forma mais significativa.

Tendo em conta este contexto, surgiu o desenvolvimento da Plataforma digital Hypatiamat. O Hypatiamat é, porventura, um dos projetos mais populares na área da Matemática, em boa parte devido ao seu acolhimento, em simultâneo, em planos de ação estratégica das escolas e em planos integrados e inovadores de combate ao insucesso escolar (PIICIE) e que em consequência do esforço de articulação com agrupamentos de escolas e professores ganhou prioridade estratégica no quadro das medidas de promoção do sucesso escolar[eTMC].

A Associação Hypatiamat/Projeto Hypatiamat são as entidades responsáveis por toda a manutenção e disponibilização dos conteúdos educativos da Plataforma. Através de protocolos e métodos de trabalho definidos, são feitos esforços no sentido de manter uma constante revisão e aperfeiçoamento daquelas que são as matérias que a Plataforma Hypatiamat oferece. A

Associação também disponibiliza formação adequada para que os docentes possam tirar o maior partido de todos os recursos e funcionalidades da Plataforma.

1.2 MOTIVAÇÃO

A principal finalidade da Plataforma Hypatiamat (`https://www.hypatiamat.com`) é a promoção do sucesso na disciplina de matemática, combatendo o elevado insucesso escolar e correspondente abandono escolar precoce. A literatura educacional aponta a promoção do sucesso da matemática e a utilização corrente das TIC na sala de aula como dois grandes desafios que a educação enfrenta na atualidade. Para dar resposta a estes desafios, o Projeto Hypatiamat organiza-se em torno da seguinte questão: Como podem as novas tecnologias, nomeadamente aplicações hipermédia utilizadas nos IWB (*interactive whiteboards*), contribuir para a promoção do sucesso escolar na Matemática[Hyp20]?

A Plataforma está orientada para alunos do 1.º ao 9.º ano de escolaridade mas é acessível por qualquer pessoa, e possui conteúdos hipermédia centrados nos respetivos programas curriculares da matemática, para cada ano de escolaridade. A par da Plataforma, emerge toda uma estratégia de intervenção pedagógica e de abordagem curricular com o objetivo de contribuir para despertar junto dos alunos o gosto pela matemática e uma melhor compreensão da sua natureza. Para tal, promove a utilização de tecnologias em sala de aula através dos recursos disponíveis na Plataforma, implicando e induzindo os professores e encarregados de educação a articularem, com autonomia os diferentes tipos de recursos pedagógicos disponíveis[eTMC].

Desta forma, a presente dissertação procura contribuir para o desenvolvimento e enriquecimento da Plataforma, juntando às aplicações já disponíveis a possibilidade de realizar TPC. Este novo componente, irá permitir criar uma ligação de aprendizagem entre o professor e o aluno para além da sala de aula, possibilitando ao docente um melhor acompanhamento e avaliação do conhecimento adquirido pelo aluno, tendo como base a utilização dos conteúdos e recursos já existentes na Plataforma Hypatiamat.

1.3 OBJETIVOS

Os objetivos definidos para esta dissertação são os seguintes:

- Planear e especificar lógica do componente, nomeadamente os *workflows* de professor e aluno;
- Acrescentar à base de dados existente os conceitos e relações necessárias;
- Desenvolvimento da API para a criação, monitorização e avaliação de TPC;
- Desenvolvimento de interfaces intuitivas à utilização do componente;
- Integração do componente desenvolvido com a Plataforma Hypatiamat já existente;

- Testes à implementação realizada.

1.4 METODOLOGIA

A metodologia adotada nesta dissertação, tem como base reuniões periódicas com o orientador e coorientador do projeto. Essas reuniões permitiram auxiliar no estudo dos requisitos necessários ao componente, assim como a respetiva validação e correção após implementação.

Numa fase posterior, o componente é colocado em produção e integrado na Plataforma Hypatiamat já existente. Assim sendo, os utilizadores da Plataforma têm oportunidade de testar o novo componente, fornecendo informações importantes para melhoria nas seguintes versões da aplicação.

1.5 CALENDARIZAÇÃO

- **Estudo do Problema** e compreensão sobre o que já foi implementado na Plataforma;
- **Levantamento e Análise de Requisitos**;
- **Desenvolvimento da API** de dados para a criação, monitorização e avaliação de TPC;
- **Desenvolvimento das Interfaces** de utilização;
- **Testes ao *Software* Desenvolvido** e análise de resultados;
- **Integração do Serviço** com a Plataforma Hypatiamat;
- **Escrita da Dissertação**.

	2020			2021								
	10	11	12	01	02	03	04	05	06	07	08	09
Estudo do Problema	■	■	■									
Levantamento e Análise de Requisitos		■	■									
Desenvolvimento da API				■	■	■	■	■	■	■		
Desenvolvimento das Interfaces					■	■	■	■	■	■	■	
Testes ao *Software* Desenvolvido						■	■	■	■	■	■	
Integração do Serviço							■	■	■	■	■	
Escrita da Dissertação			■	■	■	■	■	■	■	■	■	■

1.6 ESTRUTURA DO DOCUMENTO

Neste capítulo introdutório, foi realizada uma pequena apresentação do tema que esta dissertação aborda. Nomeadamente em que contexto se encontra inserido, a motivação

inerente à proposta de desenvolvimento deste tema, bem como quais são os objetivos a alcançar.

O segundo capítulo de Estado da Arte, começa pelo estudo de alguns casos de aplicações semelhantes ao Hypatiamat em contexto real. De seguida é explicitada a estrutura aplicacional da plataforma já existente, assim como o modelo de dados e respetivos tipos de utilizador. Também é feita uma introdução às diferentes ferramentas utilizadas para o desenvolvimento deste projeto.

No terceiro capítulo é explicada a modelação do novo componente, nomeadamente o levantamento dos requisitos funcionais assim como toda a preparação de conhecimento necessária para o desenvolvimento da aplicação, como novos modelos de dados e estrutura aplicacional.

Os capítulos seguintes, passam por demonstrar todo o processo de implementação do componente de TPC, desde o desenvolvimento da API de dados e da interface, assim como o processo de disponibilização do *software* em ambiente de produção.

O último capítulo é a conclusão deste documento, com algumas considerações sobre o trabalho desenvolvido, assim como sugestões de melhoria e trabalho futuro.

Neste primeiro capítulo, foi possível obter uma clarificação sobre o que é a Plataforma Hypatiamat, os seus princípios assim como o contexto em que está inserida. Também foi feita uma planificação do trabalho necessário à resolução do problema proposto.

O capítulo a seguir de Estado da Arte, irá expor todo o conhecimento e lógica existente no contexto da Plataforma Hypatiamat assim como a sua forma de funcionamento.

2

ESTADO DA ARTE

O presente capítulo descreve todo o Estado da Arte da Plataforma Hypatiamat. Desta forma, será feito um estudo sobre plataformas semelhantes e posteriormente uma análise detalhada sobre os conteúdos e recursos existentes na aplicação.

Também serão explicitadas as tecnologias necessárias ao desenvolvimento desta dissertação.

2.1 PLATAFORMAS DE APOIO AO ENSINO

No âmbito da temática em que esta dissertação está inserida, isto é a utilização das TIC no ensino da matemática, existem globalmente inúmeras aplicações e projetos de auxílio ao ensino de várias disciplinas durante a escolaridade.

A plataforma educativa mais reconhecida em termos de utilização de momento é a *Khan Academy*. Esta plataforma disponibiliza aos seus utilizadores vários recursos de aprendizagem, em várias áreas da educação e com conteúdos focados para diferentes idades. Para além da variedade de conteúdos, a *Khan Academy* permite a educadores a utilização no contexto de apoio aos seus alunos, inclusivamente possui uma funcionalidade de criação de tarefas, que podemos utilizar como um exemplo de comparação para com o projeto desta dissertação[Aca21]. Outras aplicações semelhantes que podem ser mencionadas são: *TalentLMS*; *Canvas*; *Blackboard Learn*; *McGraw-Hill Connect*; *Schoology*; entre muitas existentes nesta área do mercado, cada uma com diferentes vantagens e funcionalidades, seja em contexto de sala de aula como em contexto de trabalho para casa.

No panorama nacional português, para além do Hypatiamat, existem alguns projetos de *software* educativo, maior parte publicado por editoras. O mais utilizado de momento em contexto de sala de aula é provavelmente a *Escola Virtual*, com conteúdos desde o pré-escolar até ao ensino secundário[Vir21]. No que diz respeito a projetos de investigação portugueses sobre a utilização das TIC em sala de aula, podemos referir alguns como: a utilização da *Khan Academy* para combater o insucesso escolar na matemática[Soc20]; a *ClassDojo* como recurso educativo no ensino e aprendizagem da matemática[Mor19]; *GeoGebra*: um instrumento auxiliar no processo ensino/aprendizagem da matemática[Gon13]; entre muitas outras dissertações publicadas que abordam esta temática de estudo.

2.2 PLATAFORMA HYPATIAMAT

A Plataforma Hypatiamat através do trabalho que tem sido desenvolvido pelos membros da Associação Hypatiamat/Projeto Hypatiamat, disponibiliza vários e diversificados recursos educativos cujo principal objetivo é contribuir para a promoção do sucesso escolar na disciplina de matemática. A articulação de várias metodologias e ferramentas de trabalho permitem diversificar a abordagem de ensino do professor, promovendo uma aprendizagem do aluno mais rica em experiências e contextos.

Os recursos da Plataforma Hypatiamat estão organizados tendo em conta os diferentes temas presentes no currículo de matemática, o perfil do aluno à saída da escolaridade obrigatória e as aprendizagens essenciais na disciplina de matemática. Assim, os principais recursos disponibilizados pela Plataforma Hypatiamat são: uma coleção de jogos sérios e aplicações interativas com vista a promover competências específicas; disponibilização de fichas e recursos em *pdf* para utilização em contexto de sala de aula; guiões de exploração e leitura por parte do aluno, através do Mat-histórias; propostas de planificação e critérios para professores; propostas de utilização da Plataforma para auxiliar professores, alunos e encarregados de educação; existência de vídeos de apoio; uma base de dados com um número extenso de questões, orientada para trabalhar vários conteúdos da disciplina; um *backoffice* para monitorização do trabalho desenvolvido.

Figura 1: *Homepage* da Plataforma Hypatiamat

Através da figura 1, é possível visualizar a página inicial da Plataforma onde estão organizados os seus principais recursos. Na secção *Quero aprender* o aluno ao escolher um determinado tema, tem acesso a todas as aplicações, jogos e materiais de apoio relativos ao tema selecionado.

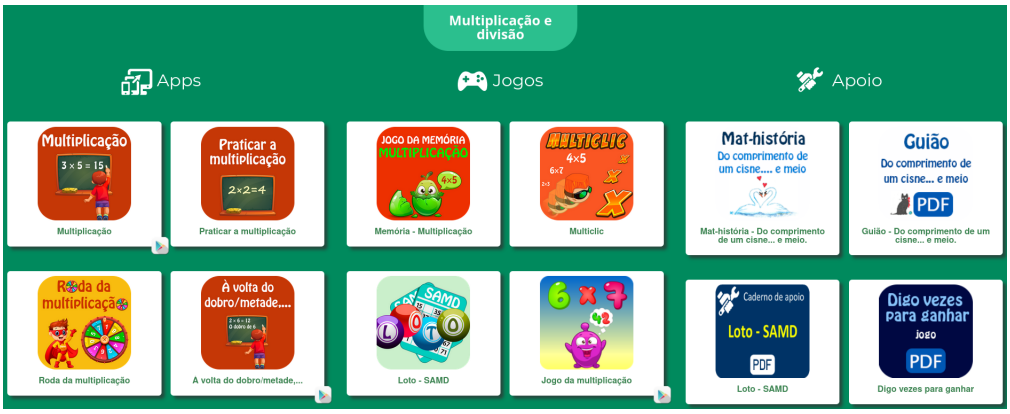


Figura 2: Hypatiamat - Secção *Quero Aprender*

A secção seguinte denominada *Quero resolver*, possibilita aos alunos a oportunidade de resolver os exercícios da base de dados disponibilizada pela Plataforma Hypatiamat, estando estes divididos em várias temáticas curriculares.

Figura 3: Hypatiamat - Secção *Quero Resolver*

Por último, a secção *Quero jogar* permite aos alunos a realização de jogos sérios interativos, para trabalhar competências matemáticas e/ou transversais e para consolidar o conhecimento. Estes jogos estão organizados consoante o seu tipo (jogos numéricos, de estratégia, geométricos e de memória/puzzles).

Figura 4: Hypatiamat - Secção *Quero Jogar*

Os professores envolvidos no Projeto são regularmente orientados através de ações de formação ou de *workshops* para tirar o melhor partido de todos os recursos e funcionalidades da Plataforma Hypatiamat. Por sua vez, o trabalho desenvolvido por ambos, professores e alunos, é monitorizado pela equipa Hypatiamat, através da utilização do *backoffice* que armazena toda a atividade desenvolvida pelos alunos. Este componente também permite ao professor acompanhar o desempenho em tempo real dos seus alunos.

Numa vertente mais lúdica, a Associação Hypatiamat promove a realização de campeonatos de cálculo mental entre alunos de escolas de norte a sul do país. Para estes eventos são utilizados os jogos e recursos da Plataforma, com o objetivo de desenvolver competências a nível de cálculo mental, sendo esta mais uma medida de sucesso naquela que é a finalidade deste Projeto.

No presente capítulo de Estado da Arte, será feita uma análise técnica à Plataforma Hypatiamat existente, abordando a estrutura aplicacional do sistema, assim como o modelo de dados e seus utilizadores. Também será analisada uma aplicação de TPC que inicialmente existia na Plataforma, que servirá como ponto de partida a esta dissertação.

2.2.1 *Aplicação de TPC Anterior*

Inicialmente a Plataforma Hypatiamat já disponibilizava, em conjunto com os seus conteúdos, uma aplicação que permitia a criação de trabalho autónomo. O professor submetia trabalhos com tarefas específicas, para serem realizados pelas suas turmas.

Uma das principais motivações para a proposta de trabalho relativa a esta dissertação, surge com o facto da tecnologia em que a anterior aplicação foi desenvolvida estar descontinuada, não sendo possível a sua utilização atualmente. A tecnologia em questão é o *Flash*, sendo que após o final do ano de 2020 passou a não estar suportada para desenvolvimento ou utilização em aplicações *web*.

A necessidade de desenvolvimento de uma nova aplicação de TPC com tecnologias mais recentes e estáveis tornou-se portanto imperativa. Para além da tecnologia, a aplicação anterior possuía várias limitações tais como: limite de cinco questões por cada trabalho;

impossibilidade de escolher alunos em específico das turmas; inexistência da possibilidade de múltiplas tentativas aquando da realização dos TPC por alunos; falta de configurações avançadas de criação; entre outras. Através da figura é possível ter uma noção de como era a aplicação de TPC entretanto descontinuada.

Figura 5: Aplicação de TPC Antiga

Todas estas circunstâncias serviram assim como pretexto para a criação de um novo componente de TPC. Este novo conteúdo aplicacional permitirá enriquecer e atualizar o Hypatiamat, numa vertente tão importante como é a realização e avaliação de trabalho autónomo por parte dos alunos.

2.2.2 *Estrutura Aplicacional*

A aplicação *web* da Plataforma Hypatiamat tem uma estrutura monolítica, sendo constituída apenas por um servidor, que é responsável por toda a lógica da plataforma assim como a apresentação de dados. O servidor realiza a comunicação com a BD para cada funcionalidade da aplicação, e a interface em *PHP* apresenta os dados ao respetivo utilizador. Através da figura seguinte, podemos visualizar a atual arquitetura.

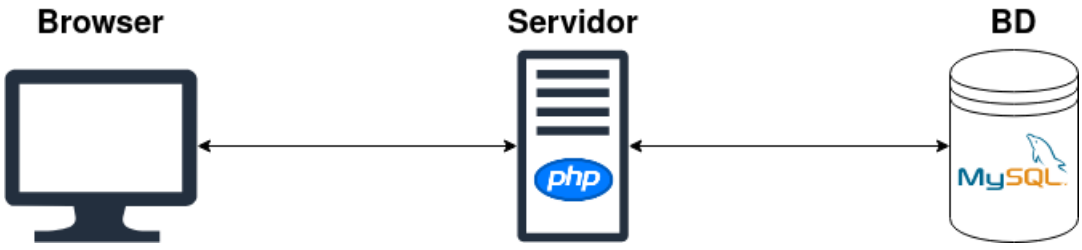


Figura 6: Arquitetura atual do Hypatiamat

O utilizador, através de um *browser*, acede à aplicação e ao interagir com esta realizará pedidos sobre o servidor em *PHP*. Este servidor por sua vez, consoante a informação que precisa, realiza pedidos ao servidor da BD, desenvolvido em MySQL. Após obtenção dos dados, o servidor *PHP* apresenta as informações necessárias através da sua interface.

2.2.3 *Modelo de Dados*

O modelo de dados representado pela base de dados da plataforma, é o local onde são armazenados todos os dados relevantes ao sistema, havendo uma estrutura específica para cada. Para além da sua função de armazenamento, a BD também permite que sejam realizadas determinadas operações sobre os dados, nomeadamente a inserção, alteração e remoção de registos.

Como já havia sido mencionado, a BD utilizada na plataforma é MySQL. O MySQL é um Sistema de Gestão de Base de Dados, desenvolvido e distribuído pela *Oracle Corporation*[Cor21]. Este sistema gere a estrutura da BD, desde o acesso aos dados em tabelas assim como a gestão dos mesmos através do *MySQL Server*. Uma das principais características deste SGBD é ser relacional, ou seja permite armazenar os diferentes dados em várias tabelas, sendo que essas tabelas são organizadas através de relações entre elas. As tabelas da BD possuem atributos específicos, caracterizados por exemplo, pelo seu tipo de dados, se é um atributo único à tabela, ou até se é uma chave primária ou estrangeira, que permite estruturar as relações entre tabelas. A linguagem utilizada para aceder e operar sobre as bases de dados do MySQL é SQL (*Structured Query Language*).

No que diz respeito ao Hypatiamat, o seu modelo de dados está dividido em quatro BD diferentes, sendo elas *Aplicações*, *Contadores*, *SAMD* e *Teste de Conhecimentos*. Como cada uma destas BD possuem um número extenso de tabelas, para efeitos de simplificação será realizado um foco nas que são relevantes para o objetivo desta dissertação (componente de gestão de TPC), nomeadamente as bases de dados *Aplicações* e *Teste de Conhecimentos*.

BD *Aplicações*

Esta base de dados tem como função principal armazenar informações acerca de elementos como professores, alunos, escolas e turmas. Cada um destes elementos é uma tabela, sendo que para além destas tabelas existem outras que não têm relevância nesta dissertação.

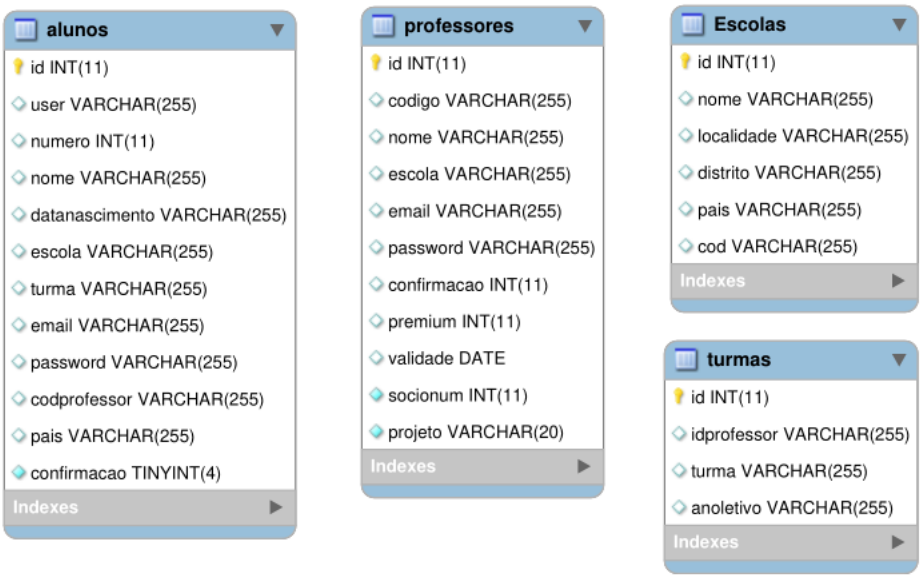


Figura 7: BD Aplicações - Modelo Lógico

No modelo lógico da figura, é possível visualizar cada uma das quatro tabelas assim como os atributos que as caracterizam.

Começando pela tabela *alunos*, esta tem a função de guardar toda a informação relativa aos alunos da plataforma e tem os seguintes atributos:

- *id* - Identificador único (numérico);
- *user* - Código identificador do aluno;
- *numero* - Número do aluno;
- *nome* - Nome do aluno;
- *datanascimento* - Data de nascimento do aluno;
- *escola* - Escola a que pertence o aluno;
- *turma* - Turma a que pertence o aluno;
- *email* - *Email* do aluno;
- *password* - *Password* de acesso do aluno;
- *codprofessor* - Código identificador do professor responsável;
- *pais* - País onde vive o aluno;

- *confirmacao* - *flag* que indica se está confirmado ou não na plataforma.

A tabela *professores* por sua vez, armazena a informação de todos os professores da plataforma. Os seus atributos são enumerados de seguida:

- *id* - Identificador único (numérico);
- *codigo* - Código identificador do professor;
- *nome* - Nome do professor;
- *escola* - Código da escola a que pertence o professor;
- *email* - *Email* do professor;
- *password* - *Password* de acesso do professor;
- *confirmacao* - *flag* que indica se está confirmado ou não na plataforma;
- *premium* - *flag* que indica o nível de acesso;
- *validade* - Data de expiração da validade da sua conta;
- *socionum* - Número de sócio Hypatiamat (caso possua);
- *projeto* - Projeto onde se encontra inserido.

No que concerne à tabela *Escolas*, esta possui todos os dados relativos às escolas do Hypatiamat, tendo como atributos:

- *id* - Identificador único (numérico);
- *nome* - Nome da escola;
- *localidade* - Localidade da escola;
- *distrito* - Distrito da escola;
- *pais* - País da escola;
- *cod* - Código identificador da escola.

Por último, a tabela *turmas* contém informação de todas as turmas que existem na plataforma. A nível de atributos temos apenas:

- *id* - Identificador único (numérico);
- *idprofessor* - Código identificador do professor responsável;
- *turma* - Código identificador da turma;

- *anoletivo* - Ano letivo ao qual diz respeito a turma.

BD *Teste de Conhecimentos*

A BD *Teste de Conhecimentos* é responsável por armazenar informações acerca das aplicações de conteúdo do Hypatiamat e da sua respetiva utilização. Sendo assim temos um número extenso de tabelas que guardam dados sobre exercícios, temas, subtemas, campeonatos, cromos, tarefas realizadas pelos alunos, entre outros.

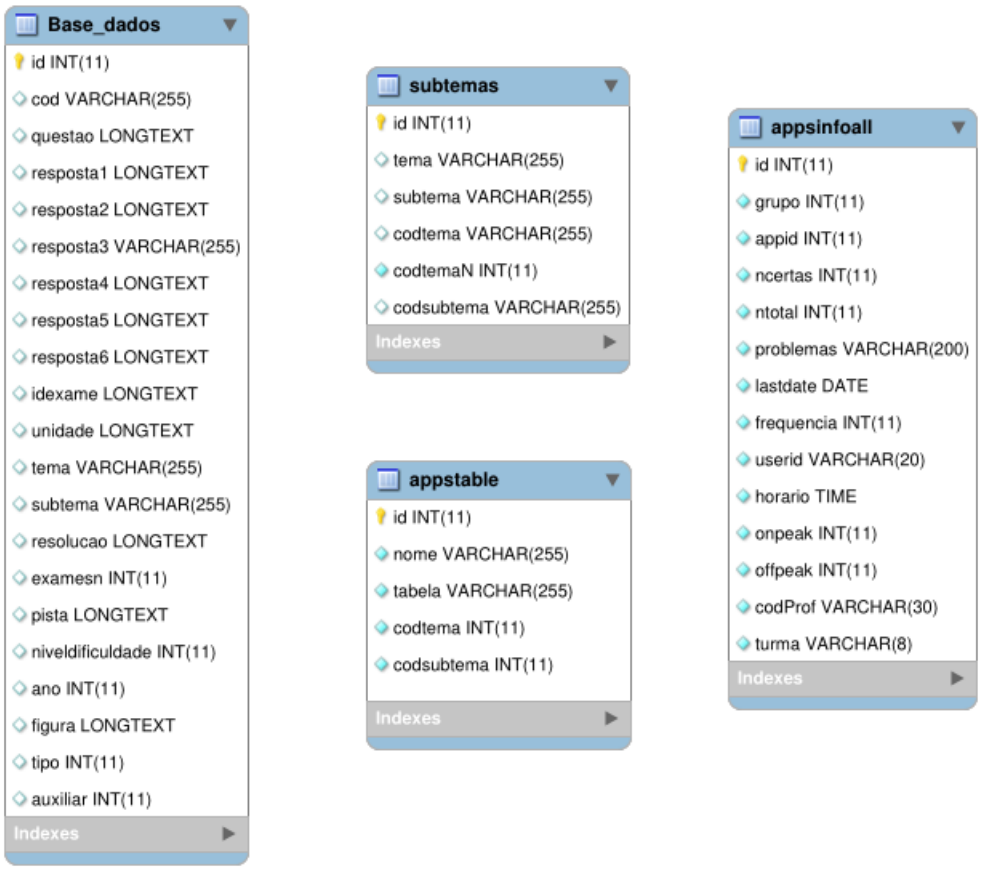


Figura 8: BD Teste de Conhecimentos - Modelo Lógico

As tabelas apresentadas através do modelo da figura representam os dados mais relevantes para o âmbito da dissertação.

A tabela *Base_dados* tem fundamental importância pois contém toda a informação acerca de exercícios que serão utilizados no componente de TPC. Esta coleção de tarefas é resultado da recolha por parte da equipa do Hypatiamat, de vários exercícios que incidem em diferentes temáticas da matemática, com diferentes níveis de dificuldade, sendo que muitos são provenientes de questões de exames.

Em relação aos seus atributos, são explicitados a seguir:

- *id* - Identificador único (numérico);
- *cod* - Código da tarefa;
- *questao* - Enunciado da tarefa;
- *resposta(1-6)* - Soluções da tarefa;
- *idexame* - Identificador do exame, se existir;
- *unidade* - Unidade da resposta;
- *tema* - Código do tema da tarefa;
- *subtema* - Código do subtema da tarefa;
- *resolucao* - *Path* para imagem da resolução;
- *examesn* - *flag* que indica se é tarefa de exame;
- *niveldificuldade* - Valor de dificuldade (1 a 4);
- *ano* - Ano escolar a que corresponde a tarefa;
- *figura* - *Path* para imagem da tarefa;
- *tipo* - Valor que indica o tipo da tarefa;
- *auxiliar* - Variável de controlo.

No que diz respeito à tabela *subtemas*, esta possui as informações sobre todo o conjunto de temas e subtemas disponíveis. Cada um dos temas possui um conjunto de subtemas relativos a este, sendo que um determinado subtema caracteriza por si um conjunto de tarefas específico.

Quanto aos seus atributos, temos:

- *id* - Identificador único (numérico);
- *tema* - Designação do tema;
- *subtema* - Designação do subtema;
- *codtema* - Código do tema correspondente;
- *codsubtema* - Código do subtema correspondente.

Como já foi mencionado, esta BD também tem a função de armazenar o desempenho dos alunos aquando da resolução das tarefas, utilizando para isso várias tabelas, sendo que neste caso destacamos duas *appstable* e *appsinfoall*.

A tabela *appstable*, serve como índice para outras tabelas, em que consoante o tema e subtema de uma tarefa, obtemos o nome da tabela a atualizar com o desempenho do aluno, para aquela tarefa em específico.

Tem os seguintes atributos:

- *id* - Identificador único (numérico);
- *nome* - Nome da temática que incide a tarefa;
- *tabela* - Designação da tabela a atualizar;
- *codtema* - Código do tema correspondente;
- *codsubtema* - Código do subtema correspondente.

Por sua vez, a tabela *appsinfoall* tem como objetivo guardar o desempenho dos alunos na resolução de todas as tarefas, para temas e subtemas diferentes.

Apresentam-se portanto os seus atributos:

- *id* - Identificador único (numérico);
- *grupo* - Código de tema;
- *appid* - Código de subtema;
- *ncertas* - N.º de tarefas respondidas corretamente;
- *ntotal* - N.º total de tarefas respondidas;
- *lastdate* - Última data em que resolveu uma tarefa;
- *frequencia* - N.º de acessos;
- *userid* - Código identificador do aluno;
- *horario* - Última hora em que resolveu uma tarefa;
- *onpeak* - N.º de acessos em horário de pico;
- *offpeak* - N.º de acessos fora de horário de pico;
- *codProf* - Código identificador do professor responsável;
- *turma* - Código identificador da turma.

2.3 Estudo das Tecnologias

A seguinte secção realiza um estudo sobre quais as tecnologias necessárias para aquele que é o objetivo deste trabalho, o componente de TPC.

Esta aplicação de TPC não será implementada na mesma estrutura e com as tecnologias sobre as quais a Plataforma Hypatiamat se encontra a funcionar. Apenas o serviço de base de dados se manterá o MySQL, sendo que será adicionada uma nova base de dados às que já existem na plataforma, dedicada ao armazenamento específico dos dados de TPC, como será explicitado em capítulo posterior.

A aplicação *web* relativa a esta dissertação, estará estruturada através de dois servidores, um servidor de *back-end* e outro de *front-end*, que se comunicarão entre si. A tecnologia a utilizar para a API de dados (*back-end*) será *Node.js*, com a particularidade que estará integrada na implementação através de um sistema denominado de *Strapi*. No que ao servidor de *front-end* concerne, a *framework* utilizada para construir as interfaces de utilizador será o *Vue.js*.

2.3.1 *Node.js*

Node.js é uma tecnologia que executa em *JavaScript*, com a característica de funcionar num ambiente assíncrono baseado em eventos. Esta ferramenta é desenhada para construir aplicações em rede que sejam escaláveis.

O *Node.js* funciona ao contrário de um modelo de concorrência, onde são utilizadas *threads*. A utilização de um modelo de *threads* em aplicações de rede é mais ineficiente e mais difícil de implementar. Portanto, o *Node.js* não tem este tipo de desafios, é feito de maneira em que não hajam *dead-locks* do processo, pois não existem *locks* explícitos[Fou21].

Sendo que o Hypatiamat já comporta um grau elevado de complexidade, e estando o *Node.js* preparado para suportar sistemas altamente escaláveis, esta foi a tecnologia adotada para o desenvolvimento da API de dados.

2.3.2 *Strapi CMS*

O *Strapi* é um *open-source headless* CMS flexível, que dá aos programadores a liberdade de escolher as suas ferramentas favoritas e *framework*, enquanto que os editores podem gerir e distribuir o seu conteúdo facilmente. Ao ter um painel de administração e uma API extensível através de um sistema de *plugin*s, o *Strapi* permite às empresas acelerar o processo de entrega de conteúdo ao mesmo tempo que constroem uma experiência digital rica e eficaz[Str21].

Esta ferramenta utiliza *Node.js* como tecnologia, disponibilizando uma série de ferramentas no que diz respeito à criação de uma API de dados. Começando pela possibilidade de modelar e criar uma base de dados, criando *collection types* com vários atributos, o que no caso do MySQL representa uma tabela. Nas propriedades de cada *collection* é possível definir as relações entre vários *collection types*, permitindo assim modelar a BD que for pretendida. Para além de oferecer esta facilidade ao programador, o *Strapi* também oferece funcionalidades como a gestão da autenticação e das permissões de cada tipo de utilizador, utilizando a norma *Json Web Token (jwt)*, que será explicitada abaixo.

Todas estas funções do *Strapi* irão traduzir-se em código, sendo possível customizar qualquer parte que for necessária, de acordo com o que se pretende na criação da API. A utilização deste sistema permite estudar e ter experiência com uma forma diferente de implementar um servidor de dados, em que em vez das ferramentas mais comuns em que é necessário construir tudo de raiz, o *Strapi* facilita principalmente na fase inicial de criação do CRUD da aplicação de dados.

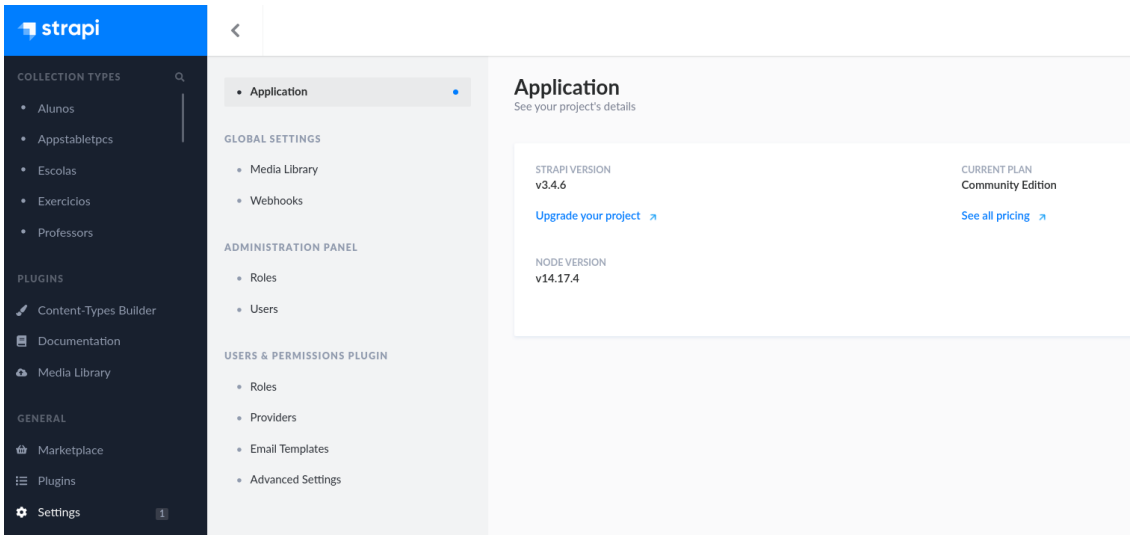


Figura 9: Painel de administração do *Strapi*

2.3.3 *JSON Web Token (JWT)*

Nesta secção será explicitado o funcionamento do JWT, que se trata de um padrão utilizado no processo de autenticação dos utilizadores de um sistema. Esta tecnologia é um dos formatos mais adotados no ambiente *web*, no que concerne à autenticação de uma aplicação, sendo que será implementada no componente TPC a desenvolver.

Definição

O *JSON Web Token* define uma norma compacta e segura de transmitir informação entre entidades, na forma de um objeto JSON. Este *token* é confiável e verificável pois é assinado digitalmente através de um segredo associado, ou através de um par de chaves pública/privada[Aut21].

O JWT é útil em cenários de autorização, em que após o utilizador autenticar-se o *token* é criado. Este é incluído em qualquer pedido do utilizador na aplicação, nomeadamente no acesso a rotas, serviços e outros recursos que sejam permitidos com aquele *token* em específico.

Para além da autorização, o *JSON Web Token* é uma boa forma de transmitir informação entre sistemas, pois sendo assinado com um par de chaves pública/privada é possível verificar o seu remetente, assim como garantir que a informação do *token* não foi alterada.

Estrutura

A estrutura do JWT divide-se em três partes: *Header* (Cabeçalho); *Payload* (Dados) e *Signature* (Assinatura).

Como tal será feita uma síntese de cada uma destas partes.

- *Header* (Cabeçalho)

O cabeçalho do *token* é constituído por dois elementos, nomeadamente o seu tipo que neste caso é JWT, assim como o algoritmo de assinatura utilizado.

Na página *web* do JWT[Aut21], podemos encontrar um exemplo de um *header* utilizando o algoritmo *HS256*, que é apresentado a seguir:

```
{
    "alg": "HS256",
    "typ": "JWT"
}
```

Exemplo 2.1: *Header* de um JWT

- *Payload* (Dados)

  O *payload* contém toda a informação que seja necessário guardar, como por exemplo o tipo de utilizador ou o próprio identificador.

  Estes dados são divididos em três tipos de atributos: registados (*Registered Claims*), públicos (*Public Claims*) e privados (*Private Claims*). Os registados são um conjunto de atributos que não sendo obrigatórios, são recomendados pois funcionam como um padrão comum a diferentes *token*. Como exemplo existe o *iss* (emissor), *exp* (tempo de expiração), *sub* (tema), entre outros.

  Os atributos públicos, são aqueles que são definidos por parte do programador, podendo adicionar os dados que achar necessário. Quanto aos privados, são atributos criados para partilhar informação entre entidades que convencionaram a sua utilização.

  No exemplo a seguir temos uma amostra de um *payload*, em que é utilizado o atributo registado *sub* como identificador do emissor, assim como estão definidos atributos como o nome e tipo do utilizador.

```
{
    "sub": "1234567890",
    "name": "John Doe",
    "admin": true
}
```

Exemplo 2.2: *Payload* de um JWT

- *Signature* (Assinatura)

  A parte final de criação do JWT é a assinatura. A assinatura é utilizada para verificar que a mensagem não foi alterada entre sistemas.

  Para tal é necessário codificar o *header* e *payload* no formato *Base64Url* e concatená-los. De seguida define-se o segredo e escolhe-se o algoritmo que irá realizar a assinatura, juntando todos os dados anteriores.

Na página do JWT[Aut21], temos um exemplo de criação de uma assinatura utilizando o algoritmo *HMACSHA256*:

```
HMACSHA256(
    base64UrlEncode(header) + "." +
    base64UrlEncode(payload),
    secret)
```

Exemplo 2.3: Algoritmo de assinatura de um JWT

2.3.4 *Vue.js*

Vue.js é uma *framework* progressiva de *JavaScript*, usada para construir interfaces de utilização (UI) e *Single-page Application* (SPA).

Esta *framework* ganhou relevância em desenvolvimento *web* devido à sua curva de aprendizagem rápida e a uma boa documentação, o que facilita na sua compreensão. Também utiliza o conceito de modularidade na forma de componentes que podem ser reutilizáveis, sendo que o *Vue* é facilmente adaptável em desenvolvimento para telemóveis (*mobile*). Disponibiliza várias ferramentas ao programador como sintaxe fácil de utilizar, *data binding*, diretivas, conceito de *props*, roteador, entre outros.

Ao contrário de outras *framework*s monolíticas, o *Vue* é desenhado desde a sua base para ser incrementalmente adotado. A sua principal biblioteca é focada na *view layer* (camada visual) apenas, e é facilmente integrada com outras bibliotecas ou projetos existentes[Vue21b].

Assim sendo, foi esta a tecnologia escolhida para a construção da interface do componente de TPC, pois disponibiliza todas as ferramentas para fazer cumprir os requisitos necessários. Para auxiliar na implementação da aplicação, também serão utilizadas as bibliotecas do *Vuetify* e *VueX* em complemento ao *Vue.js*.

Vuetify

O *Vuetify* é uma *framework* de interfaces completa, construída em cima do *Vue.js*. O objetivo desta biblioteca é fornecer aos programadores as ferramentas necessárias para construir uma experiência de utilização (UX) rica e funcional[Vue21d].

O desenvolvimento do *Vuetify* está de acordo com as especificações de *Material Design* em que cada componente é feito para ser modular, responsivo e otimizado. A aplicação pode ser construída com *layouts* únicos e dinâmicos, facilmente adaptáveis ao caso em estudo.

Para além do seu ecossistema ativo e consistente, o *Vuetify* tem uma aproximação focada no desenvolvimento *mobile*, o que nos possibilita que a aplicação funcione em computador, *tablet* ou telemóvel.

VueX

O *VueX* é uma biblioteca do *Vue.js*, que tem a função de gestão de estado. Serve como uma *Data Store* centralizada, com regras que garantem que o estado só possa ser mutado de forma correta[Vue21c]. Trata-se de um estado partilhado, em que é possivel aceder aos seus dados caso seja necessário, a partir de qualquer componente da interface.

Para o caso desta dissertação, esta biblioteca será principalmente relevante para gerir os dados do utilizador que se encontra autenticado na aplicação, nomeadamente guardando o seu JWT até ao término da sessão.

Chegando ao fim deste capítulo, foi dado a conhecer o Estado da Arte completo da Plataforma Hypatiamat.

Começou por fazer uma apresentação de projetos com um teor semelhante ao do Hypatiamat. De seguida foi explicitada toda a lógica por trás da Plataforma, assim como os conteúdos e recursos que são disponibilizados aos seus utilizadores.

Posteriormente foi feita referência aos diferentes dados que existem e que serão úteis a esta dissertação. Por fim, foram introduzidas as tecnologias necessárias ao trabalho a desenvolver.

O capítulo subsequente aborda tudo sobre o planeamento da construção do novo componente de TPC, nomeadamente as funcionalidades que terá a aplicação assim como o tipo de dados necessários.

3

MODELAÇÃO DO SISTEMA

O capítulo 3 desta dissertação, refere-se ao planeamento e modelação do sistema de TPC a desenvolver. Por conseguinte, esta modelação será constituída por diferentes partes, começando pela estrutura que irá ter a aplicação. Também será evidenciada a lógica e funcionalidades do novo componente, assim como o seu modelo de dados.

3.1 ESTRUTURA APLICACIONAL

Como já foi explicitado no capítulo 2, a aplicação de TPC tem uma estrutura diferente da Plataforma Hypatiamat. Ao contrário da anterior, esta será uma arquitetura de micro-serviços, com a existência de dois servidores. Um de *back-end* para a API de dados, fazendo a ligação com a BD, assim como um servidor *front-end* para servir a interface.

Esta aplicação, apesar de ter a sua própria estrutura e tecnologias independentes, é pensada para executar paralelamente à Plataforma Hypatiamat.

Figura 10: Arquitetura da aplicação de TPC

O utilizador ao aceder à aplicação no seu *browser*, tem acesso às páginas de interface que o servidor de *front-end* disponibiliza e este por sua vez faz pedidos de dados ao servidor da API, de forma a ser fornecida toda a informação necessária da base de dados.

3.2 REQUISITOS FUNCIONAIS

Nesta secção é iniciado o estudo e levantamento de todos os requisitos necessários à criação da aplicação de TPC.

No que diz respeito às funcionalidades que a aplicação terá que ter, podemos desde já dividir estas pelos seus dois tipos de utilizadores, ou seja Professores e Alunos.

3.2.1 *Requisitos do Professor*

Para o utilizador Professor, são a seguir listados os requisitos necessários às suas funções na aplicação:

- Um professor pode autenticar-se no sistema com as suas credenciais;
- Um professor autenticado pode proceder à criação de TPC;
- Um professor autenticado pode definir um título para o TPC;
- Um professor autenticado pode determinar um n.º de tentativas de resolução para o TPC;
- Um professor autenticado pode visualizar as suas turmas, para o ano letivo atual;
- Um professor autenticado pode escolher quais das suas turmas devem resolver o TPC;
- Um professor autenticado pode visualizar os alunos de cada turma;
- Um professor autenticado pode escolher alunos em específico de uma turma, para realizarem o TPC;
- Um professor autenticado pode definir uma data limite de resolução do TPC;
- Um professor autenticado pode definir uma hora limite de resolução do TPC;
- Um professor autenticado pode visualizar a lista de temas relativos a tarefas;
- Um professor autenticado pode selecionar um tema da lista de temas;
- Um professor autenticado pode visualizar a lista de subtemas, de um tema em específico;
- Um professor autenticado pode selecionar um subtema da lista de subtemas;
- Um professor autenticado pode visualizar e navegar pelo conjunto de tarefas, do par tema/subtema selecionado;
- Um professor autenticado pode escolher visualizar apenas tarefas de exame;
- Um professor autenticado pode escolher visualizar apenas tarefas de um certo nível de dificuldade;
- Um professor autenticado pode adicionar uma tarefa que pretenda, à lista de tarefas do TPC;
- Um professor autenticado pode adicionar o n.º de tarefas que pretender ao TPC;
- Um professor autenticado pode aceder a configurações avançadas do TPC;

- Um professor autenticado pode selecionar a configuração que define uma ordem aleatória das tarefas do TPC, para cada aluno;
- Um professor autenticado pode selecionar a configuração que define o impedimento de retroceder na lista de tarefas, ou seja, o aluno ao responder a uma tarefa e avançar para a próxima fica impedido de alterar respostas anteriores;
- Um professor autenticado pode selecionar a configuração que define o impedimento de acesso à resolução, ou seja, impede o aluno de ter acesso à solução do TPC, após este expirar;
- Um professor autenticado pode aceder à listagem de TPC criados, que se encontram ativos;
- Um professor autenticado pode aceder à listagem de TPC criados, que se encontram expirados;
- Um professor autenticado pode visualizar a configuração de um TPC criado;
- Um professor autenticado pode aceder à proposta de resolução para cada tarefa de um TPC criado;
- Um professor autenticado pode eliminar um TPC criado, removendo todas as resoluções por parte de alunos;
- Um professor autenticado pode visualizar a tabela de alunos que resolveram o TPC;
- Um professor autenticado pode visualizar as informações de cada aluno que resolveu o TPC, como número, nome, turma, n.º de tentativas, n.º questões corretas e classificação;
- Um professor autenticado pode aceder às respostas de um aluno, para cada tarefa do TPC;
- Um professor autenticado pode visualizar estatísticas gerais dos alunos, ao escolher um ano letivo, uma turma e se pretende as estatísticas para todos os TPC ou para um em específico;
- Um professor autenticado pode aceder ao *Backoffice* do Hypatiamat (componente distinto);
- Um professor autenticado tem acesso a ajudas de como utilizar a página, para todas as páginas *web* da aplicação;
- Um professor autenticado pode terminar sessão no sistema.

3.2.2 *Requisitos do Aluno*

No que diz respeito ao tipo de utilizador Aluno, temos os seguintes requisitos:

- Um aluno pode autenticar-se no sistema com as suas credenciais;
- Um aluno autenticado pode aceder à listagem de TPC ativos;
- Um aluno autenticado pode proceder à resolução de um TPC da listagem de ativos;
- Um aluno autenticado pode refazer um TPC, se este estiver ativo e houver possibilidade de repetição;
- Um aluno autenticado pode responder às tarefas do TPC, uma de cada vez;
- Um aluno autenticado pode alterar a resposta a uma tarefa, se assim for possibilitado no TPC;
- Um aluno autenticado pode aceder à listagem de TPC expirados;
- Um aluno autenticado pode ver a classificação obtida para um TPC expirado;
- Um aluno autenticado pode visualizar a proposta de solução para cada tarefa de um TPC expirado, caso seja possibilitado;
- Um aluno autenticado pode visualizar a sua última resolução e respetiva correção de um TPC expirado, caso seja possibilitado;
- Um aluno autenticado pode visualizar as suas estatísticas gerais, para todos ou para um TPC em específico;
- Um aluno autenticado pode aceder ao *Backoffice* do Hypatiamat (componente distinto);
- Um aluno autenticado tem acesso a ajudas de como utilizar a página, para todas as páginas *web* da aplicação;
- Um aluno autenticado pode terminar sessão no sistema.

3.3 DIAGRAMA DE USE CASES

Com o intuito de resumir e oferecer uma forma de visualização dos requisitos supracitados, foi criado um diagrama de casos de uso. Este diagrama é divido em dois diagramas, sendo cada um relativo a um ator do sistema, que neste caso são Professor e Aluno.

Para efeitos de simplificação, foram ignorados os casos de uso de autenticação e de terminar sessão, comuns aos dois tipos de utilizadores.

3.3.1 *Diagrama do Professor*

Funcionalidades do Professor explicitadas no diagrama da figura:

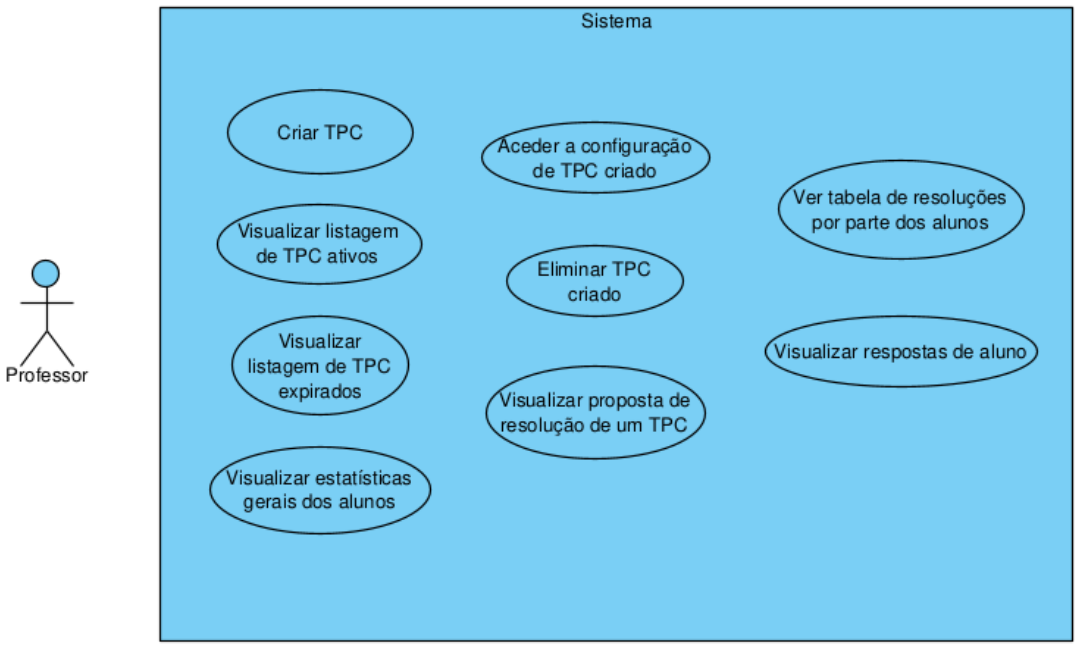


Figura 11: Diagrama de Use Cases - Professor

3.3.2 *Diagrama do Aluno*

Funcionalidades do Aluno explicitadas no diagrama da figura:

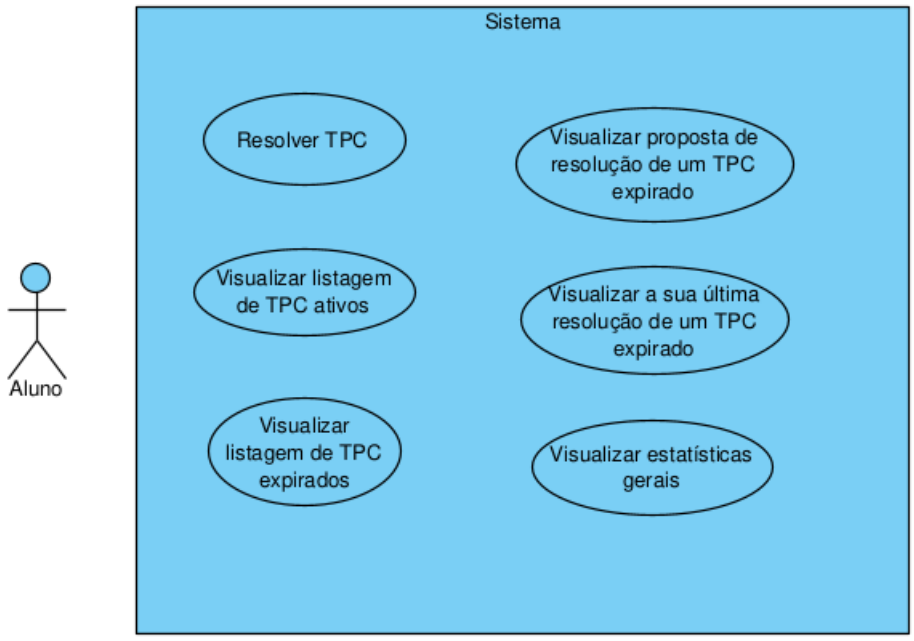


Figura 12: Diagrama de Use Cases - Aluno

3.4 DIAGRAMAS DE ATIVIDADE

Todo o estudo anterior permitiu clarificar as funcionalidades que a aplicação de TPC deve ter, no entanto, é possível destacar duas como as que têm maior importância no componente. Estas duas funcionalidades são a criação de TPC, por parte do utilizador Professor e a de resolver o TPC, por parte do utilizador Aluno.

Como tal, fazendo esta distinção entre as funcionalidades, foram concebidos dois diagramas de atividade que pretendem planificar estas ações relevantes por parte dos utilizadores da aplicação.

3.4.1 *Diagrama Criar TPC - Professor*

Na figura seguinte é apresentado o diagrama de Criar TPC, por parte do Professor:

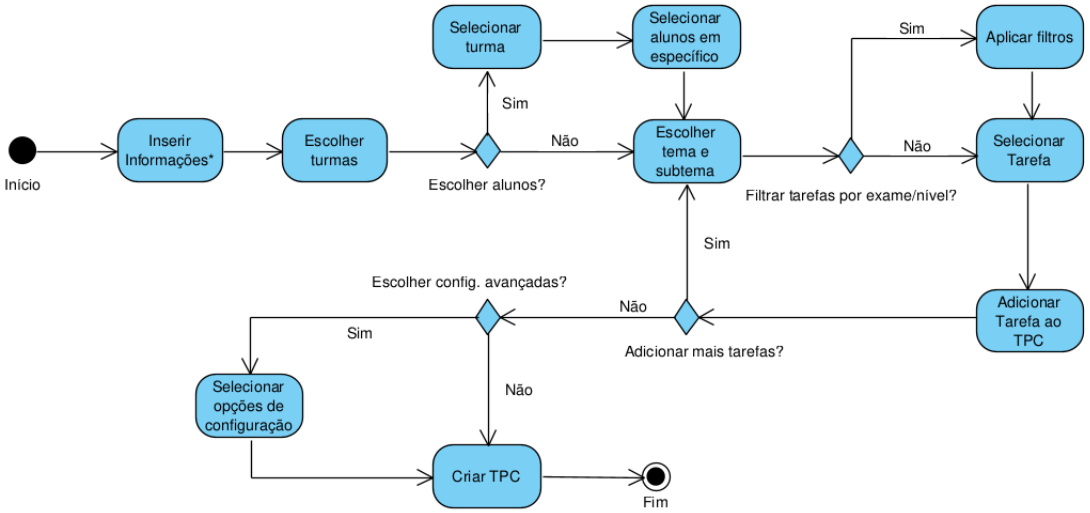


Figura 13: Diagrama de Atividade - Criar TPC

3.4.2 *Diagrama Resolver TPC - Aluno*

A seguir também temos, por parte do aluno, o diagrama de Resolver TPC:

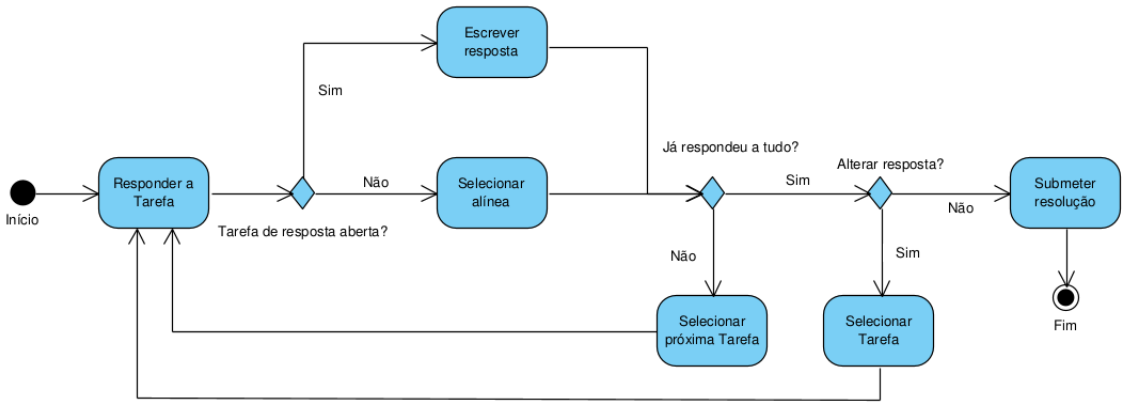


Figura 14: Diagrama de Atividade - Resolver TPC

3.5 Criação de Nova Base de Dados

Após análise e estudo das funcionalidades pretendidas para aplicação e tendo em consideração os modelos já existentes no Hypatiamat, surgiu a necessidade de criar uma nova base de dados e adicioná-la ao restante sistema da Plataforma.

Esta BD foi modelada com o intuito de armazenar os dados das entidades novas que a aplicação de TPC irá introduzir, assim como as relações que existem entre estas.

3.5.1 *Modelos a Adicionar*

A seguinte secção aborda os modelos relativos a toda a criação e gestão dos TPC. Tal como anteriormente, um modelo representa os dados de uma determinada entidade do sistema, sendo cada um destes equivalente a uma tabela de BD.

Os modelos desta nova BD e seus atributos, estão explicitados abaixo. É importante realçar que nesta parte só serão identificados os atributos base de cada modelo, sendo que também existem atributos provenientes das relações, que serão mencionados nas secções a seguir.

TPC

Esta é a entidade principal da nova base de dados, que irá ter toda a informação necessária de cada TPC. Os seus atributos serão os seguintes:

- *id* - Identificador único (numérico);
- *codProf* - Código identificador do professor;
- *tentativas* - N.º de tentativas;
- *tagname* - Título do trabalho;
- *dataInicio* - Data de criação;
- *dataFim* - Data de expiração;
- *configAleatoria* - *flag* que indica se configuração aleatória está ativa;
- *configRetroceder* - *flag* que indica se configuração retroceder está ativa;
- *configResolucao* - *flag* que indica se configuração resolução está ativa.

TPC-questões

A tabela *TPC-questões* representa as tarefas/questões que fazem parte do TPC. A nível de atributos, temos apenas:

- *id* - Identificador único (numérico);
- *codQuestao* - Código identificador da questão/tarefa.

TPC-alunos

Este modelo, em semelhança com o anterior, representa os alunos que fazem parte do TPC. Quanto às suas propriedades, são:

- *id* - Identificador único (numérico);

- *codAluno* - Código identificador do aluno;
- *codTurma* - Código identificador da turma;
- *nome* - Nome do aluno;
- *numero* - Número do aluno.

Resoluções

A entidade *Resoluções*, servirá para armazenar os dados das resoluções de cada aluno, a um respetivo TPC. Os seus atributos são os seguintes:

- *id* - Identificador único (numérico);
- *idTpc* - Identificador do TPC;
- *qRespondidas* - N.º total de questões respondidas;
- *qCertas* - N.º de questões respondidas de forma correta;
- *data* - Data em que se realizou;
- *classificacao* - Classificação obtida;
- *tentativa* - N.º da tentativa de resolução.

Respostas

Finalmente, a última entidade é *Respostas*, que representa as respostas de cada resolução por parte do aluno. Tem como propriedades:

- *id* - Identificador único (numérico);
- *codQuestao* - Código identificador da tarefa/questão;
- *resposta* - Resposta do aluno à tarefa;
- *correta* - *flag* que indica se a resposta está correta.

3.5.2 *Modelo E-R*

Finalizada a introdução das entidades, a BD foi esquematizada através da criação de um modelo entidade-relação (E-R). Este diagrama permite representar as entidades e os seus respetivos atributos, assim como as relações entre cada uma.

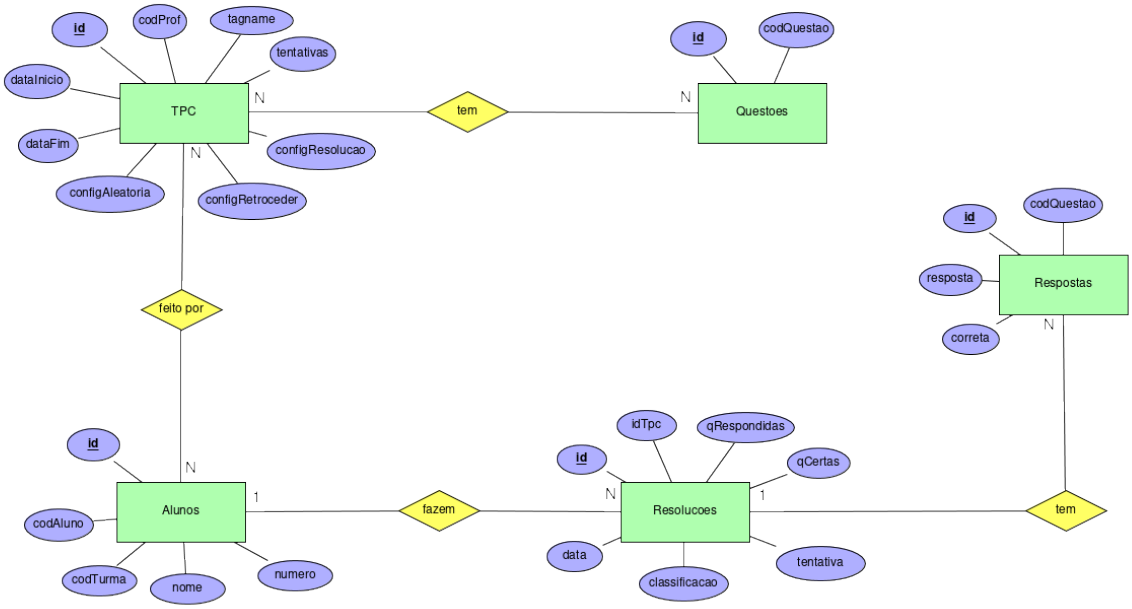


Figura 15: Modelo E-R da base de dados

Através da figura são evidenciadas as relações entre as entidades, considerando a tabela *TPC* como o componente principal. O *TPC* tem várias *Questões* assim como estas têm múltiplos *TPC* (relação N-N), sendo que o mesmo se aplica entre o *TPC* e os *Alunos*. Por sua vez, os *Alunos* realizam uma ou mais *Resoluções*, sendo estas exclusivas ao aluno (relação 1-N). A tabela *Resoluções* tem também uma relação 1-N com as *Respostas*, visto que uma resolução poderá ter uma ou várias respostas associadas.

De forma a proceder à criação da BD, serão utilizados os mecanismos do *Strapi*, que como já havia sido mencionado no estudo das tecnologias, permite modelar os dados do nosso sistema. Este processo será explicitado no capítulo a seguir sobre a implementação da API de dados.

O capítulo que agora termina, permitiu entender como foi modelado o sistema e a teoria que servirá de base ao componente de TPC.

Primeiramente foi analisada a estrutura e respetivos serviços que constituirão a aplicação a desenvolver. A seguir, foi planificado todo o conhecimento e lógica do sistema através de requisitos e diagramas. Os diagramas permitiram visualizar as funcionalidades do sistema, separadas pelos dois tipos de utilizador existentes.

A última secção abordou a modelação da nova BD de suporte ao componente de TPC. Nomeadamente, introduzindo todas as entidades que constituem o modelo e seus respetivos dados. Através de um modelo E-R, foi possível evidenciar todas as relações entre as entidades do sistema.

No capítulo seguinte, será explicado todo o processo de implementação da API de dados, com base na modelação anterior.

4

DESENVOLVIMENTO DA API

O capítulo de desenvolvimento da API descreverá o processo de implementação do servidor de dados, tendo em conta todo o estudo e planeamento realizado no capítulo anterior.

Serão explicitadas diferentes fases do desenvolvimento, desde a definição das configurações iniciais, introdução do modelo de dados e construção de rotas de disponibilização dos dados. Também será explicado o processo de autenticação e respetiva proteção do acesso às rotas.

4.1 INICIALIZAÇÃO NO STRAPI

No desenvolvimento da API será utilizado o *Strapi* e as suas ferramentas, o que irá permitir construir a nova base de dados, todas as rotas necessárias ao funcionamento da aplicação de TPC, assim como gerir a autenticação.

O *Strapi* possui um *plugin* denominado *Content Type Builder*, que oferece a possibilidade de modelar a estrutura dos dados da API. Esta ferramenta permite criar coleções (*collections*) que equivalem a tabelas da BD, em que para cada uma são adicionados os atributos, assim como as relações intrínsecas.

4.1.1 *Novos Modelos*

Primeiramente serão evidenciados os modelos pertencentes à nova BD, sendo que depois passaremos às restantes entidades.

Começando pela entidade principal da base de dados, temos o *TPC*:

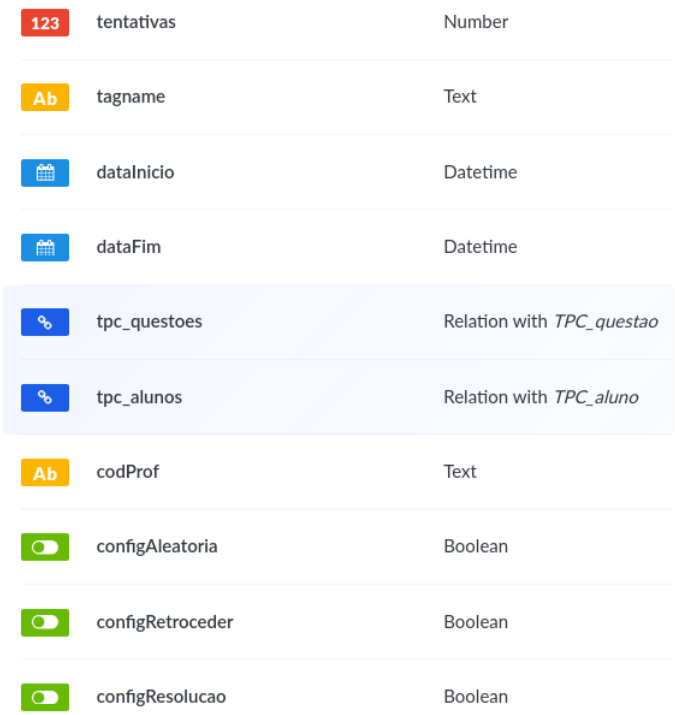


Figura 16: *Strapi Collection* - TPC

Como é possível ver através da figura, a *collection* possui todos os atributos existentes no modelo E-R anterior. Para além dos atributos criou-se duas relações, uma com a entidade *TPC-Questões* e outra com a *TPC-Alunos*. Estas relações estão de acordo com a modelação, sendo ambas relações N-N e estão exemplificadas nas figuras a seguir.

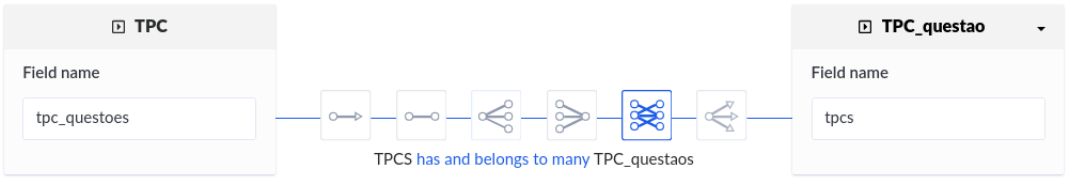


Figura 17: *Relação entre TPC e Questões*

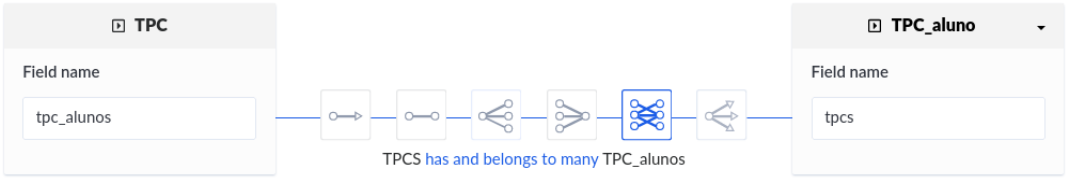


Figura 18: *Relação entre TPC e Alunos*

De seguida, passamos à entidade *TPC-Questões*:

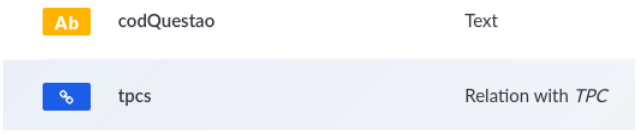


Figura 19: *Strapi Collection* - Questões (TPC)

Esta é a entidade mais pequena do modelo e também possui uma relação com o *TPC*, pois como se trata de uma relação N-N, é bidirecional às duas coleções.

Continuando o processo de inicialização, segue-se a entidade *TPC-Alunos*:

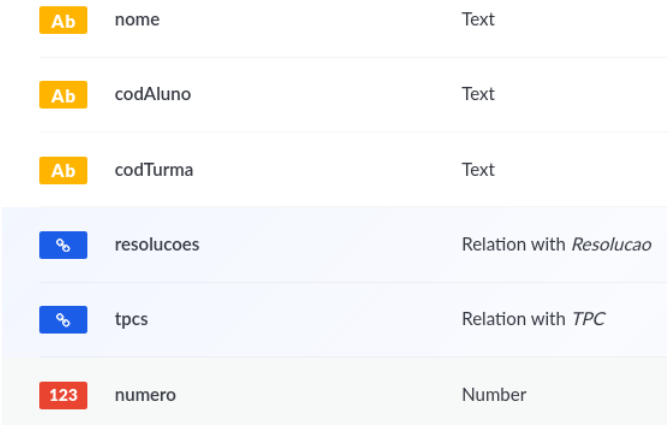


Figura 20: *Strapi Collection* - Alunos (TPC)

Novamente, a coleção possui todos os atributos necessários assim como duas relações, uma com o *TPC* e outra com as *Resoluções*. A relação com o *TPC* é originária da sua propriedade bidirecional, tal como acontecia na coleção *Questões*. No que diz respeito à relação com as *Resoluções*, é uma relação 1-N unidirecional, como está evidenciado na figura a seguir.

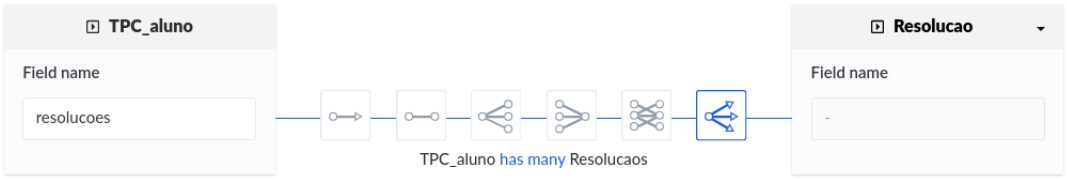


Figura 21: *Relação entre Alunos e Resoluções*

De seguida, analisamos a entidade *Resoluções*:

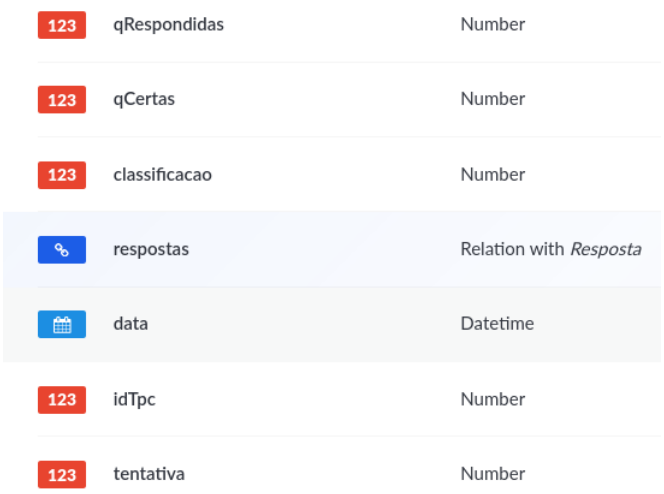


Figura 22: *Strapi Collection* - Resoluções (TPC)

A tabela *Resoluções*, para além dos seus atributos, possui uma relação 1-N modelada com a entidade *Respostas*. A relação está especificada na figura abaixo.

Figura 23: *Relação entre Resoluções e Respostas*

Finalmente, apresenta-se a última entidade da nova BD, denominada *Respostas*:

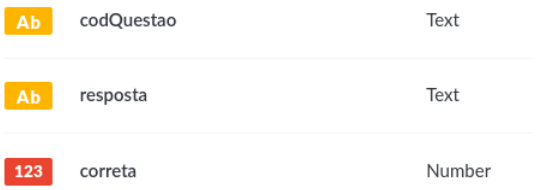


Figura 24: *Strapi Collection* - Respostas (TPC)

A coleção *Respostas* apenas possui atributos, visto que a relação unidirecional a que pertence é feita a partir das *Resoluções*, como foi apresentado anteriormente.

Todas as relações modeladas a partir do *Strapi*, serão úteis a nível da ligação de dados entre diferentes entidades. Por exemplo ao pedir a informação de um TPC, esta virá unida com os dados das suas respetivas questões e alunos, devido à existência das relações entre estes modelos.

4.1.2 *Modelos Existentes*

Com a adição das coleções anteriores, foi finalizada a criação da nova base de dados que funcionará em conjunto com as já existentes no Hypatiamat.

Sendo assim, apenas falta adicionar os modelos necessários provenientes destas outras BD do Hypatiamat, que foram explicitados no capítulo do Estado da Arte.

Começando pelos modelos da BD *Aplicações*, temos as entidades *Alunos*, *Professores*, *Escolas* e *Turmas*. É importante realçar que estas coleções não possuem relações entre elas, visto que essa modelação foi feita anteriormente a esta dissertação, não tendo sido utilizado o *Strapi*.

Coleção Alunos

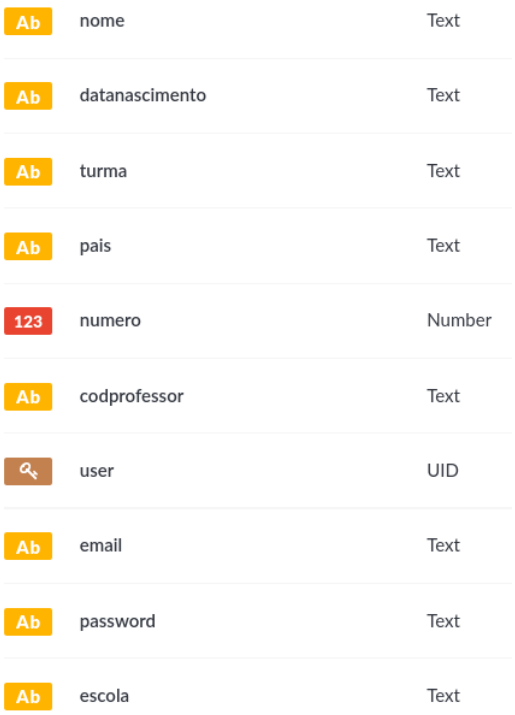


Figura 25: *Strapi Collection* - Alunos (Aplicações)

Coleção Professores

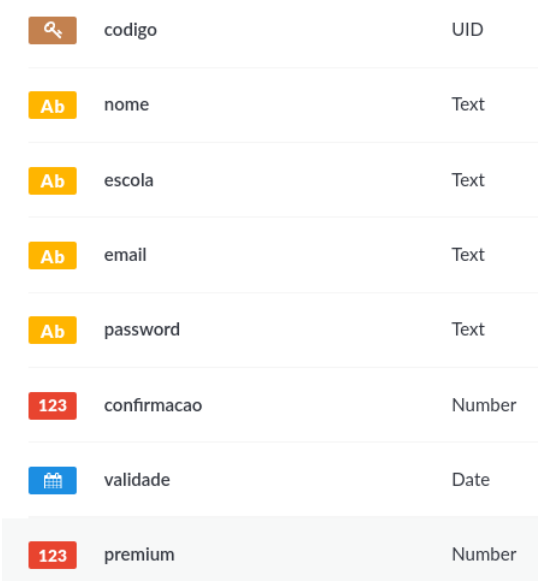


Figura 26: *Strapi Collection* - Professores (Aplicações)

Coleção Escolas

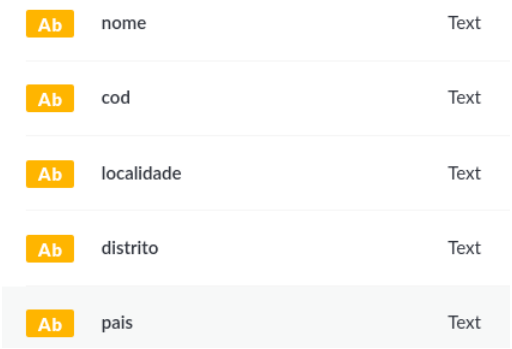


Figura 27: *Strapi Collection* - Escolas (Aplicações)

Coleção Turmas

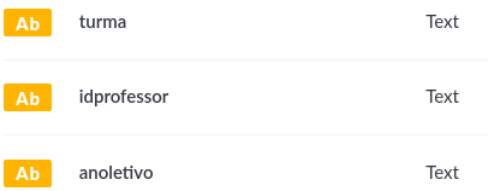


Figura 28: *Strapi Collection* - Turmas (Aplicações)

Estas foram as coleções adicionadas da BD *Aplicações*, necessárias ao funcionamento da API de dados.

No que concerne à base de dados *Teste de Conhecimentos*, temos os modelos *Exercícios* (tarefas/questões) e *Temas*. Os modelos *Appstable* e *Appsinfoall* serão falados numa secção posterior, pois não contêm dados relevantes ao funcionamento do componente de TPC, apenas servem para monitorizar o desempenho na aplicação.

Coleção Exercícios

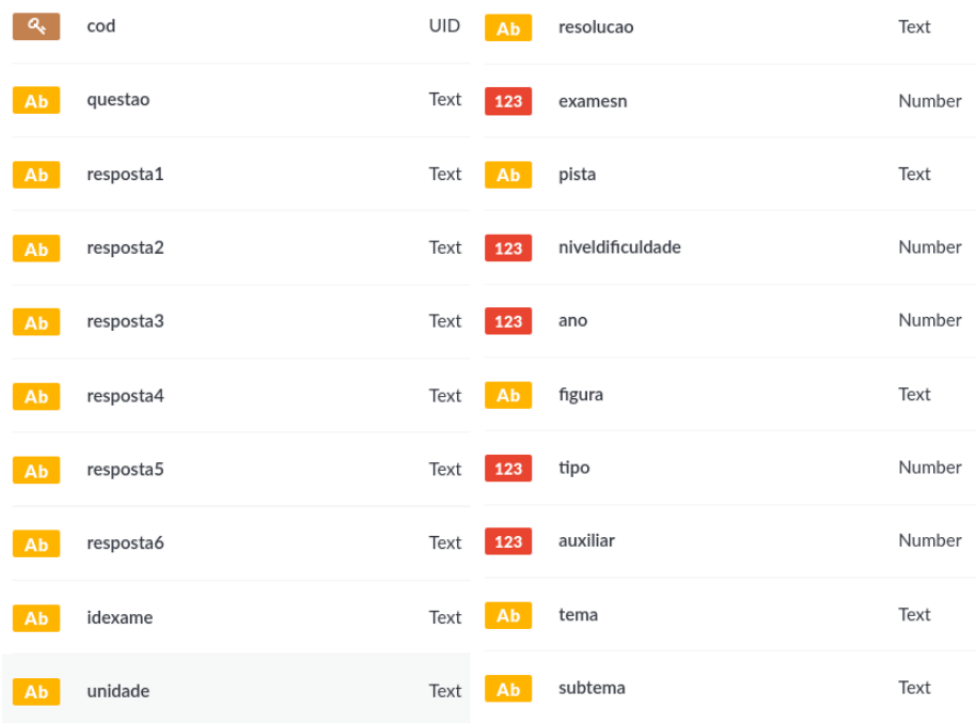


Figura 29: *Strapi Collection* - Exercícios (Teste de Conhecimentos)

Coleção Temas

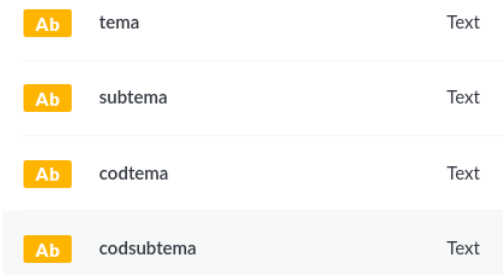


Figura 30: *Strapi Collection* - Temas (Teste de Conhecimentos)

4.2 CONSTRUÇÃO DA API DE DADOS

A parte de inicialização da API foi terminada na secção acima, sendo que agora será abordada a sua construção, que irá permitir fornecer todos os dados necessários à aplicação de TPC.

4.2.1 *Rotas implementadas*

Com a criação das coleções, o *Strapi* desenvolveu o CRUD (criação, leitura, edição e eliminação) para cada uma destas. Estão portanto reunidas as operações base necessárias para iniciar a construção e desenvolvimento das rotas da API de dados, de acordo com os requisitos estudados.

Deste modo serão explicitadas as rotas construídas e utilizadas na aplicação, separadas por grupos consoante o seu tipo de prefixo.

BD Aplicações

- **Login** (Prefixo: /auth)
  - *POST*
    - /local - Dadas as credenciais de um utilizador, se estiverem corretas, devolve um *token* assim como os seus dados associados.
- **Alunos** (Prefixo: /alunos)
  - *GET*
    - /:user - Devolve a informação de um aluno, passando o seu código de utilizador como parâmetro.
- **Escolas** (Prefixo: /escolas)
  - *GET*
    - /:cod - Devolve a informação de uma escola, passando o seu código de identificação como parâmetro.
- **Professores** (Prefixo: /professores)
  - *GET*
    - /:codigo - Devolve a informação de um professor, passando o seu código de utilizador como parâmetro.
- **Turmas** (Prefixo: /turmas)
  - *GET*
    - /prof/:idprofessor - Devolve as turmas de um professor (passando como parâmetro o seu código de utilizador) e respetivos alunos, sendo possível filtrar por turmas do ano letivo atual através da *Query String ano* (*last*);

    * /prof/:idprofessor/ano - Devolve todos os identificadores das turmas de um professor organizados por ano letivo, sendo passado como parâmetro o código do professor.

BD Teste de Conhecimentos

- **Appstabletpcs** (Prefixo: /appstabletpcs)
  - *GET*
    * / - Devolve todas as entradas da tabela de monitorização do desempenho, ou se for passado como *Query String* o código de subtema, devolve essa entrada em específico.
  - *POST*
    * / - Insere nas tabelas de monitorização o desempenho de um aluno, sendo que os argumentos colocados no *body* do pedido são: *table, user, turma, codprof, escola, tema, subtema, frame, data, certas, total, tempo.*
- **Exercícios** (Prefixo: /exercicios)
  - *GET*
    * / - Devolve todos os exercícios da BD, agrupados pelo código do seu subtema;
    * /:cod - Devolve a informação de um dado exercício, passando como parâmetro o seu código de identificação.
- **Temas** (Prefixo: /temas)
  - *GET*
    * / - Devolve todos os subtemas da BD, agrupados pelo código do tema correspondente.

BD TPC

- **TPC** (Prefixo: /tpcs)
  - *GET*
    * /:id - Devolve a informação de um dado TPC, passando como parâmetro o seu identificador;
    * /prof/:codProf - Devolve todos os TPC criados por um professor específico (através da passagem do seu código de utilizador como parâmetro) e filtra por data de expiração utilizando a *Query String time* (*active* ou *expired*);
    * /prof/:codProf/turmas - Devolve todos os identificadores de TPC criados por um determinado professor, divididos por turma (passando o código do professor como parâmetro);
    * /prof/:codProf/stats - Devolve as estatísticas de todos os TPC criados por um professor (através do seu código como parâmetro) e para uma dada turma (passando o código da turma como *Query String*);

    * /:tpcId/stats - Devolve as estatísticas de alunos para um TPC específico, passando o seu identificador como parâmetro.

  - *POST*

    * / - Insere um novo TPC, em que os argumentos passados no *body* são: *tagname, codProf, tentativas, dataInicio, dataFim, tpc_questoes (lista de id), tpc_alunos (lista de id), configAleatoria, configRetroceder, configResolucao.*

  - *DELETE*

    * /:id - Elimina um determinado TPC, através do seu identificador como parâmetro.

- **TPC-Questões** (Prefixo: /tpc-questoes)
  - *POST*

    * / - Insere uma nova questão de TPC, sendo que o *body* enviado contém apenas o *codQuestao.*

- **TPC-Alunos** (Prefixo: /tpc-alunos)
  - *GET*

    * /:codAluno - Devolve toda a informação acerca de um aluno, incluindo os seus TPC e resoluções, utilizando como parâmetro o seu código de utilizador;

    * /:codAluno/tpcs - Devolve a lista de TPC de um dado aluno, passando no parâmetro o seu código de utilizador, sendo possível filtrar por data de expiração através da *Query String time* (*active* ou *expired*);

    * /:codAluno/stats - Devolve as estatísticas relativas a todos os TPC feitos por um determinado aluno, utilizando o seu código como parâmetro;

    * /:codAluno/tpcs/:tpcId/stats - Devolve as estatísticas relativas a um determinado TPC feitos por um dado aluno, passando como parâmetros ambos o código de aluno assim como o identificador do TPC.

  - *POST*

    * / - Insere um novo aluno de TPC, em que o *body* enviado contém os seguintes argumentos: *codAluno, nome, numero, codTurma.*

  - *PUT*

    * /:codAluno - Altera a informação de um aluno (passando no parâmetro o seu código), nomeadamente no que diz respeito aos seus TPC e resoluções, sendo que no *body* enviado temos: *tpcs, resolucoes* no formato de lista de *id.*

- **Resoluções** (Prefixo: /resolucoes)
  - *GET*

    * /:id - Devolve a informação de uma determinada resolução, através do respetivo identificador como parâmetro.

- *POST*
  * / - Insere uma nova resolução de TPC, sendo o *body* do pedido o seguinte: *idTpc, tentativa, qRespondidas, qCertas, classificacao, respostas, data.*
- *DELETE*
  * /:id - Elimina uma determinada resolução, utilizando o seu identificador como parâmetro.

- **Respostas** (Prefixo: /respostas)
  - *POST*
    * / - Insere uma nova resposta de resolução, em que o *body* do pedido tem como argumentos: *codQuestao, resposta, correta.*
  - *DELETE*
    * /:id - Elimina uma dada resposta, através do seu identificador como parâmetro.

4.2.2 *Autenticação*

Após a construção das rotas da API, foi necessário configurar a autenticação do sistema.

Como foi mencionado no estudo das tecnologias, o *Strapi* disponibiliza um *plugin* denominado *Roles & Permissions*, que serve para proteger a API através de um processo de autenticação baseado em JWT. Esta ferramenta já vem implementada com todas as funções relativas à utilização de JWT, sendo apenas preciso definir o segredo e o tempo de expiração do mesmo. Estas definições foram configuradas da seguinte maneira:

```
module.exports = {
    jwtSecret: process.env.JWT_SECRET || "******",
};
```

Exemplo 4.1: Segredo do JWT

```
{
    "jwt": {
        "expiresIn": 5400
    }
}
```

Exemplo 4.2: Tempo de expiração do JWT (em segundos)

Através dos exemplos podemos ver a definição do segredo que está oculto, sendo que o tempo de expiração do *token* está definido para 5400 segundos, o que equivale a uma hora e meia.

De seguida, foi alterado o controlador de autenticação para adaptar o *payload* do *token* por cada tipo de utilizador. Os tipos de utilizador da aplicação de TPC são apenas dois, aluno e

professor. Contudo a Plataforma Hypatiamat possui mais tipos de utilizador para além destes, como município, agrupamento e administrador, sendo assim necessário definir os diferentes níveis de utilização no *payload* do JWT, de forma a uniformizar esta informação entre todos os componentes do Hypatiamat.

Utilizador	**Tabela**	***Premium***	**Tipo**	***Payload***
Aluno	*alunos*	Não possui	10	Id do utilizador Código de utilizador Email Código de escola Nome de agrupamento Tipo
Professor	*professores*	1	20	Id do utilizador Código de utilizador Email Código de escola Nome de agrupamento Tipo
Município	*professores*	2	30	Id do utilizador Código de utilizador Email Tipo Informação de Escola Escolas do Município
Agrupamento	*professores*	3	40	Id do utilizador Código de utilizador Email Código de escola Nome de agrupamento Tipo
Administrador	*professores*	5	50	Id do utilizador Código de utilizador Email Código de escola Nome de agrupamento Tipo

Tabela 1: *Payload* do *token* por utilizador

Visualizando a tabela acima, podemos verificar com que informação é criado o JWT na autenticação de cada tipo de utilizador, consoante os seus diferentes atributos. Em termos do

funcionamento na aplicação de TPC, todos estes tipos de utilizadores são agrupados pela sua tabela de BD correspondente (como é visível na coluna *tabela*), dando origem aos dois perfis de utilização, professores e alunos.

4.2.3 *Proteção de rotas*

Concluído o processo de autenticação no sistema e respetiva geração de JWT, é necessário proteger as rotas da API. Para tal, quase todos os pedidos à API deverão ser acompanhados do devido *token* de autorização, caso contrário o pedido não será realizado.

Em paralelo com a secção anterior, o *Strapi* no *plugin Roles & Permissions* também oferece uma ferramenta de gestão de permissões e proteção de rotas. Para uma rota protegida, a ferramenta verifica se o pedido tem o *token* associado e se o respetivo utilizador tem as permissões adequadas de acesso.

A configuração de proteção de rotas é feita na interface do *Strapi*, sendo divididas em rotas públicas e rotas protegidas. Os pedidos a rotas públicas não necessitam de autenticação, enquanto que as protegidas necessitam de autenticação do utilizador.

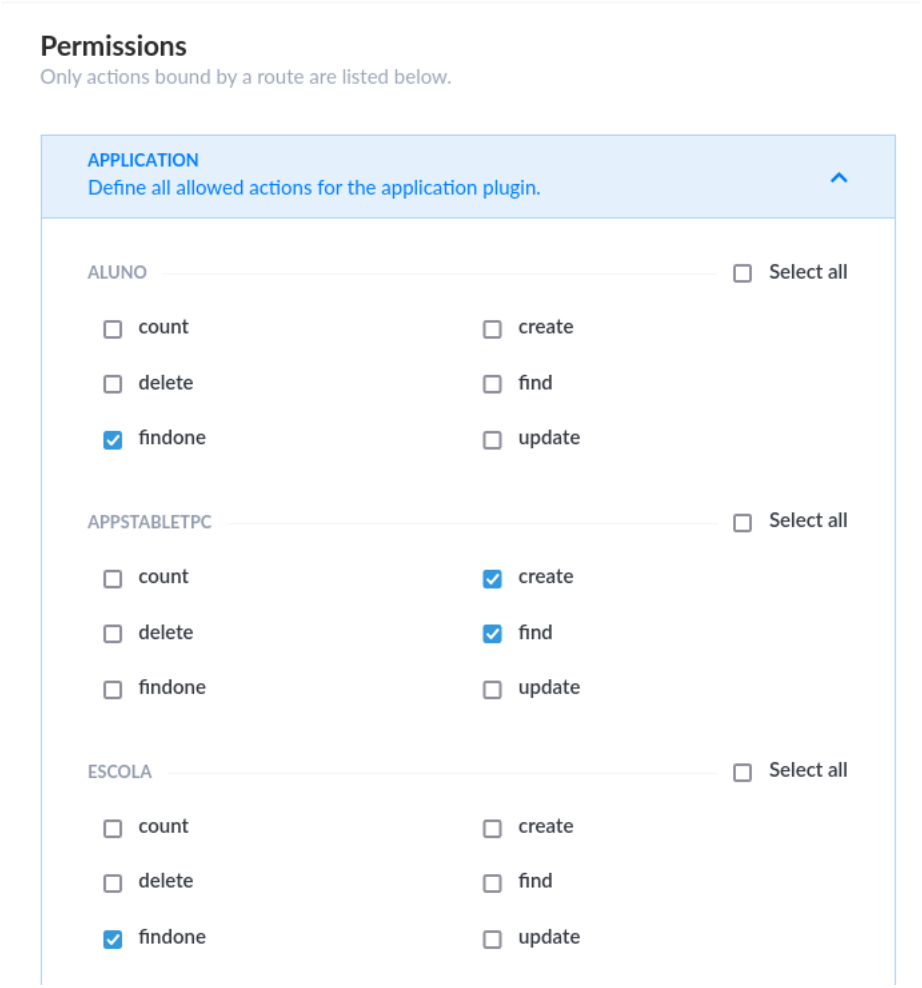


Figura 31: *Strapi* - Gestão de Permissões

Através do painel apresentado na figura, são definidas as permissões de todas as rotas da API de dados. A única rota que está definida como pública é a de *login* (autenticação), sendo que todas as restantes rotas da aplicação estão protegidas, logo é necessário o utilizador estar autenticado.

4.3 Monitorização de Desempenho

A Plataforma Hypatiamat realiza uma monitorização do desempenho dos seus alunos, durante a utilização das diferentes aplicações e conteúdos disponíveis. Com isto, foi requerido criar essa mesma monitorização para o novo componente, aquando da realização dos TPC por parte dos alunos.

Existem três tabelas da BD *Teste de Conhecimentos*, que são a base desta monitorização, sendo elas *appstabletpc*, *appsinfo* e *appinfoall*. Começando por estas duas últimas, as tabelas *appsinfo* e *appsinfoall* possuem a mesma estrutura de atributos:

- *Id* - Identificador único;
- *grupo* - Código de tema da tarefa;
- *appId* - Código de subtema da tarefa;
- *ncertas* - N.º de tarefas corretas;
- *ntotal* - N.º total de tarefas resolvidas;
- *lastdate* - Data da última tarefa resolvida;
- *userId* - Código de utilizador do aluno;
- *horario* - Hora da última tarefa resolvida;
- *onpeak* - N.º de tarefas resolvidas em hora de pico;
- *offpeak* - N.º de tarefas resolvidas fora da hora de pico;
- *codProf* - Código do professor responsável;
- *turma* - Código da turma do aluno.

Sabendo que as tarefas da aplicação estão agrupadas por tema e subtema, as entradas destas tabelas correspondem às resoluções de um determinado aluno, a um grupo de tarefas de um dado par tema/subtema.

A diferença entre as duas, é que a *appsinfo* guarda o desempenho totalizado para um dado conjunto de tema/subtema, atualizando sempre os dados com a última resolução de tarefa. Por outro lado a *appsinfoall* mantém um acompanhamento diário, ou seja para cada dia guarda o desempenho do aluno aquando da resolução de tarefas.

Quanto à tabela *appstabletpc*, serve como apontador para uma quantidade extensa de outras tabelas da BD, em que cada uma diz respeito a um conjunto tema/subtema. Após o aluno concluir as tarefas, o sistema vai à *appstabletpc* e verifica para cada uma qual a tabela a atualizar. A estrutura dessas tabelas é comum entre elas e é a seguinte:

- *Id* - Identificador único;
- *user* - Código de utilizador do aluno;
- *turma* - Código de turma;
- *codprof* - Código do professor;
- *escola* - Código de escola;
- *tema* - Código do tema;
- *subtema* - Código do subtema;
- *frame* - *flag* que indica o código da tarefa;
- *data* - Data em que realizou tarefa;
- *certas* - N.º de tarefas corretas:
- *total* - N.º total de tarefas resolvidas.

Após este pequeno estudo, foi implementado na API de dados todo este processo de monitorização. Através desta funcionalidade, a Plataforma Hypatiamat além de acompanhar a utilização nos seus diferentes componentes, passa também a recolher os dados relevantes sobre a aplicação de TPC.

Com o finalizar do capítulo de desenvolvimento da API, dá-se por terminada a clarificação de todo o processo de implementação do servidor de dados.

Começando pela inicialização do servidor no *Strapi*, onde foram introduzidas as coleções da BD, respetivos atributos e relações. Assim sendo, foram construídas as rotas tendo por base as funcionalidades idealizadas e de forma a disponibilizar os dados necessários.

Foi também explicitada a autenticação por parte dos utilizadores do sistema e respetiva configuração de proteção das rotas. Por fim, foi abordada a implementação da monitorização no componente de TPC, que fornece os dados de desempenho à Plataforma Hypatiamat.

No capítulo a seguir de desenvolvimento da interface, será especificada toda a implementação do servidor de interfaces, que por sua vez irá consumir os dados fornecidos pela API.

5

DESENVOLVIMENTO DA INTERFACE

Dada a conclusão do capítulo de desenvolvimento da API, temos o suporte necessário para dar início à abordagem de desenvolvimento das interfaces. O servidor de interfaces, como já esteve patente no capítulo das tecnologias, será desenvolvido em *Vue.js*.

Através das diferentes vistas da aplicação serão aplicados os conteúdos e funcionalidades previstos nos requisitos estudados. Nomeadamente no que diz respeito aos dois tipos de utilizador da aplicação, professores e alunos, a interface terá diferentes páginas de acordo com a experiência de utilização pretendida para cada um.

5.1 INICIALIZAÇÃO

No início do desenvolvimento da interface, foram definidas algumas configurações. As funcionalidades escolhidas na instalação do projeto foram o *Router* e o *VueX*, para além do *Vuetify* que foi adicionado posteriormente.

O *Vue* permite criar uma SPA, ou seja é uma aplicação de página única. Para dar a possibilidade de existirem várias páginas na SPA, existe o *Router*, que permite definir rotas consoante as diferentes vistas pretendidas para a aplicação. Em termos de código a interface é dividida em *views* e componentes, sendo que cada *view* corresponde a uma determinada rota e os componentes são partes da interface que podem ser integradas nas várias vistas.

As páginas e respetivas rotas, estruturadas para esta aplicação foram as seguintes:

- *Dashboard* - Vista inicial do utilizador e de TPC ativos;
- *Historic* - Vista de TPC expirados;
- *Stats* - Vista de estatísticas;
- *CreateTPC* - Vista de criação de TPC;
- *DoTPC* - Vista de resolução do TPC;
- *SeeTPC* - Vista de informações do TPC;
- *Results* - Vista de resultados do TPC.

A maior parte das rotas são comuns a professores e alunos, contudo existem rotas que são exclusivas a cada um deles. Para impedir o acesso de um utilizador a uma determinada vista, é necessário proteger as rotas da interface.

Como tal foi definido que um utilizador tem que estar autenticado (presença do *token*) para aceder às várias páginas da aplicação, restringindo assim as rotas necessárias. As vistas que não são partilhadas entre os dois tipos de utilizador, são o *DoTPC* (exclusiva ao aluno), o *CreateTPC* e *Results* (exclusivas ao professor), sendo que se tentarem aceder a uma rota proibida, o acesso é impedido e são redirecionados para a vista inicial.

Outra preocupação a ter em conta na implementação, é o facto da aplicação ter que funcionar em tempo real, respeitando os tempos de expiração dos TPC. A interface precisa de ser responsiva no sentido que se um TPC expirar enquanto o utilizador se encontra na aplicação, essa mudança tem que ser refletida nas diferentes páginas e no acesso a estas. De modo a lidar com este fator foi utilizado o *VueX*, introduzido anteriormente.

O *VueX* permite esta gestão de estado em tempo real através da sua *store*, sendo possível fazer o controlo de que TPC estão ativos ou expirados, para um determinado utilizador. No caso do aluno, existe uma *flag* (*isEditing*) que indica se o aluno está no modo de resolução, de maneira a impedir que navegue para fora da página. Esta *store* também trata da gestão dos dados de autenticação, nomeadamente o armazenamento do JWT. Através da tabela a seguir podemos ver as variáveis utilizadas e o módulo a que correspondem.

Módulo	**Variáveis de Estado**
Autenticação	*token* *userId* *userType*
TPC	*lista de TPC ativos* *lista de TPC expirados* *isEditing*

Tabela 2: *Store* da aplicação de TPC (VueX)

Em modo de conclusão, com o objetivo de tornar a experiência de utilização mais agradável e completa, foi utilizado o *plugin* *Vuetify* que oferece todo o tipo de modelos e componentes de interface funcionais.

5.2 TIPOS DE TAREFAS

Em antecedência à mostragem das vistas concebidas para a interface da aplicação, nesta secção será feito um destaque aos tipos de tarefas disponíveis. Os exercícios existentes na coleção da Plataforma Hypatiamat, têm diferentes tipologias, sendo necessário diferenciar na interface do programa.

Neste momento existem cinco tipos diferentes de questões: tarefas de escolha múltipla; escolha múltipla com imagens; resposta aberta; ângulos; simetrias. Os modelos desenvolvidos para cada uma destas variantes de tarefas, são exibidos a seguir. Também é de realçar que as questões podem ter auxílio de uma imagem ou não, focando-se apenas no enunciado.

Escolha Múltipla (Normal)

Começando pelo tipo mais básico, temos as questões de escolha múltipla, que consistem na apresentação da questão e de várias alíneas como possível resposta. Apenas uma das opções disponíveis é que pode ser a correta.

As escolhas múltiplas podem ter um variado número de alíneas, desde apenas duas até a um máximo de seis. Através da figura podemos ver um exemplo deste tipo.

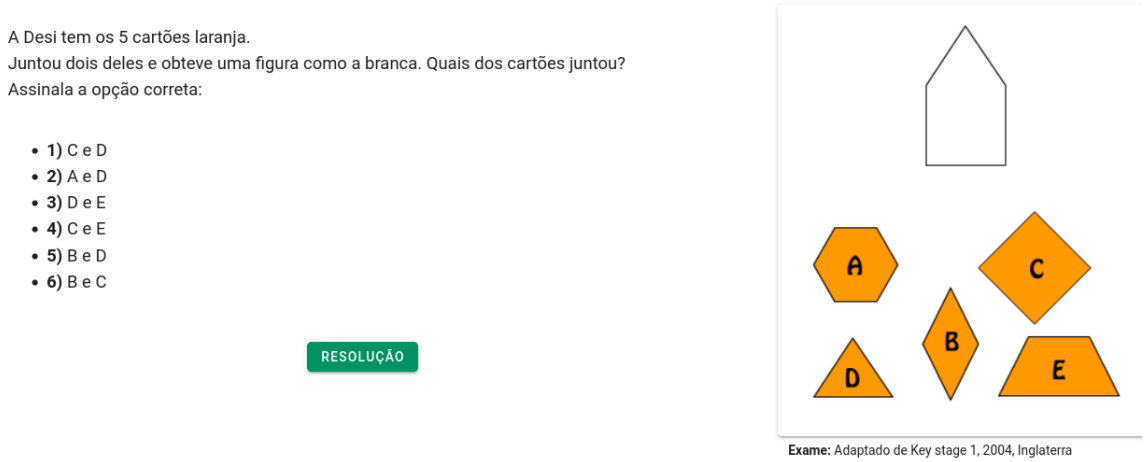


Figura 32: Tarefa de escolha múltipla (normal)

Escolha Múltipla (Imagens)

Outro tipo de tarefa, semelhante à anterior, são as questões de escolha múltipla com recurso a imagens. Mais uma vez, apenas uma das opções será a correta.

Podemos ver na figura uma tarefa deste tipo, sendo que neste caso o número possível de opções é sempre quatro.

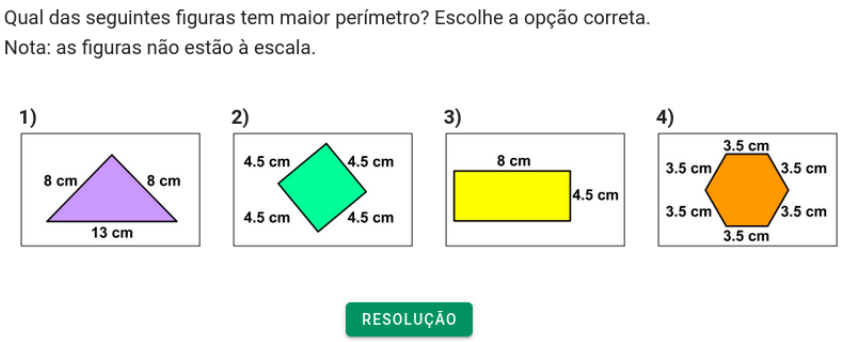


Figura 33: Tarefa de escolha múltipla (imagens)

Resposta Aberta

O próximo tipo de tarefa é o de resposta aberta. Nestas tarefas, o aluno tem liberdade de resposta sendo que utiliza um teclado para poder escrever o que pretende. Este teclado será exibido na próxima secção do capítulo.

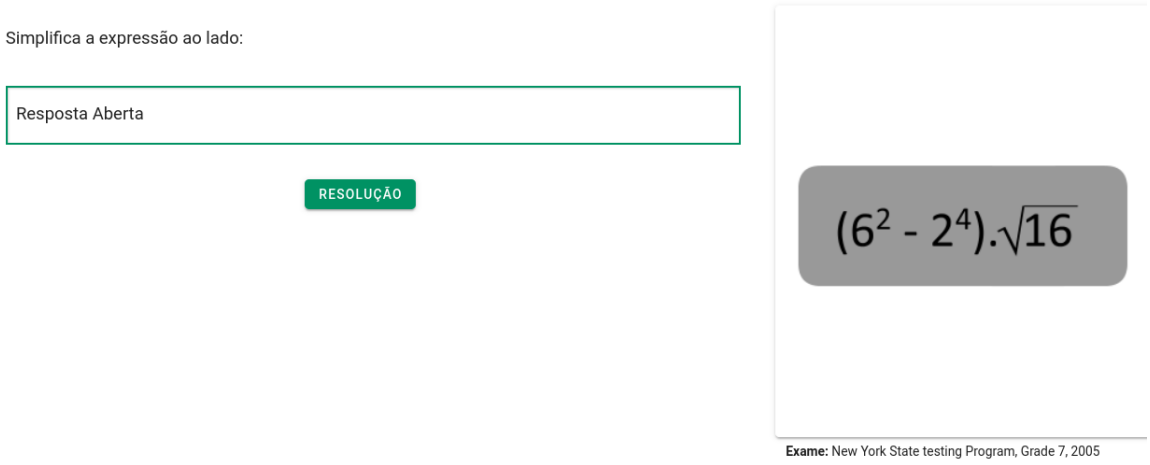


Figura 34: Tarefa de resposta aberta

Ângulos (Resposta Aberta)

Em continuidade do tipo de resposta aberta exibido em cima, existe outra especificação que são os ângulos. Nestas tarefas de medição de ângulos, o aluno terá auxílio de um transferidor no momento da resolução, assim como o teclado para escrever a sua resposta. Novamente, esta visão por parte do aluno será mostrada na secção posterior.

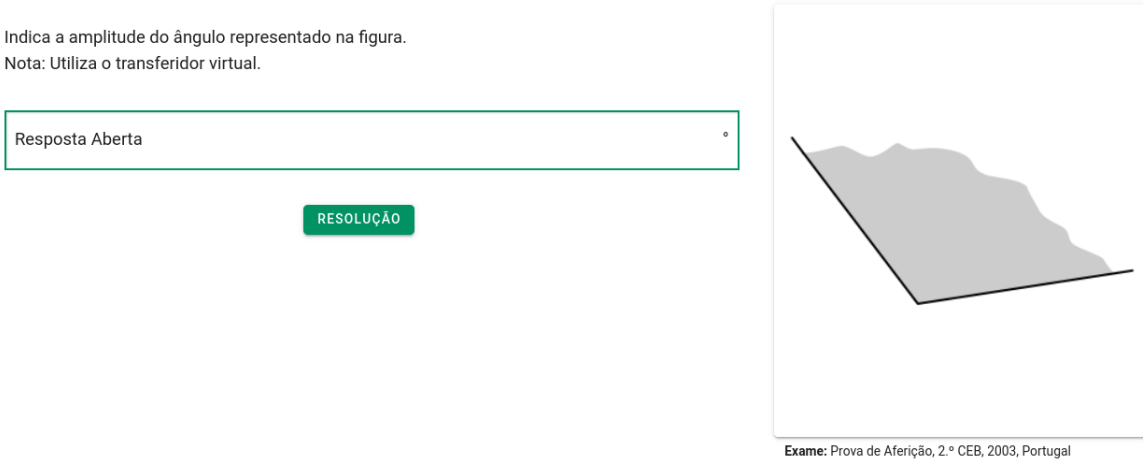


Figura 35: Tarefa de medição de ângulos

Simetrias (Grelha)

Finalmente podemos analisar a última tipologia de tarefa, sendo que este caso é completamente diferente de todos os tipos revistos anteriormente. Esta variância tem como tema as simetrias, em que são apresentadas duas grelhas. Uma das grelhas tem uma imagem definida, cabendo ao aluno completar a outra grelha construindo a imagem simétrica da original.

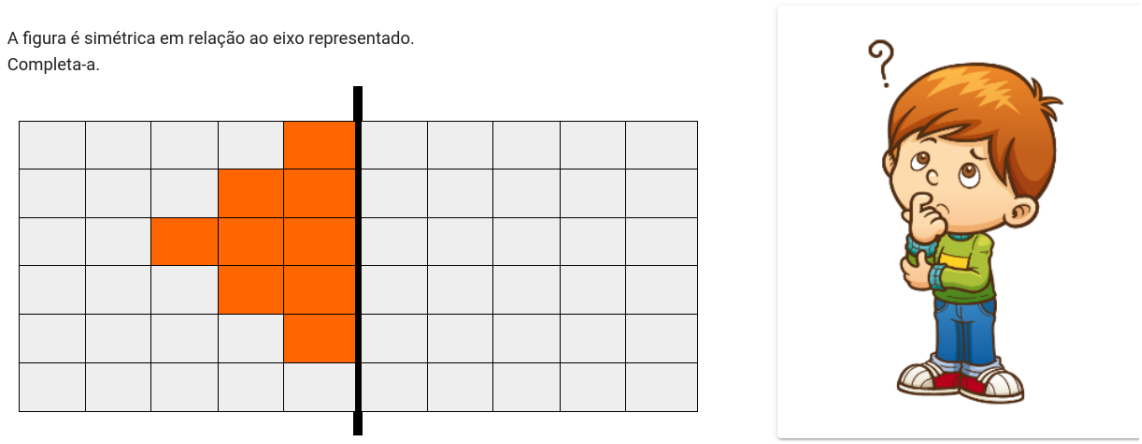


Figura 36: Tarefa de simetrias

5.3 Interface por utilizador

Esta secção servirá para demonstrar as vistas e componentes criados aquando do desenvolvimento da interface da aplicação. A apresentação e respetiva descrição, será repartida pelos dois tipos de utilizadores, *Professor* e *Aluno*.

A única vista comum aos dois utilizadores é a de autenticação, onde estes inserem as suas credenciais de acesso, sendo esta apresentada na figura abaixo. Toda a interface implementada

é responsiva, estando adaptada para ser acedida no *browser* do computador ou através do telemóvel/*tablet*.

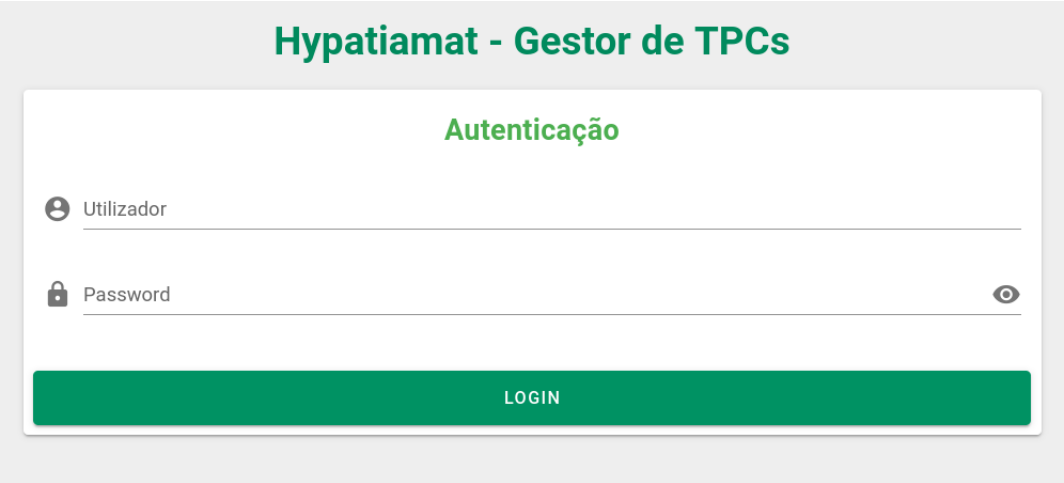


Figura 37: Interface - Login

5.3.1 *Interface do Professor*

Barra de Navegação

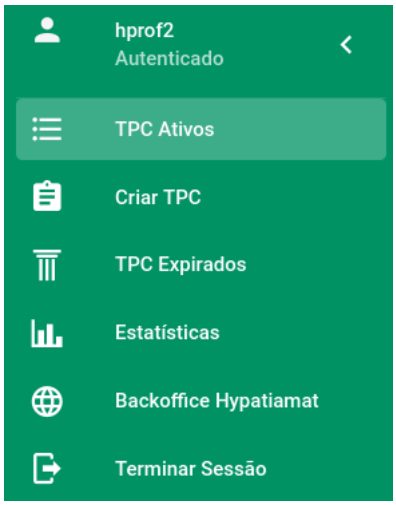


Figura 38: Interface - Barra de Navegação do Professor

A barra de navegação do professor, apresenta as diferentes páginas que este pode aceder. Estas páginas, que foram descritas anteriormente, permitem a um professor aceder à listagem de TPC ativos, à página de criação de TPC, à listagem de TPC expirados e às estatísticas. Por fim, o utilizador também tem possibilidade de entrar no *Backoffice* do *Hypatiamat* (componente à parte), assim como proceder à terminação da sua sessão.

Criar TPC

A página de criação de TPC, uma das principais funcionalidades da aplicação, pode ser visualizada a seguir. Esta página será repartida em várias imagens, para uma melhor compreensão dos seus elementos.

Figura 39: Interface - Criar TPC (Informações)

Na primeira figura, podemos visualizar a parte inicial da página. Contém os vários campos de informação a preencher no TPC, nomeadamente o título, n.º de tentativas, data limite, hora limite, turmas e por fim as configurações extra.

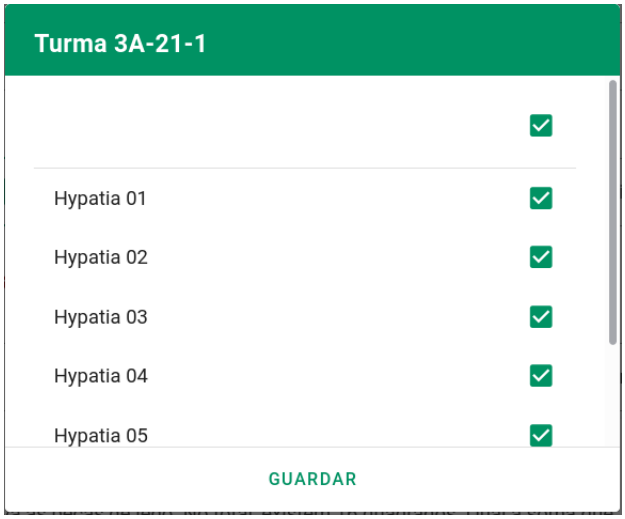


Figura 40: Interface - Criar TPC (Turmas)

Através da figura anterior, podemos ver o componente de turma, em que para uma dada turma o professor pode escolher os alunos a quem quer submeter o TPC.

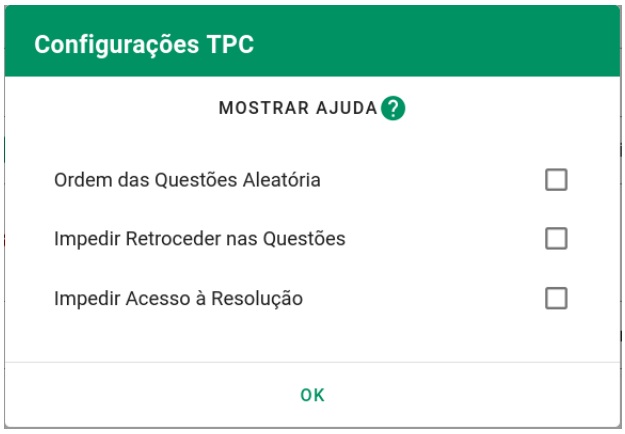


Figura 41: Interface - Criar TPC (Configurações)

Continuando no âmbito das informações, é apresentado o componente de configurações, em que o professor escolhe que tipo de opções extra pretende para o TPC.

A segunda parte da página de criar TPC é o *dashboard* de seleção de tarefas, que será exposto abaixo.

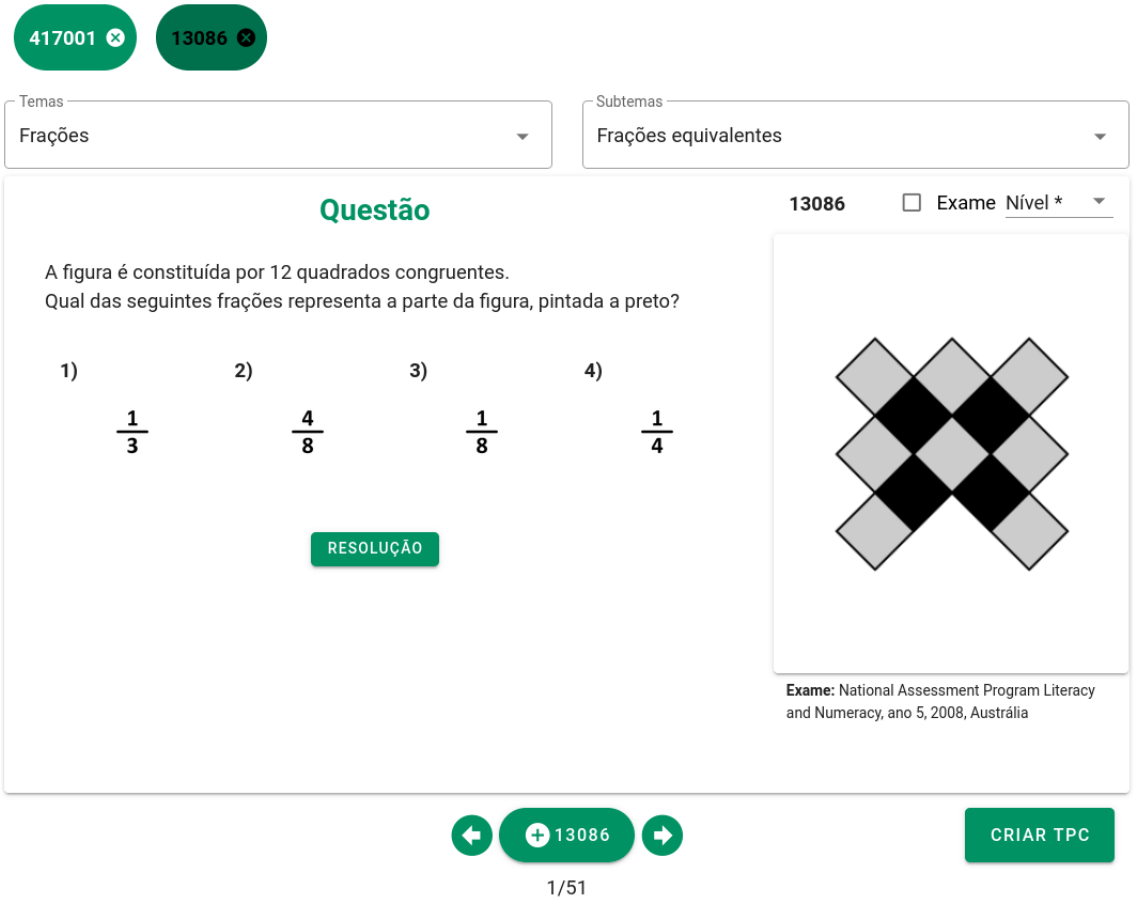


Figura 42: Interface - Criar TPC (*Dashboard*)

O *dashboard* de seleção de questões, tem vários elementos que serão descritos por ordem.

O elemento do topo trata-se da lista de tarefas já adicionadas pelo professor no *dashboard*. Cada tarefa é representada pelo seu código, sendo possível removê-la caso seja necessário.

No que diz respeito ao processo de adicionar tarefas à lista do TPC, inicialmente temos as listagens de temas e subtemas. O professor escolhe o conjunto para o tipo de questões que pretende. Estas questões são apresentadas no componente abaixo. Como já foi possível verificar na secção de tipos de tarefas, este componente contém o enunciado da questão, as respostas, uma imagem auxiliar (caso exista) e acima desta os campos de exame e nível que podem ser utilizados para filtrar as questões. Também é possível visualizar a proposta de resolução da respetiva tarefa.

A navegação entre as várias tarefas, é feita a partir das setas na base, sendo que se o professor quiser adicionar uma dada questão, utiliza o botão do meio (com o código da questão) para acrescentá-la à lista do TPC.

Por fim, após o preenchimento de todos os campos e seleção das questões pretendidas, o professor utiliza o botão do canto inferior direito para proceder à criação do TPC.

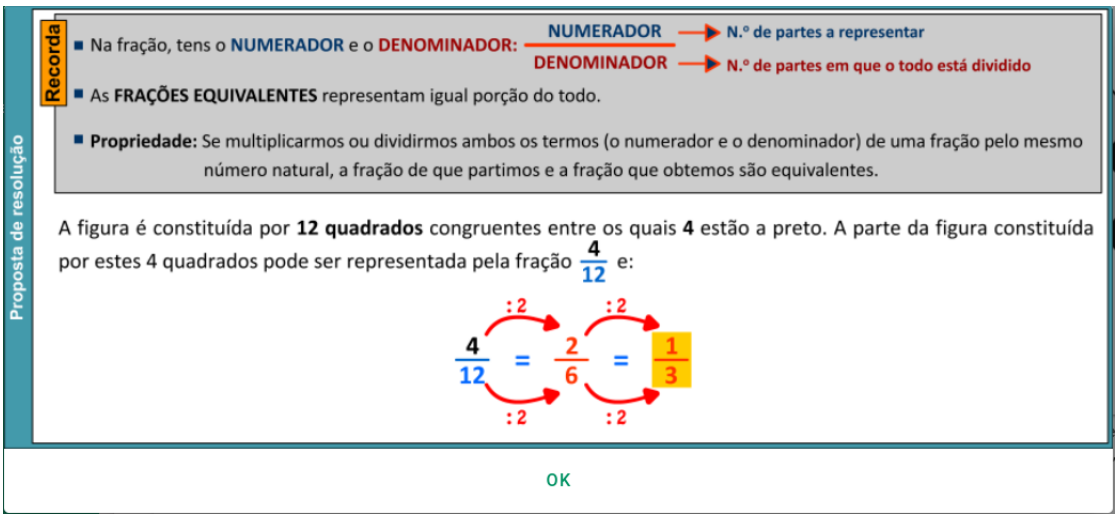


Figura 43: Interface - Criar TPC (Proposta de Resolução)

Na imagem superior, é demonstrado um exemplo de uma proposta de resolução, à qual o professor tem acesso para as várias tarefas.

TPC Ativos

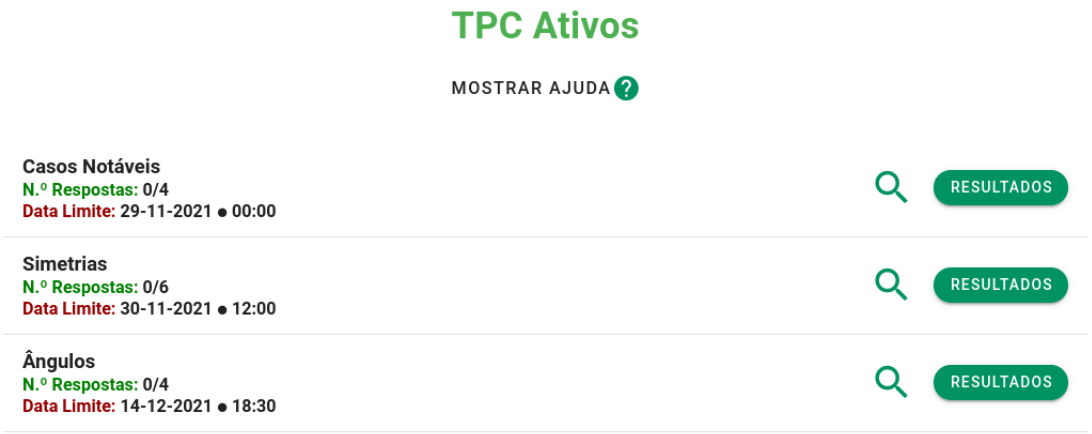


Figura 44: Interface - TPC Ativos do Professor

A página de TPC ativos, pela visão do professor, tem o estilo demonstrado na figura anterior. Para cada TPC criado, temos três informações: o título do TPC, o n.º de alunos que já deram resposta e a data limite. Além destas informações, o professor pode ver as configurações do TPC assim como os resultados dos alunos.

Através do botão da lupa, o professor pode ver as informações do TPC que criou. Esta página pode ser visualizada nas duas figuras seguintes.

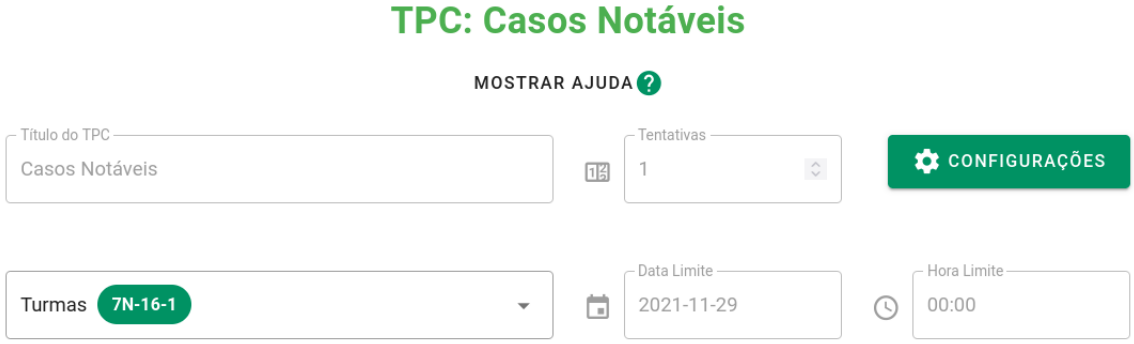


Figura 45: Interface - Ver TPC (Informações)

Nesta imagem, podemos ver as informações básicas do TPC, assim como as configurações e turmas/alunos escolhidos.

Figura 46: Interface - Ver TPC (*Dashboard*)

A imagem anterior demonstra o *dashboard* das questões do TPC. Ao navegar em cada questão, através dos botões no topo ou das setas em baixo, podemos analisar todas as suas respetivas características. Também temos a presença do botão de apagar a vermelho, que elimina o TPC e suas dependências.

Destacando o botão de resultados da primeira imagem novamente, onde o professor tem acesso aos resultados dos alunos, são agora exibidas as interfaces correspondentes.

Figura 47: Interface - Resultados TPC

Visualizando a figura anterior, temos a página de resultados de um determinado TPC. Esta página contém uma tabela com os resultados dos vários alunos e respetivas informações como número, nome, turma, n.º da tentativa, questões corretas e classificação em percentagem. A última coluna das respostas é onde o professor pode visualizar as respostas de cada aluno, através do botão da lupa.

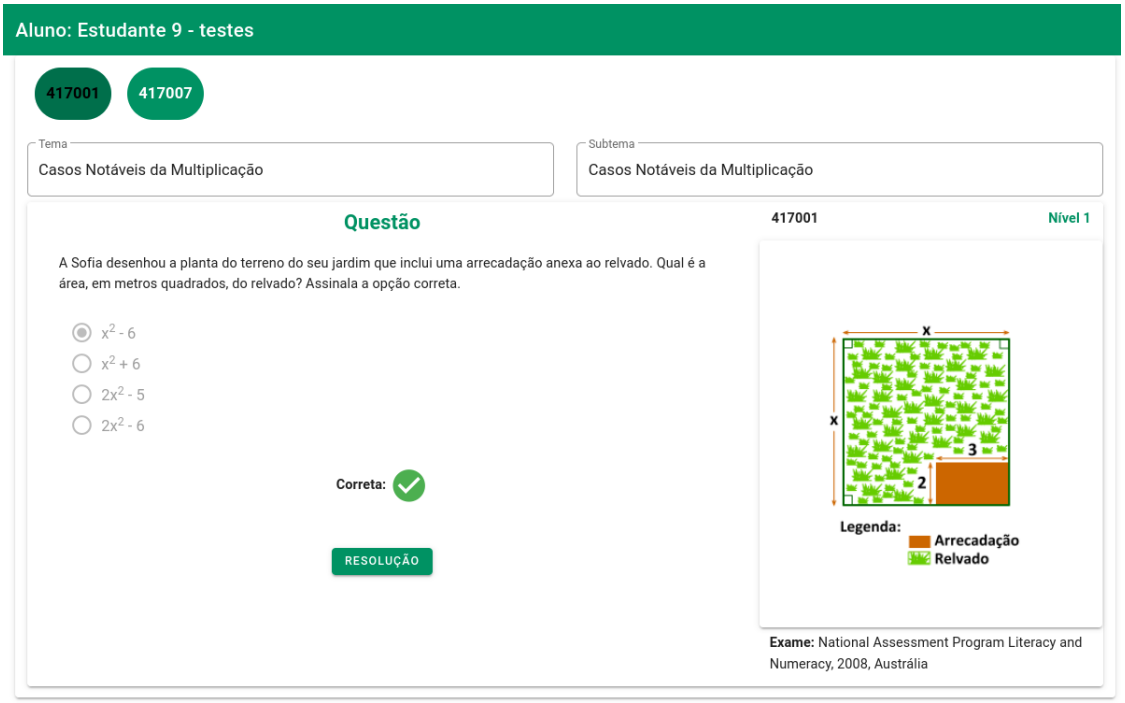


Figura 48: Interface - Respostas do Aluno

Neste último componente das respostas, que segue o mesmo modelo de *dashboard* revisto antes, o professor tem acesso às respostas do aluno a cada tarefa tendo indicação se a resposta dada está correta ou não.

TPC Expirados

Figura 49: Interface - TPC Expirados do Professor

A página de TPC expirados, tem o mesmo aspeto de interface que a página de TPC ativos tem. As funcionalidades através dos botões apresentados também são as mesmas, sendo que a única diferença está no facto dos TPC apresentados estarem expirados, ou seja, os alunos já não podem submeter qualquer resolução.

O professor continua a ter acesso às informações de cada TPC expirado, assim como aos resultados dos alunos que o completaram.

Estatísticas

Figura 50: Interface - Estatísticas do Professor

A última vista exibida, é relativa à página de estatísticas que o professor tem acesso na aplicação. Esta página apresenta dados em tabela do desempenho de alunos, aquando da resolução dos TPC.

Como tal, o professor escolhe o ano letivo, a turma que pretende analisar e no final escolhe se pretende avaliar os dados de todos os TPC realizados ou simplesmente de um específico. Em semelhança com a página de resultados, são disponibilizadas informações como o número e nome do aluno, n.º de TPC realizados, n.º de questões corretas e classificação total (em percentagem).

5.3.2 *Interface do Aluno*

Barra de Navegação

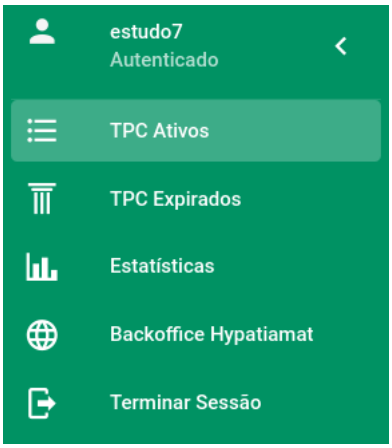


Figura 51: Interface - Barra de Navegação do Aluno

A barra de navegação do aluno exibe as diferentes páginas à qual este tem acesso. As páginas da interface do aluno são semelhantes às do professor, variando quanto às funcionalidades disponíveis que são sempre diferentes entre os dois. O aluno autenticado pode visitar as páginas de listagem de TPC ativos e expirados, assim como a página de estatísticas.

Tal como anteriormente, o aluno consegue aceder ao *Backoffice* do *Hypatiamat* e também proceder à terminação da sua sessão.

TPC Ativos

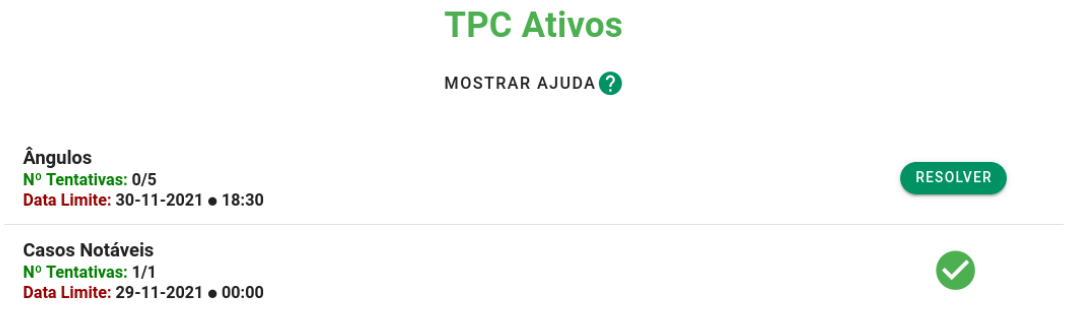


Figura 52: Interface - TPC Ativos do Aluno

A página de TPC ativos para o aluno, é apresentada na figura anterior. Para cada TPC que se encontre ativo, existem informações como o título, n.º de tentativas disponíveis e data limite.

No caso do aluno ter possibilidade de realizar o TPC, se ainda tiver tentativas disponíveis, surgirá o botão de resolver para proceder à respetiva resolução. Em caso contrário, se já não tiver tentativas restantes, o TPC aparecerá com um ícone de certo e o aluno não poderá exercer mais tentativas de resolução.

Na secção a seguir, será destacado o processo de resolução de um TPC assim como as ferramentas disponibilizadas ao aluno para resolver cada tipo de tarefa já mencionado.

Resolver TPC

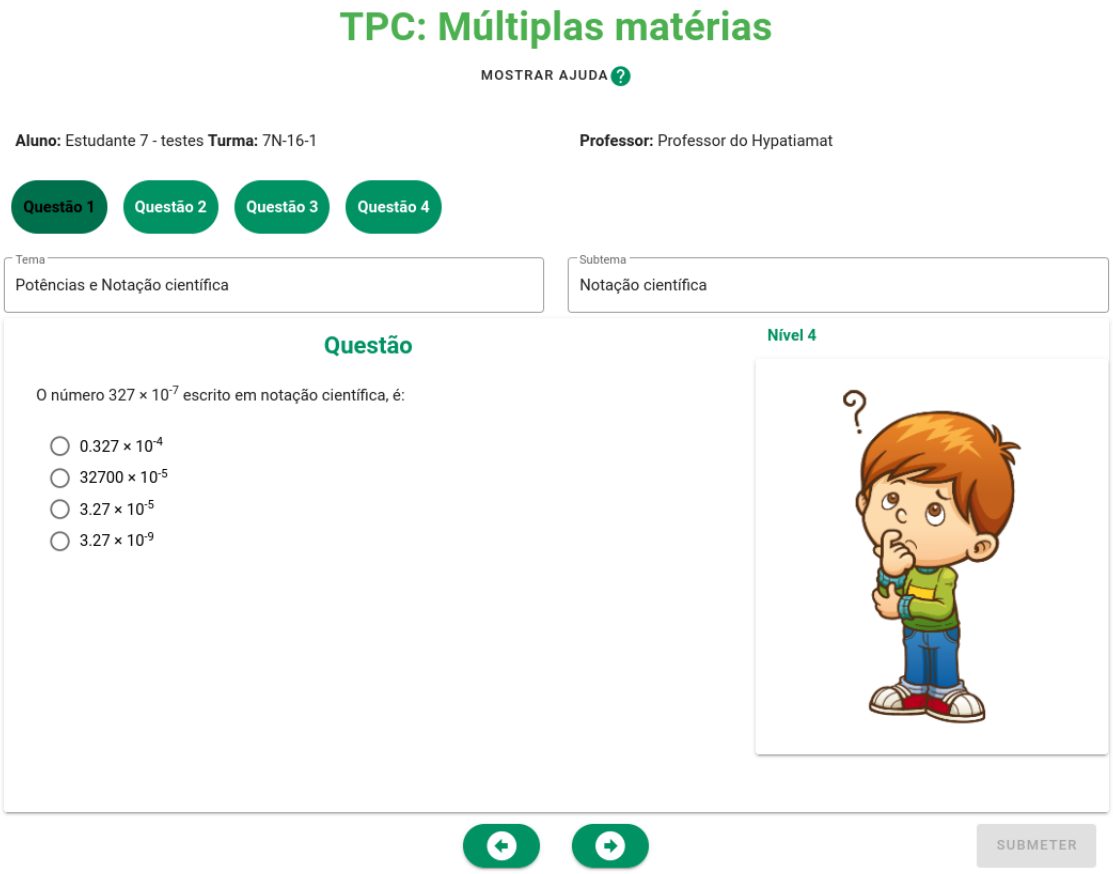


Figura 53: Interface - Resolver TPC

A visão do aluno aquando da resolução de um TPC, é exibida através da figura anterior. Nesta página existem algumas informações como nome do aluno, turma e respetivo professor. O aluno tem o *dashboard* onde navega pelas tarefas do TPC, sendo que terá que dar resposta a cada uma delas.

Como anteriormente já foram expostos os diferentes tipos de tarefas da aplicação, agora será demonstrada a visão sobre cada tipo por parte do aluno. Começando pelas questões de escolha múltipla, o aluno só tem que selecionar a opção que pretende, como está exemplificado em baixo. Este comportamento é igual sendo escolha múltipla com imagens ou não.

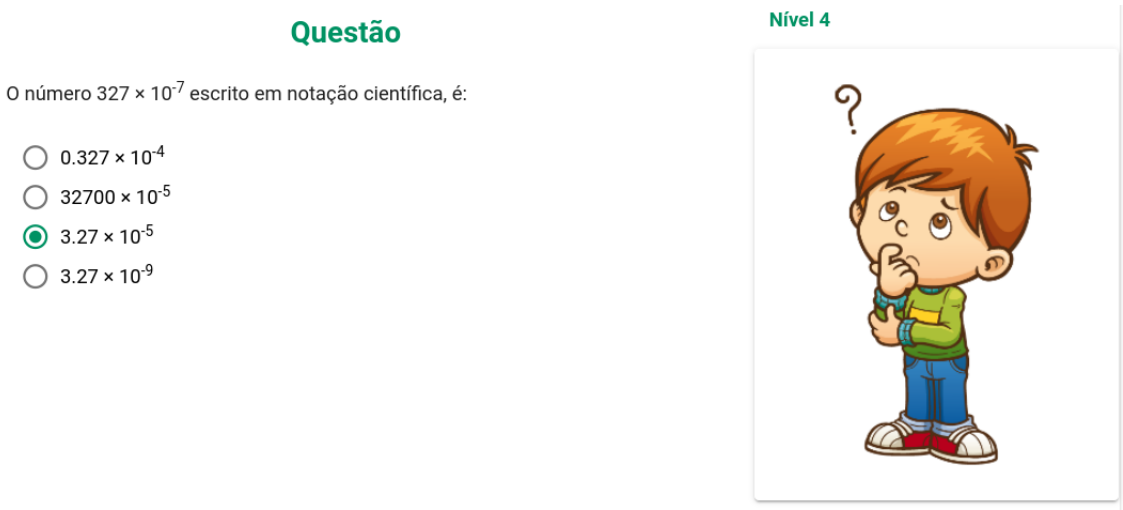


Figura 54: Interface - Resolver TPC (escolha múltipla)

O tipo de questões seguinte é o de resposta aberta. O aluno para estes exercícios, tem disponibilizado um teclado preparado para vários tipos de resposta. Podemos analisar a presença deste teclado na imagem seguinte.

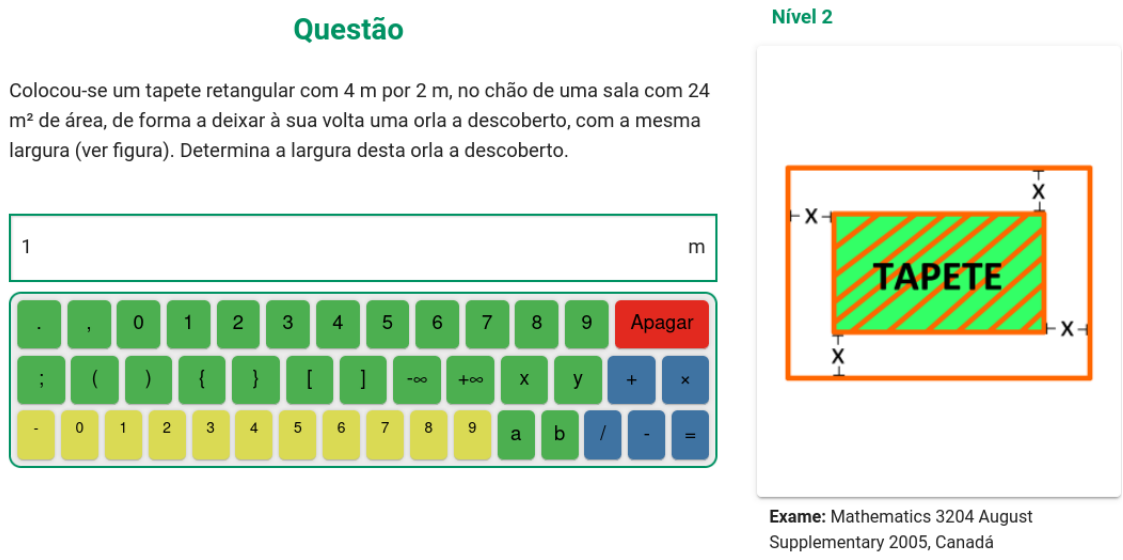


Figura 55: Interface - Resolver TPC (resposta aberta)

Em paralelo com o tipo anterior, temos as tarefas de medição de ângulos que são também de resposta aberta mas com a particularidade do aluno ter acesso a um transferidor. Este transferidor é disponibilizado através de um botão, sendo que o aluno tem que utilizá-lo para medir o ângulo e dar resposta ao enunciado. Este comportamento está explícito na figura subjacente.

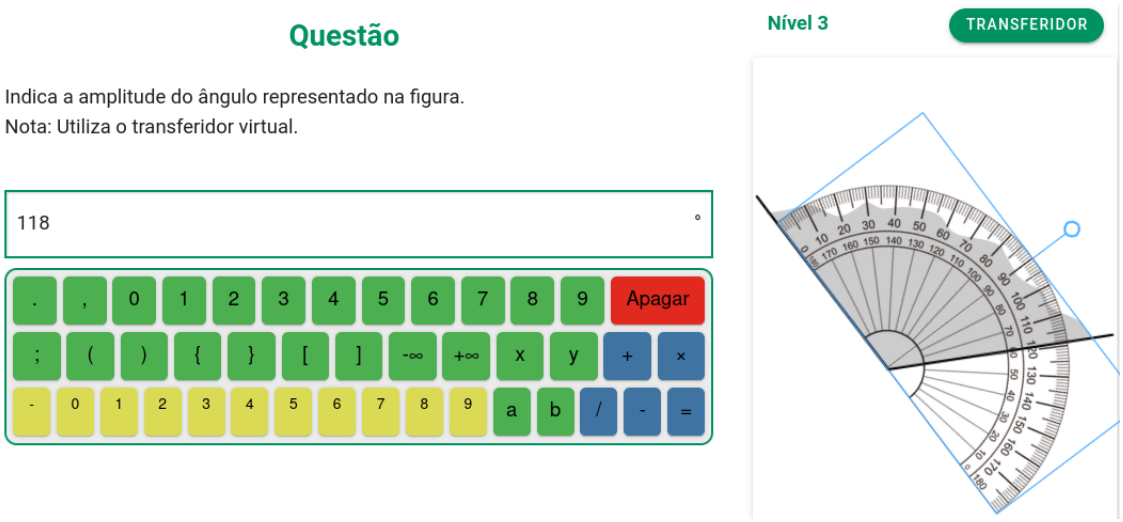


Figura 56: Interface - Resolver TPC (medição de ângulos)

O último tipo de tarefa que falta analisar é o de simetrias. Este é bastante simples no sentido que o aluno apenas tem que clicar sobre os quadrados da grelha, com o intuito de replicar a simetria com a imagem da grelha auxiliar. Através da imagem a seguir é possível visualizar este processo, sendo que a grelha de construção da simetria neste exemplo é a da esquerda.

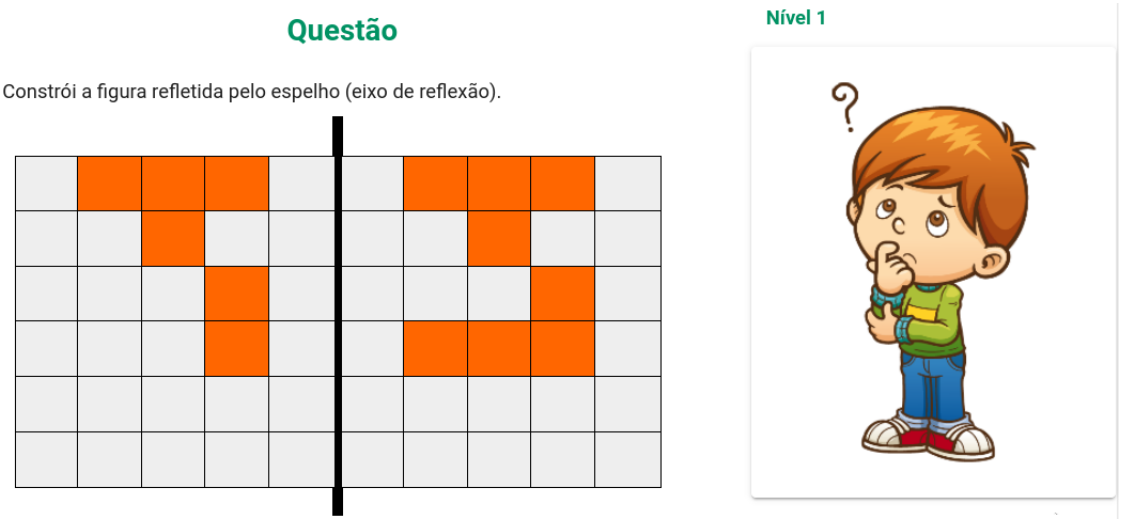


Figura 57: Interface - Resolver TPC (simetrias)

Finalizada a análise dos diferentes tipos de tarefas, consegue-se ter uma melhor ideia de como foram implementados estes exercícios assim como as ferramentas necessárias para que o aluno consiga fornecer diferentes tipos de resposta.

Quando o aluno der resposta a todas as questões do TPC, o botão de submeter fica disponível, podendo enviar a sua resolução para o sistema.

TPC Expirados

Figura 58: Interface - TPC Expirados do Aluno

A listagem de TPC expirados do aluno, tem muita semelhança com as funcionalidades das listagens do professor, revistas anteriormente.

Cada TPC apresenta informações como o título, classificação obtida e em que data expirou. Também em cada entrada, temos dois botões que levam a diferentes interfaces: o da lupa, que serve para o aluno rever o formato do TPC e outro botão de resolução onde o aluno pode ver as suas respostas às tarefas do TPC.

De notar no exemplo da figura sobrejacente, existem entradas em que os botões estão desativados ou diferentes, isto deve-se ao comportamento da aplicação. A primeira entrada é o caso normal em que o aluno deu resposta ao TPC. A segunda entrada, tem o botão de resolução desativado, isto significa que o professor optou por impedir os alunos de verificarem as soluções ou a correção das suas respostas. No terceiro e último caso, é visível o ícone de errado em vez do botão de resolução, isto deve-se ao facto do aluno não ter realizado nenhuma tentativa de resolução do TPC, sendo portanto impossível aceder às suas respostas.

Estatísticas

Figura 59: Interface - Estatísticas do Aluno

Para finalizar, temos a vista relativa à página de estatísticas do aluno. Esta página permite ao aluno ver os dados do seu desempenho nas resoluções dos TPC.

Como tal, o utilizador pode escolher na lista um TPC específico que realizou, ou selecionar a opção *Todos*. A tabela mostrará informação como o professor, o n.º da tentativa da resolução, o n.º de TPC realizados (no caso da opção *Todos*), assim como a quantidade de respostas corretas e respetiva classificação (em percentagem).

O capítulo que agora termina, abordou o processo de implementação do servidor de interfaces. Primeiramente foi inicializado o desenvolvimento com a configuração das funcionalidades necessárias, assim como a definição das vistas de utilização. Também foi explicado o processo de controlo de estado no servidor, através do *Vuex*.

De seguida, foi realizada a construção das interfaces tendo em conta as funcionalidades e requisitos idealizados anteriormente. Da mesma forma, foi feito um foco nos diferentes tipos de tarefas e respetivas variâncias na interface.

Finalmente, foram apresentadas todas as diferentes vistas e componentes de utilização, divididas por cada tipo de utilizador do sistema.

No capítulo a seguir, será analisado o processo de *deployment* da aplicação e respetiva disponibilização aos seus utilizadores.

6

DEPLOYMENT DA APLICAÇÃO

Neste capítulo de *deployment*, será abordado o processo de integração da aplicação de TPC na Plataforma Hypatiamat e consequente passagem para ambiente de produção.

Primeiramente foi utilizado o *Github* como ferramenta de controlo de versões, tendo sido criados dois repositórios, um para a API de dados e outro para a interface. Estes repositórios são úteis para atualizar a aplicação sempre que existirem novas versões do sistema.

A disponibilização em ambiente de produção, tem o intuito de fornecer aos utilizadores finais a possibilidade de testarem a aplicação e desta forma, possibilitar receber opiniões relevantes sobre o seu funcionamento e utilização. Como tal, foi necessário orientar com os administradores do servidor e domínio *web* do Hypatiamat todo este processo.

A Plataforma Hypatiamat utiliza o *Nginx* como servidor de *reverse proxy*, logo foi solicitada a criação de dois domínios para os dois serviços desenvolvidos de *back-end* e *front-end*: `apitpc.hypatiamat.com` e `tpc.hypatiamat.com`.

Após finalizar todo este processo de transição para ambiente de produção, os dois serviços ficaram funcionais e disponíveis *online*, sendo possível aos utilizadores a experimentação do novo componente.

Com o decorrer do desenvolvimento, através dos testes realizados e em conjunto com a importante opinião dos utilizadores, foram alteradas algumas funcionalidades assim como adicionadas novas funções à aplicação de TPC.

6.1 PRODUÇÃO DA API

O servidor da API de dados em ambiente de produção, necessita de uma forma de gestão do seu funcionamento. Ao contrário do ambiente local, no que diz respeito a iniciar, parar e reiniciar o servidor, são ações realizadas de modo diferente em produção, sendo que neste caso foi utilizado o módulo *PM2*.

O *PM2* é um gestor de processos que auxilia na gestão e manutenção de uma aplicação *online*[PM221]. Oferece uma utilização acessível, através da sua CLI simples e intuitiva.

O facto de não ser possível gerir o processo do servidor de maneira direta como em ambiente local, faz com que o *PM2* tenha um papel fundamental neste aspeto. Com o *PM2* é possível visualizar dados de funcionamento e de erro do servidor e utilizar os diferentes comandos

disponíveis como *start, restart, reload, stop, list*. Outra importante característica do *PM2*, é que em caso de erro na API, este reinicia automaticamente o processo o que permite manter o servidor sempre em funcionamento.

6.2 Produção da interface

A produção do servidor de interface é mais simples e direta que a da API. É apenas necessário servir a interface desenvolvida de uma forma estática na *web*.

No caso de termos um servidor de *back-end* separado da interface, o *front-end* será uma aplicação puramente estática[Vue21a]. Como tal, de forma a gerar os ficheiros necessários para produção, é executado o comando de *build* do *npm*. Este comando irá criar os ficheiros para uma diretoria *dist*, cujo conteúdo estará no repositório de *Github* da interface. O servidor irá portanto servir estes ficheiros estáticos, disponibilizando assim a interface *online*.

6.3 Autenticação entre componentes

Com a aplicação já a funcionar em produção, foi pedido que se uniformizasse a autenticação entre o Hypatiamat e os componentes *TPC* e *Backoffice*. O que se pretendia era que, o utilizador ao autenticar-se num dos componentes, passasse a estar autenticado também nos restantes sem precisar de inserir as credenciais novamente.

Começando pela ligação entre a Plataforma Hypatiamat e os componentes *TPC* e *Backoffice*, apenas foi possível definir uma autenticação unidirecional. Isto deve-se ao facto do processo de autenticação do Hypatiamat ser diferente dos outros dois, que como é conhecido utilizam o JWT. Ou seja, um utilizador ao autenticar-se na plataforma principal, estaria também autenticado no *TPC* e *Backoffice*, sendo que o contrário não seria possível.

Quanto à autenticação entre as duas aplicações *TPC* e *Backoffice*, visto recorrerem ao mesmo tipo de autenticação, foi possível estabelecer uma ligação bidirecional entre estes. Para tal, como já foi referido no capítulo de desenvolvimento da API, era necessário que os *token* gerados pelas aplicações tivessem o mesmo segredo, assim como um *payload* semelhante. De forma a estabelecer a ligação entre os dois componentes, foi utilizado o módulo *cross-storage*, que será explicitado na secção seguinte.

6.3.1 *Módulo cross-storage*

O *cross-storage* é um módulo de armazenamento local, com permissões. Permite que múltiplas janelas/separadores de *browser*, entre variados domínios, partilhem um único *localStorage* (armazenamento no browser)[Cs21].

Este módulo divide-se em dois componentes, *Hub* e *Client*. O *Hub* é o componente cujo domínio se pretende aceder e modificar, enquanto que o *Client* será o domínio que irá criar uma conexão ao *Hub*, acedendo assim ao ao seu armazenamento local.

Deste modo, tanto o *TPC* como o *Backoffice* possuem um *Hub* que permite conexões a partir do domínio *hypatiamat.com*, sendo que cada um também é *Client* um do outro, estabelecendo portanto uma ligação bidirecional.

Ao entrar numa das duas aplicações, o módulo verifica se já existe presença de um JWT no *localStorage* da outra aplicação e em caso afirmativo, a autenticação é realizada automaticamente. Caso o utilizador pretenda terminar a sessão numa das aplicações, o *cross-storage* também remove o respetivo *token* da outra aplicação, mantendo assim a coerência deste processo de autenticação.

Todo o processo de produção e disponibilização da aplicação de TPC na Plataforma Hypatiamat, foi explicitado no presente capítulo.

Foram introduzidos os mecanismos necessários à criação dos domínios *web* de acesso aos dois servidores, da API de dados e de interfaces. Os procedimentos de produção da API e da interface também foram referidos, mencionando os detalhes e tecnologias que os suportam.

Por fim, foi descrita a implementação da autenticação entre os vários componentes da Plataforma Hypatiamat, que permite a partilha do mesmo *token* de utilizador entre estes.

O capítulo subsequente corresponde ao último desta dissertação, onde serão prestadas as conclusões finais sobre todo o trabalho realizado neste projeto.

7

CONCLUSÕES E TRABALHO FUTURO

O trabalho descrito através desta dissertação, faz a junção de duas áreas como o desenvolvimento de *software* para a *web* e a educação, mais especificamente na área curricular da matemática.

Inicialmente no capítulo 1, é efetuada uma introdução ao contexto do projeto a desenvolver e são definidos os objetivos e metodologias de trabalho.

No capítulo 2 foi feita uma investigação detalhada, de modo a compreender e produzir o estado da arte sobre o trabalho da Associação Hypatiamat/Projeto Hypatiamat, mais especificamente sobre toda a Plataforma Hypatiamat e os seus recursos aplicacionais. Também foi feito o estudo das tecnologias necessárias ao desenvolvimento do projeto de dissertação.

Após a abordagem de investigação e compreensão dos objetivos do projeto, foi realizado no capítulo 3 todo o planeamento do componente de *software* a produzir. Começando pela definição da estrutura do sistema e clarificação de toda a lógica necessária à aplicação, explicitando todos os requisitos funcionais assim como visualização das funcionalidades através de diagramas. Do mesmo modo foi feita a modelação da nova base de dados que serve de suporte ao componente de TPC.

Os capítulos 4 e 5 descrevem a implementação das funcionalidades requeridas no sistema, respetivamente através do desenvolvimento da API de dados e da construção das interfaces de utilização. Na API foram construídas as rotas necessárias e respetivos mecanismos de proteção, mas também a definição da autenticação dos utilizadores. Na parte do desenvolvimento das interfaces, é explicitada a abordagem executada, recorrendo à apresentação das diferentes vistas e componentes por cada tipo de utilizador.

O capítulo 6 apresenta todo o processo de integração da aplicação desenvolvida na Plataforma Hypatiamat e consequente disponibilização em ambiente de produção aos seus utilizadores.

Neste último capítulo da dissertação, serão expressas as considerações finais de todo o trabalho desenvolvido, assim como alguns comentários quanto aos objetivos cumpridos. Em adição, também serão apontadas algumas sugestões de melhoria para o futuro deste projeto.

7.1 CONSIDERAÇÕES FINAIS

Com o finalizar deste projeto de mestrado, pode-se afirmar que o objetivo proposto foi cumprido: especificar e construir um componente que realizasse a gestão de trabalhos para

casa, dentro do âmbito da Plataforma Hypatiamat. Este processo revelou-se desafiante, mas ao mesmo tempo permitiu obter uma experiência muito positiva ao juntar duas áreas de interesse comum, nomeadamente o desenvolvimento de *software* para a *web* e a aplicação de conteúdos educacionais.

Ainda que os requisitos tenham sido cumpridos, a implementação do componente de TPC apresentou alguns obstáculos, tanto a nível tecnológico como a nível de aplicação dos conteúdos. Tendo em conta que este produto de *software* foi produzido para ser utilizado em contexto real, o rigor necessário foi superior aquando do desenvolvimento das suas funcionalidades. Também foi requerido uma maior quantidade de testes de forma a manter a aplicação livre de erros.

Foi essencial conhecer e estudar o funcionamento da Plataforma Hypatiamat, de forma a que fosse criado o novo componente dentro do contexto do projeto existente. A nível de implementação da API, o facto de utilizar uma tecnologia nova como o *Strapi*, exigiu muita leitura da sua documentação e código, apesar de depois tornar o desenvolvimento mais facilitado.

Em termos da construção das interfaces, embora já possuir experiência anterior com o *Vue*, o desenho e implementação das vistas de acordo com os requisitos fez surgir também algumas dificuldades, particularmente na construção dos vários tipos de exercícios existentes.

No entanto estes obstáculos foram eventualmente ultrapassados, dando origem a uma aplicação que se encontra funcional e disponível para todos aqueles que são os seus utilizadores.

7.2 Trabalho Futuro

Ao longo do desenvolvimento da aplicação foram surgindo algumas necessidades extra, que não puderam ser implementadas, face ao tempo que seria necessário dedicar a estas. É importante realçar que qualquer produto de *software* que seja produzido para utilizadores reais, carece sempre de uma manutenção constante e adição de novas atualizações. Sendo assim, são apresentadas essas necessidades assim como algumas sugestões que seriam interessantes para a melhoria do componente no futuro.

Em primeiro lugar, seria relevante para os professores que existisse a funcionalidade de exportar cada TPC criado em formato de *pdf*. Assim como a melhoria do processo de criação de TPC, com mais configurações e opções. Também por parte do aluno, o processo de resolução dos TPC poderá ser melhor executado e planificado.

Quanto às estatísticas, estas têm muita margem de progressão pois foram o aspeto menos focado durante a dissertação. Nomeadamente a apresentação de uma maior quantidade de dados relevantes ao utilizador, de uma forma interativa através de gráficos, diagramas e tabelas.

Em suma, este componente de TPC servirá como base sólida para novos requerimentos aplicacionais, face aquele que é o objetivo de evolução constante da Plataforma Hypatiamat e dos seus conteúdos e aplicações.

BIBLIOGRAFIA

[Aca21] Khan Academy. Our mission is to provide a free, world-class education for anyone, anywhere. https://www.khanacademy.org/about, 2021. Acedido a 2021-01-29.

[Aut21] Auth0. Introduction to JSON Web Tokens. https://jwt.io/introduction, 2021. Acedido a 2021-01-26.

[Cor21] Oracle Corporation. What is MySQL? https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html, 2021. Acedido a 2021-01-25.

[Cs21] Cross-storage. cross-storage - npm. https://www.npmjs.com/package/cross-storage, 2021. Acedido a 2021-06-24.

[eTMC] José Verdasca, Ana Maria Neves, Helena Fonseca, José Alberto Fateixa, Marta Procópio e Teodolinda Magro-C. Melhorar aprendizagens em Matemática pelo uso intencional de recursos digitais - O Hypatiamat como intervenção preventiva na CIM do Ave. https://pnpse.min-educ.pt/estudo4. Acedido a 2021-01-19.

[Fou21] OpenJS Foundation. About Node.js. https://nodejs.org/en/about/, 2021. Acedido a 2021-01-24.

[Gon13] Letícia Catarina Teixeira Gonçalves. GeoGebra: um instrumento auxiliar no processo ensino/aprendizagem da matemática. https://digituma.uma.pt/handle/10400.13/398, 2013. Acedido a 2021-01-31.

[Hor20] Ana Rita Mota Hortênsio. A Influência da Plataforma Hypatiamat na Resolução de Situações Problemáticas Envolvendo a Adição e Subtração. https://comum.rcaap.pt/bitstream/10400.26/33215/1/ANA_HORTENSIO.pdf, 7 2020. Acedido a 2021-01-18.

[Hyp20] Hypatiamat. Hypatiamat: Um desafio para professores, alunos e encarregados de educação. https://www.hypatiamat.com/apresentacao.php, 2020. Acedido a 2020-10-04.

[Mor19] Sara Augusta Fernandes da Silva Moreira. A ClassDojo como recurso educativo no ensino e aprendizagem da Matemática: uma experiência no 2.º Ciclo do Ensino Básico. https://repositorio.utad.pt/handle/10348/9275, 2019. Acedido a 2021-01-31.

[PM221] PM2. PM2 - Quick Start. https://pm2.keymetrics.io/docs/usage/quick-start/, 2021. Acedido a 2021-06-20.

[Soc20] Portugal Inovação Social. Khan academy – norte. https://inovacaosocial.portugal2020.pt/project/aprender-e-ensinar-matematica-com-a-khan-academy, 2020. Acedido a 2021-01-30.

[Str21] Strapi. What is Strapi? https://strapi.io/documentation/developer-docs/latest/getting-started/introduction.html#what-is-strapi, 2021. Acedido a 2021-01-24.

[Vir21] Escola Virtual. Escola Virtual. https://www.escolavirtual.pt, 2021. Acedido a 2021-01-30.

[Vue21a] Vue.js. Deployment. https://cli.vuejs.org/guide/deployment.html#general-guidelines, 2021. Acedido a 2021-06-23.

[Vue21b] Vue.js. What is Vue.js? https://vuejs.org/v2/guide, 2021. Acedido a 2021-01-26.

[Vue21c] Vue.js. What is VueX? https://vuex.vuejs.org/#what-is-a-state-management-pattern, 2021. Acedido a 2021-01-26.

[Vue21d] Vuetify. What is Vuetify? https://vuetifyjs.com/en/introduction/why-vuetify/#getting-started, 2021. Acedido a 2021-01-26.