

DESENVOLVIMENTO DE SISTEMA DE GESTÃO DE AULAS PARTICULARES DE INGLÊS

Eduarda Cardoso dos Santos¹

Rômulo Ferreira Douro²

RESUMO

Este projeto tem por objetivo informatizar o processo de gestão de aulas, alunos e turmas de um professor particular de inglês, trazendo informações como frequência e atrasos do aluno, se ele fez ou não as atividades passadas em aula e para casa, podendo gerenciar também os agendamentos das aulas, e os materiais que serão utilizados. Projeto desenvolvido com as tecnologias VueJS, NestJS e MongoDB, será implantado como uma plataforma web responsiva podendo usar em smartphones, tablets e computadores.

ABSTRACT

This project aims to computerize the process of managing the classes and students of a private English teacher, bringing information such as students' attendance and delays, and whether or not they did the previous class activities and homework, while also being able to manage the class schedules, and the materials that will be used. The project was developed using the VueJS, NestJS and MongoDB technologies, and will be implemented as a responsive web platform that can be used on smartphones, tablets and computers.

¹ Graduanda do curso Sistema de Informação do Centro Universitário UniSales

² Mestre em Informática

1. INTRODUÇÃO

Os aplicativos têm se tornado cada vez mais parte do nosso dia a dia com a ideia de facilitar a comunicação do usuário com serviços, exemplo disso o aplicativo do banco em que você não precisa ir até lá para resolver os problemas, tudo pode ser resolvido da palma da sua mão. Pensando nisso por que não promover através do aplicativo facilitar a comunicação de um professor com seu aluno?

Esse problema foi apresentado por um professor de inglês que tinha dificuldades para organizar e mostrar o desenvolvimento do aluno, alguns deles quando não atingiam o resultado que desejavam o culpava sendo que os mesmos nem apareciam para aula, em relato outros professores disseram o mesmo e com base nisso veio a ideia para esse sistema, em que o professor pode organizar e estruturar as aulas e alunos de sua turma.

Este projeto tem por objetivo o desenvolvimento de uma aplicação PWA (Progressive Web App) que faça o gerenciamento de alunos, turmas e aulas de um professor particular de inglês. Para isso será inicialmente desenvolvido uma aplicação web responsiva utilizando as tecnologias VueJS para o desenvolvimento do front-end, NestJS para back-end e MongoDB como base de dados.

2. REFERENCIAL TEÓRICO

Este capítulo tratará sobre as definições e os conceitos de *software* de gestão escolar e algumas das suas principais ferramentas. Trará também resumos de trabalhos relacionados com propostas similares, além de falar os fundamentos básicos sobre as tecnologias e ferramentas utilizadas para o desenvolvimento do protótipo sugerido.

2.1 Software de gestão escolar

O conceito de gestão escolar traz consigo a ideia de gerenciamento e administração eficiente de uma instituição de ensino. Com esse pensamento, gestão escolar nada mais é do que organizar todas as áreas e os aspectos determinantes para garantir a eficiente aprendizagem do estudante (PIEADADE e PEDRO, 2014).

Na pratica o software de gestão escolar e um sistema que controla todos os processos da instituição: financeiro, contábil, recebimento, retenção entre outras tarefas que são indispensáveis para a gestão escolar (VINDI, 2022).

No presente, softwares de gestão escolar estão sendo muito discutidos mediante a tantos avanços tecnológicos. Automatizando os processos garante uma integração em todos as etapas educacionais, seja nas atividades finais e nas atividades a sua volta, garantindo, portanto, uma gestão eficiente (PIEADADE e PEDRO, 2014).

2.1.1 Agenda Escolar

O sistema de agenda de papel sempre se fez muito presente no cotidiano escolar sendo um dos principais itens da lista de material escolar requisitada pelas instituições a fim de contribuir para comunicação entre professor, instituição e responsável. Mas acredita-se que esse meio não e 100% eficaz, afinal a entrega do comunicado pode falhar por qualquer motivo (SANTANA, 2018).

Por isso a implementação do uso de tecnologias no sistema educacional que facilitem, agilizem e torne confiável a troca de informações entre as entidades envolvidas é tão buscado nos dias de hoje já que o uso das mesmas tem se tornado tão presente no cotidiano da população (SILVA, 2019).

2.1.2 Pauta Eletrônica

Sistemas de controle de presença normalmente utilizados por pautas de papel em que o aluno é chamado e precisa responder que está presente é comumente utilizado nas escolas, o que não facilita nem agiliza o processo de gestão nem para o lado do professor nem do aluno que não consegue verificar sua quantidade de faltas até o final do trimestre (ou semestre) (TEBECHRANI, 1999).

Em piores casos o controle de presença é realizado através da coleta de assinaturas dos estudantes em um papel que roda pela sala.

Pensando nisso software de gestão escolar incluem a ferramenta de controle de presença como o Etutore e o Dksoft (BORRACHA).

2.2 Trabalhos relacionados

Durante a realização deste trabalho, foram encontrados outros temas de abordagem com propostas similares.

2.2.1 Implementação de um sistema de gestão escolar em uma escola particular de São Paulo

Esse artigo tem por objetivo demonstrar a funcionalidade de um aplicativo chamado *ClassApp* para substituir a agenda de papel. A autora explica os benefícios do uso do aplicativo e as facilidades que ele trará na gestão escolar já que engloba todas as áreas da instituição. Foi demonstrado as dificuldades referentes a implantação do sistema e utilizou de formulários para definir a satisfação do usuário. Como resultado mostrou uma otimização do processo de comunicação mesmo apresentando dificuldades no processo (SANTANA, 2018).

2.2.2 Gestão escolar de um centro de línguas: a criação do software ci-web

O artigo tem por objetivo a criação de um *Software* para gerenciamento de um centro de idiomas, que tem por objetivo auxiliar no processo administrativo. O autor descreve a forma que será desenvolvido e as tecnologias utilizadas no *Software*. Como resultado ele apresenta algumas telas das funcionalidades do sistema e menciona um avanço incremental no desenvolvimento para melhorar os resultados (COELHO, 2021).

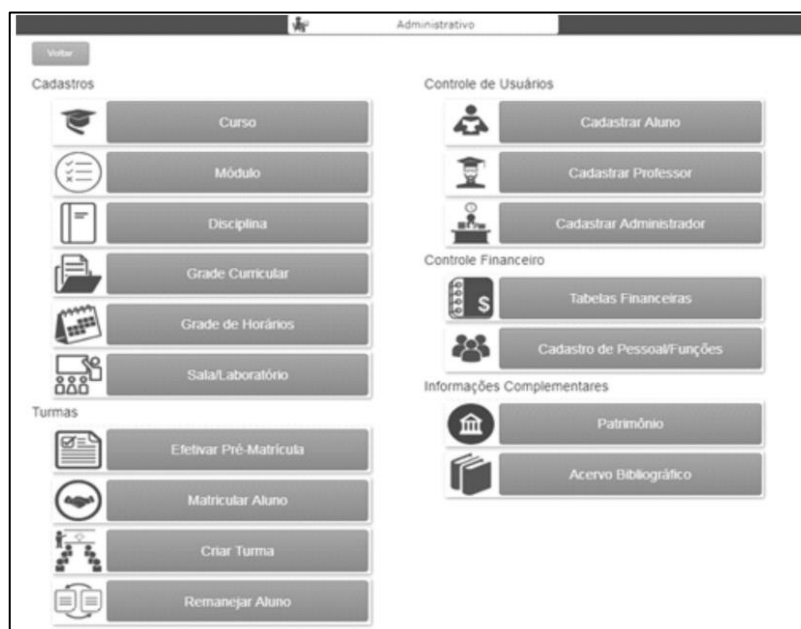
As figuras 1 e 2 são telas montadas pela escritora do artigo para demonstrar o sistema desenvolvido.

Figura 1 - Captura da tela principal do software CI-Web



Fonte: Artigo Gestão escolar de um centro de línguas: a criação do software ci-web escrito por Iandra Maria Weirich da Silva Coelho

Figura 2 - Captura de tela da seção administrativa do software CI-Web



Fonte: Artigo Gestão escolar de um centro de línguas: a criação do software ci-web escrito por Iandra Maria Weirich da Silva Coelho

2.2.3 O desenvolvimento de um sistema online de gerenciamento escolar

Esse artigo tem por objetivo implementar um sistema de gerenciamento escolar para auxiliar no processo administrativo e reduzir o consumo de papel. O autor apresenta todo o processo de desenvolvimento do software, os diagramas, casos de uso, modelagem do banco de dados. Como resultado ele traz um sistema funcional demonstrando as telas do sistema e possíveis melhorias para o futuro (PEREIRA, 2019).

2.3 Ferramentas para desenvolvimento da aplicação

Como intuito de relatar o processo de criação dessa aplicação é descrito neste capítulo as ferramentas e tecnologias utilizadas para o desenvolvimento dessa atividade.

2.3.1 Protocolo HTTP

Protocolo HTTP vem de *HyperText Transfer Protocol* (Protocolo de Transferência de Hipertexto), é sobre isso que toda a estrutura da *web* é construída, o protocolo em si é formado por muitas estruturas: a URL de destino, o verbo, cabeçalhos, códigos de status e o corpo da mensagem. E tudo isso faz parte da comunicação entre cliente e servidor utilizando o sistema de *request/response* (requisição/resposta) (LISBOA, 2012).

Na tabela abaixo estão listados os tipos de verbos também chamados de métodos que são enviados juntamente com a URL informando qual a natureza da requisição.

Tabela 1 - Verbos HTTP

VERBO	FUNÇÃO
GET	Solicitações de uso devem apenas recuperar dados
HEAD	O método pede uma resposta idêntica a um pedido, mas sem o corpo de resposta.
POST	O método submete uma entidade ao recurso especificado, muitas vezes causando uma alteração no estado ou efeitos colaterais no servidor.
PUT	O método substitui todas as representações atuais do recurso de destino pela carga de solicitação.
DELETE	O método exclui o recurso especificado.

CONNECT	O método estabelece um túnel para o servidor identificado pelo recurso alvo.
OPTIONS	O método descreve as opções de comunicação para o recurso de destino.
TRACE	O método realiza um teste de loop-back de mensagens ao longo do caminho para o recurso de destino.
PATCH	O método aplica modificações parciais a um recurso.

Fonte: Adaptado de (MOZILLA, 2022)

Como pode-se ver existem vários métodos para utilizar na requisição, porém os mais comuns são o *GET*, *POST*, *PUT* e *DELETE* utilizados para criar um serviço denominado *RESTful* que promove um conjunto simples de funcionalidades para as requisições, essas são chamadas de CRUD (*Create, Read, Update, Delete* – Criar, Ler, Atualizar, Apagar) as vezes é visto implementações que utilizam o método *PATCH* mas não é comum (MITCHELL, 2013).

Nas figuras abaixo estão listados os cabeçalhos mais utilizados nas requisições e respostas segundo Lorna Jane.

Tabela 2 - Cabeçalhos HTTP mais comuns

Cabeçalho	Usado	Observação
Accept	Solicitação	Mostra os formatos, com uma indicação da preferência. De maneira que o cliente solicitante possa atender
Authoriza-tion	Solicitação	São informações de formato livre para provar a identidade do usuário
Cookie	Solicitação	São pares de chave/valor separados por ponto e virgula
Content-Length	Solicitação/Res-posta	Qualquer solicitação ou resposta com conteúdo no corpo tem que tem Content-length com tamanho em bytes no cabeçalho.
Content-Type	Solicitação/Res-posta	Prover informações sobre o formato do conteúdo

ETag	Resposta	É identificador da versão do recurso que está sendo utilizado
If-Modified-Since e If-None-Match	Solicitação	Informa ao servidor que há uma cópia em cache desse recurso
Last-Modified	Resposta	Fornece informações sobre quando esse recurso foi modificado
Location	Resposta	Fornece informações sobre uma localização quando houver redirecionamento
Set-Cookie	Resposta	Envia Cookies para que sejam armazenados no cliente
User-Agent	Solicitação	Fornece informações sobre o software do cliente que está fazendo a solicitação

Fonte: Adaptado de (MITCHELL, 2013) página 134 e 135 apêndice B

Já os códigos de status são utilizados somente nas respostas as requisições para informar ao cliente o que ocorreu afim de padronizar a comunicação entre o cliente e o servidor. Na tabela a seguir de códigos que podem ser utilizados (FERNANDES, 2020).

Tabela 3 - Status mais comuns

Código	Significado	Observação
1xx	Informativo	Informa que a requisição ainda está em processamento e o cliente pode aguardar uma resposta
2xx	Sucesso	Indica que a requisição foi recebida, compreendida, aceita e processada
3xx	Redirecionamento	O cliente deve realizar mais alguma ação
4xx	Erro do lado do cliente	Informa que contém algum erro na requisição do cliente
5xx	Outros erros	Erros genéricos ou não tratados do lado do servidor

Fonte: Adaptado de (MITCHELL, 2013) e (FERNANDES, 2020)

2.3.2 Progressive Web App (PWA)

Progressive Web App (PWA) são aplicações web desenvolvidas para funcionar em plataformas específicas. Por serem lançadas em seu próprio ambiente local tem acesso a ler e gravar dados, acessar a câmera, microfone, *Bluetooth* e até mesmo ser usado sem acesso à rede entre outras permissões. Em termos de alcance e recursos aplicações nativas se saem melhor em termos de capacidade, mas as aplicações web se saem melhor em termos de alcance e acesso dos usuários, já o *PWA* tem o melhor dos dois mundos, por ser um desenvolvimento *web* e funcionar de forma nativa. Esse tipo de sistema é executado em uma janela independente, fora do navegador (RICHARD e LEPAGE, 2020).

2.3.3 Vue.js

Vue.js é um *framework JavaScript* que tem sua arquitetura parcialmente inspirada na estrutura MVVM (*Model-View-View-Model*). Tem como por objetivo ser uma ferramenta que proporciona a criação de aplicações de pequena e larga escala garantindo a fácil integração com projetos já existentes e outras bibliotecas (VUE, 2022)

Utiliza-se de componentes que podem ser reaproveitados em qualquer parte do código, são formados por um único arquivo (*.vue*) que encapsula o *HTML*, *CSS* e *JS* facilitando a manutenção. Uma das características principais do *Framework* é ser reativo, os dados são objetos *JavaScript* que sofrem alteração somente quando a camada de visualização é atualizada (VUE, 2022).

2.3.4 Node.js

Node.js foi desenvolvido em *JavaScript* para o *Back End*, executa de forma assíncrona em tempo de execução e é orientado a eventos, projetado para aplicações de rede escaláveis lidando assim com múltiplas conexões. (CASCIARO e MAMMINO, 2020)

Seu núcleo é construído com o menor conjunto de funcionalidades permitindo a criação de um ecossistema que vive fora do núcleo chamado *userland* (ou *userspace*), que traz um enorme impacto por dar liberdade para a comunidade (desenvolvedores que utilizam *Node.js*) a experimentar os mais diversos conjuntos de soluções ao invés de uma solução lenta e rígida que é vista em outros sistemas (CASCIARO e MAMMINO, 2020).

Outra grande característica do Node.js e seus sistema de módulos que vem da ideia de construir pequenos blocos de códigos, não na questão de tamanho, mas de escopo que sejam reutilizáveis para criar aplicativos e bibliotecas, desenvolvendo assim uma aplicação que seja mais fácil de usar, compreender, testar, manter e de fácil utilização em navegadores (CASCIARO e MAMMINO, 2020).

2.3.5 NestJS

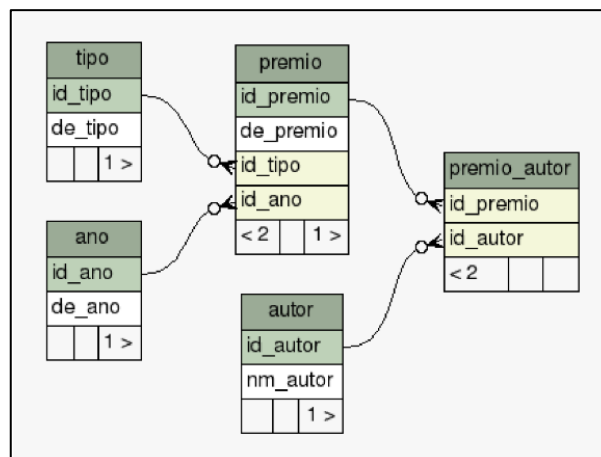
NestJS é um *framework* para construção de aplicações escaláveis e eficientes em *Node.js* no lado do servidor, usa-se *JavaScript* progressivo, suporta desenvolvimento em *TypeScript*, mas permite usar *JavaScript* puro e combina elementos de *OOP* (programação orientada a objetos), *FP* (programação funcional) e *FRP* (programação reativa funcional). Por debaixo dos panos usa as estruturas de servidor *HTTP* (NESTJS).

2.3.6 MongoDB

MongoDB é um banco de dados orientado a documento, ou seja, não é um banco de dados relacional (SQL), ele não usa o sistema de tabelas convencional e sim o de *collections* o que permite que o banco de dados se comporte da maneira que a aplicação deseja não precisando de uma estrutura anteriormente definida como no sistema de tabelas (MONGODB).

Na figura a seguir é demonstrado como seria a estrutura do banco de dados que contenham a informação sobre os ganhadores do prêmio IgNobel na estrutura de dados relacionais e na Figura 4 como seria o código necessário para selecionar os ganhadores (HOWS, MEMBREY e PLUGGE, 2019).

Figura 3 - Modelo do banco de dados relacional



Fonte: (HOWS, MEMBREY e PLUGGE, 2019)

Figura 4 - Código necessário para selecionar os dados

```
select
p.de_premio, t.de_tipo, a.de_ano , au.nm_autor
from premio p, tipo t, ano a, premio_autor pa, autor au
where p.id_premio = pa.id_premio
and p.id_tipo = t.id_tipo
and p.id_ano = a.id_ano
and pa.id_autor = au.id_autor
```

Fonte: (HOWS, MEMBREY e PLUGGE, 2019)

Vê-se que é necessário no mínimo quatro tabelas e como um prêmio pode ter mais de um autor é preciso criar uma tabela auxiliar, que chamamos de *premio_autor*. Seu código é extenso e precisando fazer o *select* em várias tabelas isso provoca lentidão o que para um site em produção que precisa exibir esses dados com agilidade não é nada satisfatório (HOWS, MEMBREY e PLUGGE, 2019).

Por outro lado, a figura abaixo mostra as mesmas informações dos ganhadores do IgNobel sendo armazenadas com a estrutura de *collections* do MongoDB e o código necessário para retornar os ganhadores (HOWS, MEMBREY e PLUGGE, 2019).

Figura 5 - Estrutura do dado em um banco de dados não relacional

```
{
  "ano" : 1992,
  "tipo" : "Medicina",
  "autores" : [
    "F. Kanda",
    "E. Yagi",
    "M. Fukuda",
    "K. Nakajima",
    "T. Ohta",
    "O. Nakata"],
  "premio" : "Elucidação dos Componentes Químicos Responsáveis
    pelo Chulé do Pé (Elucidation of Chemical
    Compounds Responsible for Foot Malodour),
    especialmente pela conclusão de que as pessoas
    que pensam que têm chulé, têm, e as que pensam
    que não têm, não têm."
}
```

Fonte: (HOWS, MEMBREY e PLUGGE, 2019)

Figura 6 - Código necessário para selecionar os dados

```
ganhadoresPremioIgNobel.find()
```

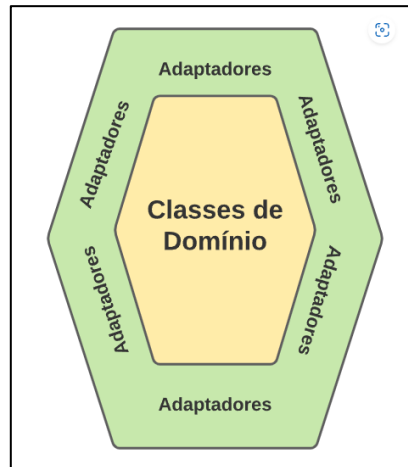
Fonte: Próprio autor

Nessa estrutura todas as informações do ganhador são representadas por uma única *collection* o que torna menor a linha de código que precisamos fazer para retornar toda a lista de ganhadores do prêmio tornando assim o sistema muito mais ágil (HOWS, MEMBREY e PLUGGE, 2019).

2.3.7 Arquitetura Hexagonal

O modelo de arquitetura hexagonal tem como seu objetivo construir códigos reutilizáveis, fáceis de testar e que sejam tecnologicamente independentes. Essa arquitetura divide o sistema em duas camadas comumente chamadas de *domínio* e *adapter*, pode-se ver na figura a seguir como visualmente essa arquitetura é representada (VALENTE, 2022).

Figura 7 - Arquitetura Hexagonal



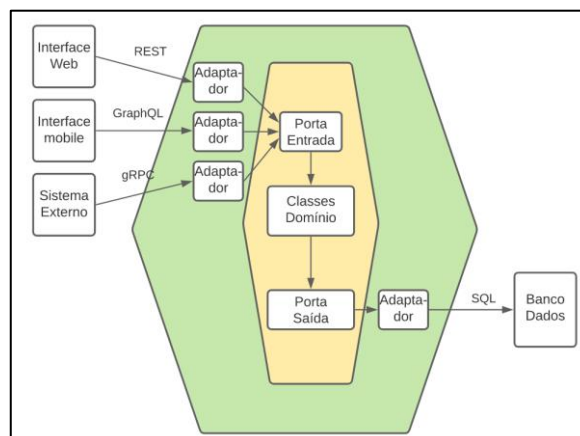
Fonte: (VALENTE, 2022)

Na camada de *domínio* está todo o código responsável pela regra de negócio e na camada de *adapter* códigos relacionados a infraestrutura, integrações com API externas, interfaces com o usuário entre outros (VALENTE, 2022).

Essa separação permite desacoplar o negócio da tecnologia, por exemplo, caso seja necessário mudar o banco de dados, a camada de domínio não será afetada, isso também facilita caso o sistema seja distribuído em mais de uma plataforma (Web, mobile etc.), a camada de domínio onde está toda a regra de negócio pode ser compartilhada para essas outras plataformas sem problemas (VALENTE, 2022).

Na figura a seguir mostra como seria a utilização do modelo de arquitetura hexagonal em um sistema de gerenciamento de bibliotecas.

Figura 8 - Arquitetura Hexagonal do sistema de bibliotecas



2.3.8 Visual Studio Code

Visual Studio Code é um editor de código-fonte leve, tem suporte integrado a *JavaScript*, *TypeScript* e *Node.js* e tem uma vasta biblioteca de extensões para outras linguagens como *C++*, *C#*, *Java*, *Python*, *PHP*, *Go*, *.NET*. Disponível para *Windows*, *Linux* e *macOS*. Alguns dos recursos que essa ferramenta oferece são *IntelliSense*, que agiliza a digitação do código e *Run And Bug* que lhe permite depurar o código em tempo de execução (CODE, 2022).

3. METODOLOGIA DE PESQUISA

O artigo em questão tem por fim facilitar o processo de gestão dos alunos de um professor de inglês, onde ele conseguirá expor de forma mais clara o progresso e avanço do aluno.

Em vista disso o seguinte artigo constitui-se por uma pesquisa de natureza aplicada, e tem como classificação de seus objetivos a pesquisa bibliográfica e descritiva visando descrever sobre *softwares* de gestão escolar, e apurar a satisfação do professor na utilização da ferramenta. Foi utilizado a abordagem qualitativa que visa examinar evidências baseadas nos resultados adquiridos.

Para a realização da pesquisa foi disponibilizado ao professor o link do *web-site* para que ele pudesse testar. Depois dos testes foi solicitado que respondesse a um formulário simples de satisfação.

A pesquisa foi desenvolvida para um professor particular de inglês que leciona de forma online para uma ou um grupo de pessoas.

4. DESENVOLVIMENTO

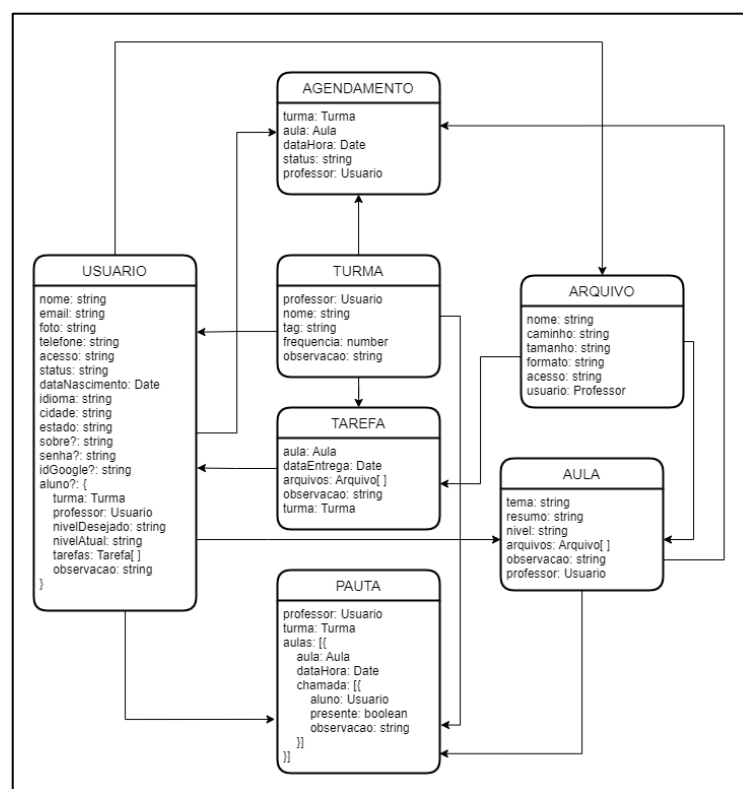
A aplicação desenvolvida para esse trabalho trata-se de um *software* de gestão escolar para um professor particular, onde ele poderá criar, editar e excluir turmas, alunos, tarefas e agendamento das aulas.

4.1 Modelagem de dados

Como havia sido dito, o banco de dados escolhido para o desenvolvimento dessa aplicação foi o MongoDB, que não utiliza um modelo de dados definido e estruturado, sendo assim cada entidade representada na Figura 9 são as possíveis *collections* do sistema.

Os atributos que contêm uma interrogação logo após o nome como “senha?” são atributos não obrigatórios quando o banco for salvar os registros.

Figura 9 - Modelo do banco de dados



Fonte: Próprio autor

4.2 Arquitetura

A escolha da arquitetura hexagonal vem da possibilidade de crescimento do protótipo, como está sendo trabalhado para uso aplicado as chances de alterar alguma tecnologia posteriormente ou alterar algum contrato entre cliente e servidor é bem grande e essa arquitetura traz essa facilidade de mudança como um dos seus pontos chaves além da facilidade de integração com outras plataformas caso desejado.

4.3 Modelagem de fluxo

As figuras utilizadas para demonstrar o fluxo do sistema foram criadas utilizando a ferramenta *online* draw.io³ visando esclarecer de forma breve os processos nas ações que podem ser executadas pelo professor.

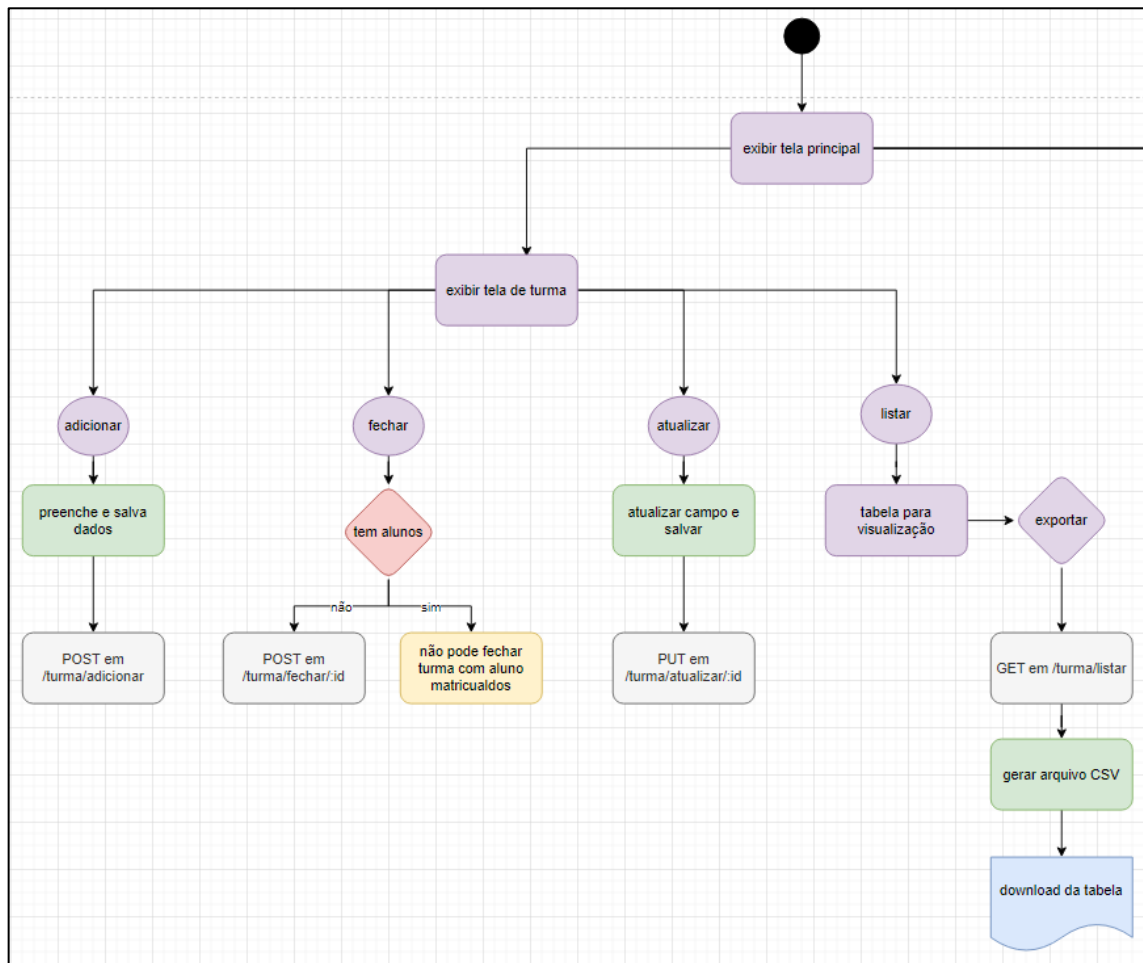
O fluxo da tela principal foi montado de forma que o professor consiga ter as informações principais e de maior interesse logo após o login, como o quantitativo de alunos, o quantitativo de turmas entre outros dados isso ficará mais claro no tópico que abordará as telas do protótipo.

Após a construção de alguns fluxos percebeu-se que pelas boas práticas de designer e por sugestões do professor o interessante seria que as ações ficassem somente em uma única página. Pensando nisso as figuras a seguir demonstram os fluxos presentes nas páginas de turma, aula e aluno.

A figura 10 representa o processo na página de turma, de forma objetiva o professor terá as ações de visualizar e exportar esses dados, editar, excluir e adicionar uma turma. Ao chegar no *back-end* os dados são processados e validados retornando erro caso ocorra algum problema, o campo de professor presente na modelagem é preenchido automaticamente e somente alguns campos podem ser editados, o sistema também valida caso o fechamento da turma seja solicitado, caso exista alunos matriculados essa ação não será permitida.

³ Disponível em <<https://drawio-app.com/>> Acesso em: 22 out, 2022.

Figura 10 - Fluxo na tela da turma



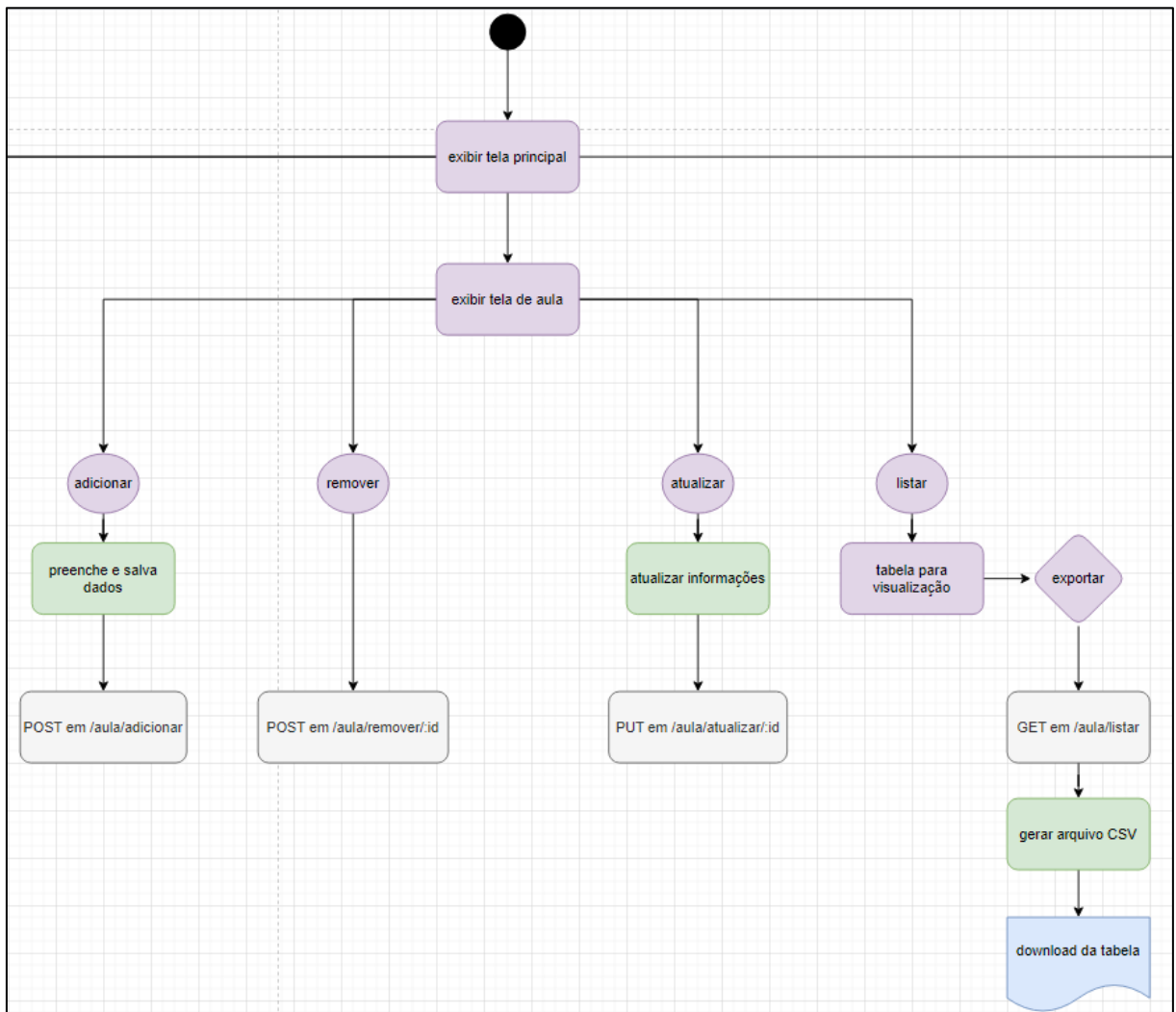
Fonte: Próprio autor

A figura 11 representa a página de aula, o professor terá as ações de visualizar e exportar esses dados, editar, excluir e adicionar uma aula. Ao receber esses dados o *back-end* válida e processa essas informações e salva o registro.

O professor somente poderá excluir uma aula caso ela não tenha sido atribuída a nenhuma tarefa ou agendamento.

No processo de geração do relatório será trago além das informações das aulas os agendamentos e tarefas que estão atribuídas.

Figura 11 - Fluxo na tela de aula

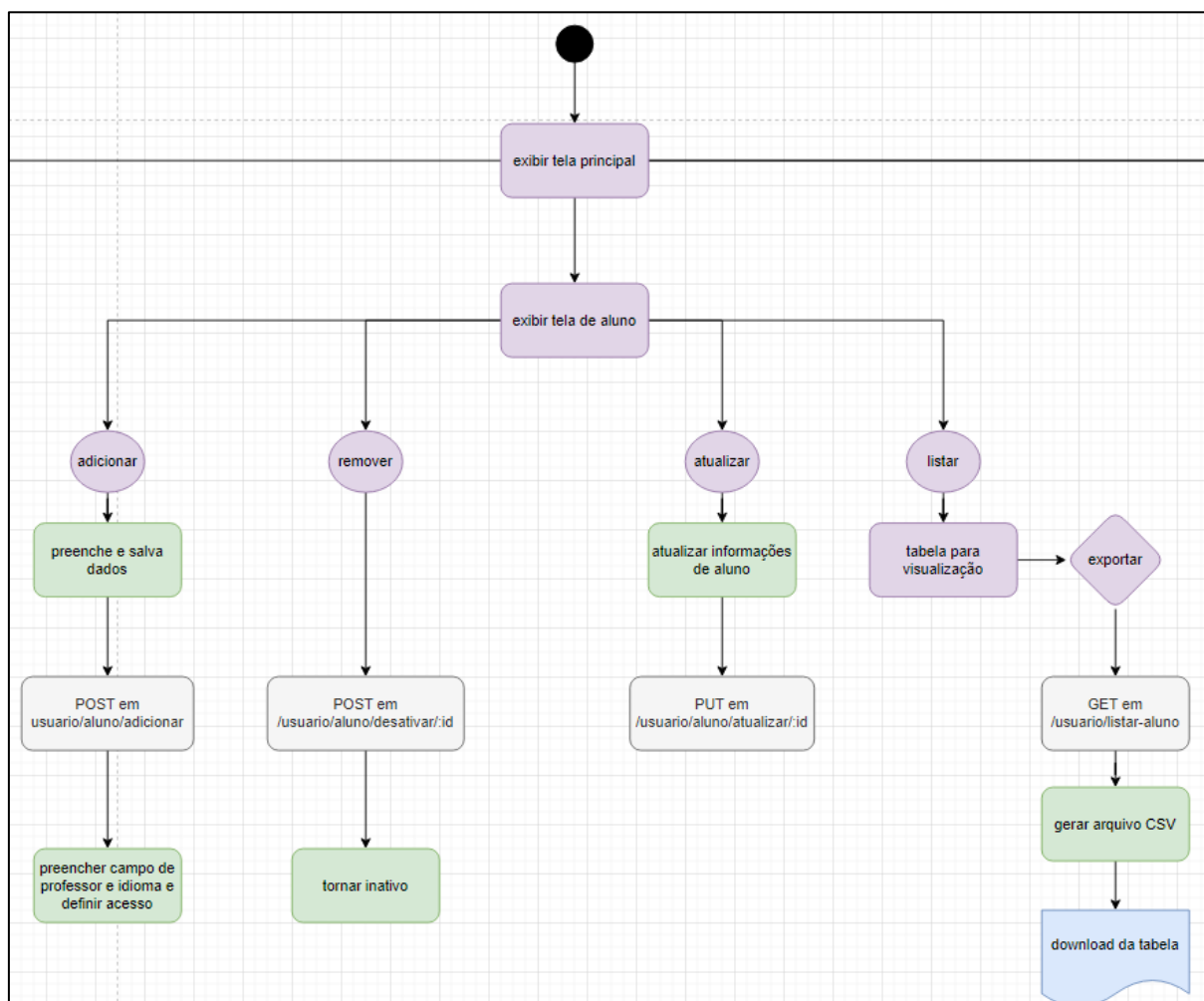


Fonte: Próprio autor

A figura 12 representa a página de aluno, o professor terá as ações de visualizar e exportar esses dados, editar, excluir e adicionar um aluno. Quando um professor adiciona um usuário automaticamente o *back-end* entende que ele é um aluno assim preenchendo os dados conforme a modelagem de dados.

Assim como no processo de exportação de aula, quando for solicitado a exportação dos dados dos alunos trará todas as informações que ele estiver associado, como relação de tarefas e pauta.

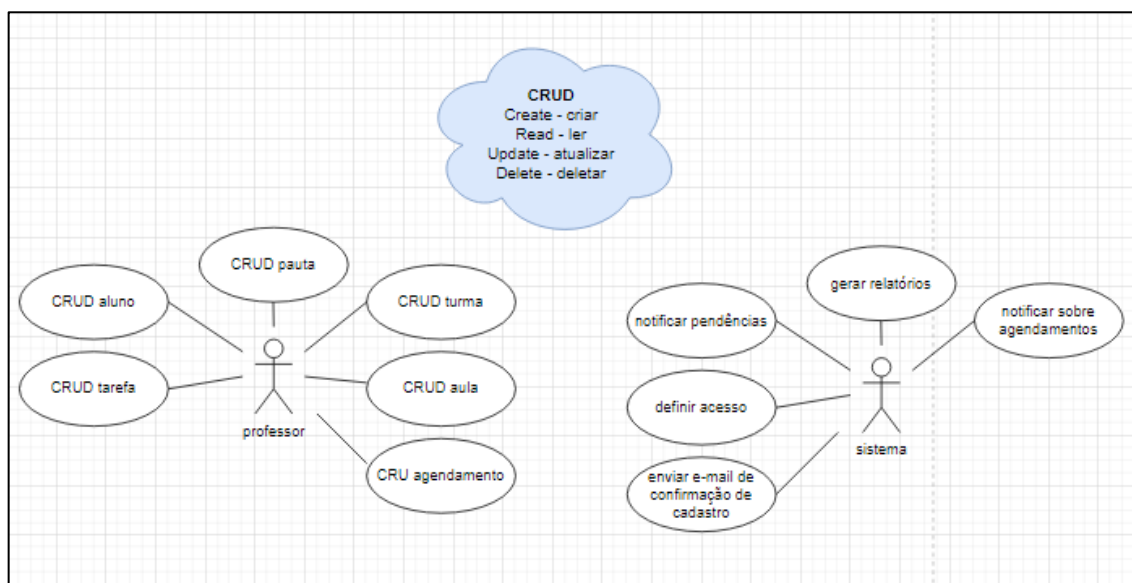
Figura 12 - Fluxo da página de aluno



Fonte: Próprio autor

Para complementar a figura a seguir é um diagrama de caso de uso ilustrando outras ações que o professor e a API executarão durante o funcionamento do sistema.

Figura 13 - Diagrama de caso de uso



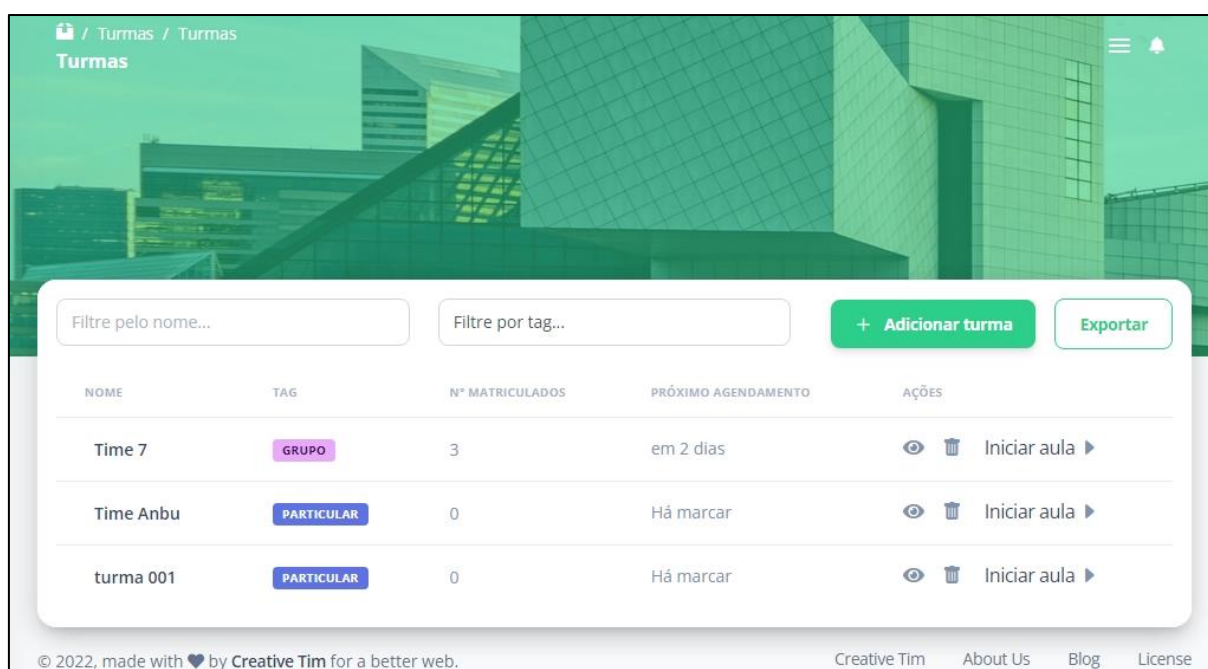
Fonte: Próprio autor

4.4 Protótipo das telas

Como a proposta deste trabalho é a entrega de um protótipo de *software* de gestão funcional foram desenvolvidas telas para cadastro, listagem e apresentação de resultados.

Na figura a seguir vê-se a tela listagem de turmas, ao clicar no botão “Adicionar turma” abre-se um modal com os campos para serem preenchidos. Ao clicar sobre o olho na coluna de ação é redirecionado a tela de perfil da turma

Figura 14 - Tela da tabela de turmas



Fonte: Próprio autor

Na tela de perfil que é demonstrado na figura 15, tem-se *cards* com informações dos alunos matriculados nessa turma, os agendamentos pendentes, tarefas cadastradas, e as informações sobre a turma que podem ser editadas nessa mesma tela. No *card* de “Aulas ministradas” é demonstrado o histórico de aulas que essa turma já participou.

Figura 15 - Tela de perfil da turma

Time 7
GRUPO

Informações

Nome
Time 7

Tag
GRUPO

Frequência
5 vezes por semana

Observação
teste de educação

Salvar mudanças

Próximos agendamentos

28/11/2022 13:30:00
Teste de sistema

Marcar Aula

Aulas ministradas

26/11/2022 10:30:23
Fortaleza de Gato

Matriculados

Sakura Haruno
Hi! I need more information...

Sasuke Uchiha
Hi! I need more information...

Naruto Uzumaki
Hi! I need more information...

Adicionar Aluno

Tarefas

28/11/2022 13:30:00
Enviar um documento em inglês listando seus filmes favoritos e um resumo com suas próprias palavras sobre ele. Tudo em inglês.

25/11/2022 13:30:00
testes

22/11/2022 13:30:00
testes

Adicionar Tarefa

© 2022, made with by Creative Tim for a better web. Creative Tim About Us Blog License

Fonte: Próprio autor

Na figura 16 vê-se a tela de “Aula em andamento” que é exibida logo após o professor clicar na ação “Iniciar aula” na figura 14. Aqui o professor consegue visualizar os

alunos e marcar quem estava presente na aula e escrever alguma observação necessária, como, informar o atraso do aluno. Ele também tem acesso ao plano de aula que ele estabeleceu na criação da aula.


Figura 16 - Tela de aula em andamento

Aula em andamento

Chamada


Marque quem estiver presente

Sakura Haruno




Escreva detalhes sobre o aluno durante a aula

Sasuke Uchiha



Escreva detalhes sobre o aluno durante a aula

Naruto Uzumaki



Escreva detalhes sobre o aluno durante a aula

Fortaleza de Gato


Objetivo:

Pegar uma cópia da pata de Nekomata

Realização:

O time 7 precisa se infiltrar dentro da fortaleza, e se camuflar a fim de conseguir se aproximar do felino, o mesmo é da raça Ninneko, gatos que não são dóceis, ou que pelo menos não gostam de muito contato dos humanos.

Arquivos:

 arquivo de teste.pdf

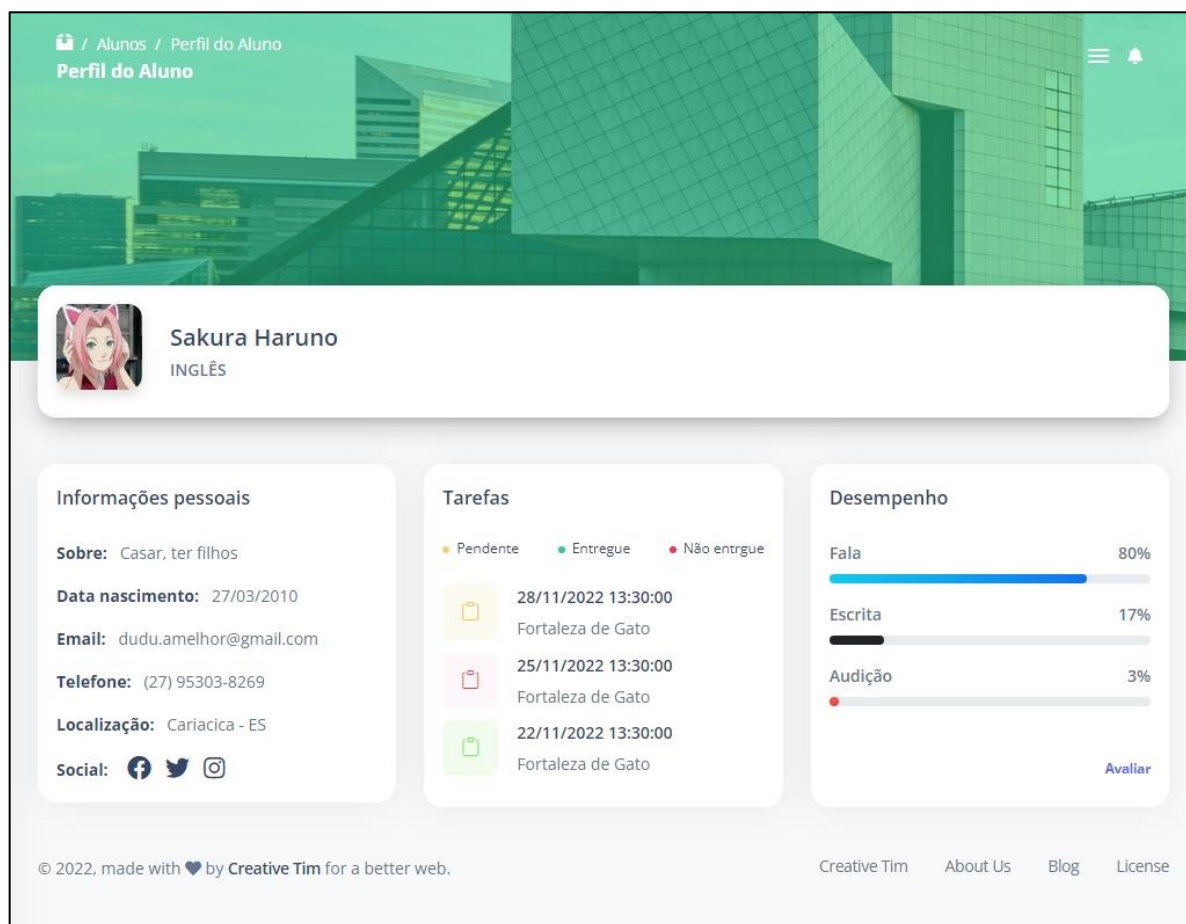
Ver

Finalizar aula

Fonte: Próprio autor

Na figura 17 tem-se a tela de perfil do aluno que o professor acessa ao selecioná-lo na tela de listagem de alunos. Aqui o professor consegue mostrar ao estudante caso necessários as tarefas que foram passadas e *status* delas (se o aluno fez ou não a atividade proposta) e desempenho dele.

Figura 17 - Tela de perfil do aluno



Fonte: Próprio autor

Foram criadas também telas de listagem de alunos como foi mencionado, aulas, agendamentos e tarefas, mas o estilo e estrutura é bem semelhante a de turma por isso não foi visto necessidade de demonstração.

5. CONSIDERAÇÕES FINAIS

O presente trabalho teve como objetivo o desenvolvimento de um protótipo funcional de um PWA de *software* de gestão escolar para um professor particular de inglês.

O principal objetivo foi alcançado. O sistema é funcional permitindo que o professor já possa utilizar tanto via *web-browser* como pela instalação do *PWA* até o presente momento da escrita deste artigo o sistema está rodando somente em *localhost*⁴.

As telas para o acesso do aluno foram deixadas para trabalhos futuros, mesmo assim o *back-end* já lida com esse controle de acesso para as rotas criadas através do *token* gerado após o processo de autenticação.

⁴ Rodando somente em servidor local na máquina do autor

Para o sistema de autenticação o usuário pode escolher tanto via OAuth2 com o google ou padrão de *e-mail* e senha, mas não foi abordado nesse trabalho já que o foco estava no desenvolvimento do software.

Como dito anteriormente uma das melhorias futuras seria a implantação das telas para o acesso de ALUNO para que ele mesmo possa acompanhar o próprio desenvolvimento e enviar suas atividades por exemplo. Um outro ponto seria a criação de uma tela de *dashboard* para análise de dados e algum tipo de validação condicional para o botão de “iniciar aula” presente na tela de turma, para que não seja possível iniciar aula para uma turma sem agendamento ou com muita antecedência por exemplo.

6. REFERÊNCIAS

BORRACHA, Blog B. S. D. Barcelona Superficies. **7 sistemas de gestão escolar que vão tornar sua escola mais eficiente.** Disponível em: <https://barcelonasuperficies.com.br/blog/sistemas-gestao-escolar/>.

BOYD, Ryan. **Getting Started with OAuth 2.0.** 1. ed. California: O'Reilly Media, 2012.

CASCIARO, Mario; MAMMINO, Luciano. **Node.js Design Patterns.** Birmingham: Packt Publishing Ltd., 2020.

CLASSAPP, WebSite. WebSite ClassApp. **Quais as vantagens de se trocar a agenda por um app escolar?** Disponível em: <https://www.classapp.com.br/artigos/app-escolar-vantagens>.

COCKBURN, Alistair. WikiWikiWeb. **Hexagonal Architecture.** Disponível em: <http://wiki.c2.com/?HexagonalArchitecture>.

CODE, Visual S. Microsoft. **Visual Studio Code**, 13 set. 2022. Disponível em: <https://code.visualstudio.com/docs/editor/intellisense>.

COELHO, Iandra M. W. D. S. **Gestão escolar de um centro de línguas: a criação do software ci-web.** Amazonas. 2021.

FERNANDES, Henrique M. Blog. **Lista Completa de Códigos HTTP**, 2020. Disponível em: <https://marquesfernandes.com/tecnologia/lista-completa-de-codigos-http/#:~:text=4XX%20%E2%80%93%20Erro%20de%20cliente%20%20%20,n%C3%A3o%20foi%20encontrado.%20%20%20%20more%20rows>. Acesso em: 12 out. 2022.

HOWS, David; MEMBREY, Peter; PLUGGE, Eelco. **Introdução ao MongoDB.** São Paulo: Novatec, 2019.

LISBOA, Flávio G. D. S. **Criando Aplicações PHP com Zend e Dojo.** 2. ed. São Paulo: novatec, 2012.

MITCHELL, Lorna J. **Web Services em PHP APIs para a web moderna.** 1. ed. São Paulo: Novatec, 2013.

MONGODB. MongoDB. **MongoDB.** Disponível em: <https://www.mongodb.com/>.

MOZILLA, Developer. mdn web docs. **HTTP request methods**, 2022. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>. Acesso em: 11 out. 2022.

NESTJS, Blog. NestJS. **NestJS**. Disponível em: <https://nestjs.com/>.

NODE.JS. Node.js. **Node.js**. Disponível em: <https://nodejs.org/en/about/>.

PEREIRA, Daniel B. **O desenvolvimento de um sistema online de gerenciamento escolar**. Niterói. 2019.

PIEIDADE, João; PEDRO, Neuza. **Tecnologias digitais na gestão escolar: Práticas, proficiência e necessidades de formação dos diretores escolares em Portugal**. Instituto de Educação da Universidade de Lisboa. Lisboa, p. 25. 2014.

RICHARD, Sam; LEPAGE, Pete. O que são Progressive Web Apps? **web.dev**, 24 Fevereiro 2020. Disponível em: <https://web.dev/what-are-pwas/>.

SANTANA, Lorena K. R. **IMPLEMENTAÇÃO DE UM SISTEMA DE GESTÃO ESCOLAR EM UMA ESCOLA PARTICULAR DE SÃO PAULO**. São Paulo. 2018.

SILVA, Raquel S. D. **Tecnologias da informação e comunicação na educação infantil**. UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL. Porto Alegre, p. 39. 2019.

TEBECHRANI, Elione A. C. **Frequência em sala de aula: Um estudo a partir da ótica de alunos e professores de um curso de graduação**. Universidade Estadual de Campinas. Campinas, p. 141. 1999.

VALENTE, Marco T. **Engenharia de Software Moderna**. 1. ed. [S.l.]: Independente, 2022.

VINDI, Blog. Vindi. **Vind**, 08 Setembro 2022. Disponível em: <https://blog.vindi.com.br/os-12-principais-sofware-de-gestao-escolar/>.

VUE. Vue.js. **Introdução a Vue.js**, 2022. Disponível em: <https://vuejsbr-docs-next.netlify.app/guide/introduction.html#o-que-e-vue-js>. Acesso em: 20 ago. 2022.

APENDICE A – FORMULÁRIO DE SATISFAÇÃO

1) O que achou da usabilidade do sistema?

É um sistema simples e fácil de usar, os cadastros são simples e bem sucintos.

2) Encontrou alguma dificuldade em utilizar o sistema?

Não.

3) O sistema atende suas funcionalidades?

Sim.

4) O quanto satisfeito está com o sistema?

Bastante, vai me ajudar a ter um controle melhor das faltas dos alunos e poder mostrar a eles.

5) Sugere alguma melhoria?

Que possa trocar o idioma da plataforma, atualmente está tudo em português, mas seria interessante poder trocar para inglês.