
**UNIVERSIDADE FEDERAL DE CAMPINA GRANDE**
**CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA**
**CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

**JOSÉ THIAGO DOS SANTOS SILVA**

# ATAUX:
# UMA FERRAMENTA PARA AUXILIAR O ACOMPANHAMENTO DE SUBMISSÕES DE QUESTÕES DE PROGRAMAÇÃO

**CAMPINA GRANDE - PB**

**2021**

**JOSÉ THIAGO DOS SANTOS SILVA**

# ATAUX:
# UMA FERRAMENTA PARA AUXILIAR O ACOMPANHAMENTO DE SUBMISSÕES DE QUESTÕES DE PROGRAMAÇÃO

**Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.**

**Orientadora: Professora Dra. Melina Mongiovi.**

**CAMPINA GRANDE - PB**

**2021**


G635a Silva, José Thiago dos Santos.
ATAUX – Uma ferramenta para auxiliar o acompanhamento de submissões de questões de programação. / José Thiago dos Santos Silva. – 2021.

12 f.

Orientadora: Profa. Dra. Melina Mongiovi.
Trabalho de Conclusão de Curso - Artigo (Curso de Bacharelado em Ciência da Computação) - Universidade Federal de Campina Grande; Centro de Engenharia Elétrica e Informática.

1. Programação. 2. Problemas de programação. 3. Scrum. 4. Post-Study System Usability Questionnaire - PSSUQ. 5. Usabilidade de sistema – avaliação. 6. Algoritmos. 7. Disciplina Análise e Técnicas de Algoritmos - UFCG. 8. Desenvolvimento de software. 9. ATAUX – ferramenta de programação. I. Mongiovi, Melina. II. Título.

CDU:004.4(045)

**Elaboração da Ficha Catalográfica:**

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

**JOSÉ THIAGO DOS SANTOS SILVA**

# ATAUX:
# UMA FERRAMENTA PARA AUXILIAR O ACOMPANHAMENTO DE SUBMISSÕES DE QUESTÕES DE PROGRAMAÇÃO

**Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.**

## BANCA EXAMINADORA:

**Professor Dr.(a.) Melina Mongiovi**

**Orientador – UASC/CEEI/UFCG**

**Professora Dr.(a.) João Arthur Brunet Monteiro**

**Examinador – UASC/CEEI/UFCG**

**Professor Tiago Lima Massoni**

**Professor da Disciplina TCC – UASC/CEEI/UFCG**

**Trabalho aprovado em 25 de maio de 2021.**

**CAMPINA GRANDE - PB**

# RESUMO (ABSTRACT)[1]

Codeforces is a platform used to improve the skill for developing algorithms to solve programming problems. However, it is limited to be used directly in the classroom, because it cannot fulfill some academic needs required in a course, for example, resolutions control, to keep up with students performance, provide grade spreadsheets, etc. Therefore, the current system aims to supply these issues at creation and tracking lists of algorithms questions. The methodology adopted at the system development was based on scrum, furthermore, the method used to evaluate the usability was based on Post-Study System Usability Questionnaire (PSSUQ). The achieved results at the system evaluation had shown good satisfaction rate by the testers, in such a way that the professors can count with Ataux to help in an algorithms subject.

[1] Caso seu artigo esteja em inglês, coloque aqui o resumo em português; caso esteja o artigo em português, coloque aqui o resumo em inglês.

# ATAUX - Uma ferramenta para auxiliar o acompanhamento de submissões de questões de programação

José Thiago dos Santos Silva
jose.thiago.silva@ccc.ufcg.edu.br
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil

Melina Mongiovi
melina@computacao.ufcg.edu.br
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil

## RESUMO

O Codeforces é uma plataforma utilizada para treinar o desenvolvimento de algoritmos para resolver problemas de programação. No entanto, ela possui algumas limitações para que seja usada diretamente na sala de aula, pois este não consegue suprir algumas necessidades acadêmicas que uma disciplina requer, por exemplo, ter controle das respostas, acompanhar os alunos da turma, gerar planilha de notas, entre outras. Diante disso, o presente sistema tem como objetivo oferecer essas funcionalidades na criação e acompanhamento de listas com problemas de programação. A metodologia adotada no desenvolvimento do sistema foi baseada no scrum, além disso, para avaliar sua usabilidade, foi adotado o método baseado no Post-Study System Usability Questionnaire (PSSUQ). Os resultados obtidos na avaliação do sistema demonstraram bom grau de satisfação dos usuários que o testaram, de tal forma que os professores possam contar com o Ataux para auxiliar em uma disciplina de algoritmos.

## REPOSITÓRIO

https://github.com/thiaguin/Ataux

## 1 INTRODUÇÃO

Na computação, um algoritmo pode ser definido como sendo uma sequência de procedimentos lógicos bem definidos, com um número finito de passos, que tem o intuito de resolver determinado problema [2]. Um problema é definido, em termos gerais, por sua especificação, da relação entre um conjunto de valores de entrada e sua respectiva saída esperada. O algoritmo é o responsável por alcançar essa relação.

Para resolver problemas computacionais de forma eficiente, é de fundamental importância o estudo de técnicas de algoritmos. No curso de Ciência da Computação da Universidade Federal de Campina Grande, a disciplina de Análise e Técnicas de Algoritmos (ATAL) é responsável por apresentar e ensinar algumas técnicas famosas de algoritmos para resolver problemas de programação. Esse assunto é de fundamental importância na carreira de um bacharel nessa área, tendo em vista que várias empresas utilizam problemas que são resolvidos com o que é apresentado na disciplina, como parte da entrevista técnica na hora da contratação.

Sendo assim, é importante estudar e praticar o assunto abordado em ATAL de forma prática. Felizmente, existem várias plataformas para treinar e aprimorar a habilidade de desenvolver algoritmos, como o Codeforces, LeetCode, HackerRank, entre outras. Esses sites oferecem questões contextualizadas para simular problemas reais, que frequentemente são encontrados no desenvolvimento de sistemas computacionais. Essas plataformas disponibilizam desafios nos quais seus usuários escolhem, entre as possíveis opções, uma linguagem de programação para submeter suas soluções, obtendo uma resposta se o algoritmo enviado está correto ou não.

Diante disso, com o intuito de melhorar a didática da disciplina, um professor de ATAL quer organizar listas de questões para seus alunos, referente a cada técnica abordada em sala de aula, de modo que seja possível acompanhar as resoluções e a corretude do algoritmo enviado por eles. Se optar por utilizar o Codeforces [6], por exemplo, o professor precisa enviar o link das questões para os seus alunos, depois procurar manualmente na plataforma por cada aluno da turma, verificar as respostas que ele submeteu para cada questão da lista, com a finalidade de poder ver a resolução e o resultado da submissão.

Nesta perspectiva, observa-se o quão dificultoso é para o professor utilizar diretamente essas plataformas como objetivo de acompanhar as soluções que um aluno enviou para determinada questão. O que se torna ainda mais problemático na medida que em uma turma da disciplina existem vários alunos e várias técnicas para serem ensinadas.

Diante disso, uma ferramenta para acompanhamento de submissão de questões de programação, poderá propiciar diversos benefícios acadêmicos. Considerando que potencializa o aprendizado das técnicas vistas em sala de aula para os alunos, pois viabiliza o estudo de forma prática. Além disso, os professores poderão contar com uma ferramenta que irá auxiliá-los a ter um controle de como a turma está se saindo na disciplina.[1]

## 2 SOLUÇÃO

O presente projeto tem por objetivo desenvolver um sistema web para criação e acompanhamento de listas de questões de problemas de programação. Para, com isso, proporcionar um melhor controle aos administradores que gerenciam essas listas.

Sendo assim, o administrador poderá cadastrar uma turma, onde será possível criar listas de questões, que serão oriundas e submetidas através de plataformas onlines de concurso de programação. Os membros da turma serão vinculados ao sistema através do *handle* (nome identificador de usuário dessa plataforma).

Desse modo, os administradores poderão criar listas de exercícios com o conteúdo que acharem mais apropriado, acompanhar o resultado das submissões dos membros da turma, ver as soluções

destes, bem como gerar uma planilha com as notas dos integrantes da turma.

## 2.1 Funcionalidades

O sistema tem tipos diferentes de usuários e diferentes funcionalidades para cada tipo. Os tipos são: Administrador, Colaborador e Membro. Quando um novo usuário é cadastrado no sistema ele é definido por padrão como Membro. Um usuário administrador (já presente no banco de dados) poderá editar o tipo de qualquer outro usuário.

Os usuários do tipo Membro podem ver todas as questões dos sistemas, filtrando por dificuldade e técnica. Eles também podem se registrar em novas turmas e acessar as turmas que eles já são registrados, onde podem acessar as listas de questões daquela turma, para resolvê-las através do Codeforces, bem como sincronizar as questões, individualmente ou da lista completa, para obter os resultados das submissões. Além disso, ver os detalhes das submissões feitas por ele, como linguagem, resultado, código da resolução entre outros. As questões podem ser enviadas mesmo depois que exceder o prazo limite da lista, entretanto a nota será penalizada com base na quantidade de dias de atraso.

Os colaboradores e administradores podem acessar, criar, remover e editar questões, turmas, listas e tags, que são as técnicas ou palavras-chave referentes às questões, como também, podem adicionar ou remover tags referente à uma questão e adicionar ou remover questões de uma lista. Como também, sincronizar as questões de todos os membros, seja individualmente, por questão de uma lista, ou da lista completa. Além disso, gerar planilhas com as notas dos membros referente à uma lista ou a turma toda. Apenas os administradores podem editar ou remover um usuário.

# 3 ARQUITETURA

A arquitetura web adotada neste sistema foi o modelo cliente-servidor [1], no qual o cliente faz uma requisição solicitando determinado serviço e o servidor retorna uma resposta com os recursos solicitados. No Ataux, o sistema é separado em duas partes: o *frontend* da aplicação, que assume o papel do cliente, responsável por mostrar ao usuário as informações que são recebidas do servidor de forma amigável, e o *backend*, que assume o papel do servidor, responsável por aplicar as regras do negócio e persistir os dados. A comunicação entre os dois segue o modelo REST (*Representational State Transfer*).

## 3.1 Backend

Nesta seção serão descritas as principais decisões referente ao servidor da aplicação.

*3.1.1 Tecnologias do Backend.* O backend foi desenvolvido utilizando o NestJS [7], um framework para desenvolver aplicações NodeJS, do lado do servidor, de forma escalável e eficiente. Essas tecnologias foram escolhidas por serem de fácil configuração e manutenção, como também por serem amplamente utilizadas na comunidade, facilitando a pesquisa por soluções em fóruns e plataformas como o Stack Overflow. A linguagem de programação utilizada foi o Typescript, um superset Javascript, que utiliza tipagem estática.

Com o intuito de garantir a consistência dos dados, através de transações, e por ter bem definido as estruturas de dados que serão armazenadas, foi escolhido para persistir os dados o PostgreSQL[4], banco de dados relacional de código aberto. A comunicação do servidor com o banco de dados é feita através do TypeORM um ORM (object relational mapping), que é utilizado para fazer o mapeamento das tabelas em objetos conhecidos da linguagem escolhida, como também, para auxiliar na criação de *queries* ao banco de dados. O diagrama de entidade e relacionamento das entidades pode ser visto na figura 4. Para auxiliar o desenvolvimento e não precisar instalar e configurar o PostgreSQL localmente foi utilizado o Docker.

*3.1.2 Estrutura do Backend.* A estrutura de pastas padrão que o NestJS oferece foi adotada no backend, de modo que são separadas em 5 camadas principais: *Controllers, Modules, Middlewares, Services, Entities*. Com a finalidade de deixar o projeto coeso e leǵivel, essas camadas são agrupadas por componente, como pode ser visto na figura 5.

Os *Controllers* são responsáveis por tratar as requisições que são recebidas e retornar a resposta para o cliente. Essa camada apresenta um mecanismo de roteamento que identifica qual controller recebe a requisição para chamar as funções dos *Services* e repassar o retorno obtido. Os *Services* são responsáveis por realizar a lógica de manipulação de dados e aplicar a regra de negócio do sistema.

Os *Modules* são utilizados como forma de organizar os componentes do projeto. Existe o *module* raiz do projeto, que servirá como ponto de partida para construir o grafo de dependências utilizado no NestJS. Além disso, cada componente terá seu próprio *module* que é responsável por encapsular o conjunto dos seus recursos. É nessa camada que podem ser chamados os *Middlewares*, que são funções que são executadas antes de chegar no tratamento das rotas que é feita nos *controllers*. Eles têm acesso aos objetos de requisição e resposta da chamada, facilitando o tratamento das requisições antes de cada operação.

As *Entities* são responsáveis por definir e mapear uma tabela do banco de dados para um objeto que é conhecido pela linguagem de programação. Elas consistem basicamente em colunas e relacionamentos. Para auxiliar o desenvolvimento, na manipulação dos dados foi utilizado uma subcamada *Repository*. Além disso, a fim de facilitar a manipulação de dados internamente nos *services* foi utilizado o padrão DTO (*Data Transfer Object*).

## 3.2 Frontend

Nesta seção serão descritas as principais decisões sobre o lado do cliente da aplicação.

*3.2.1 Tecnologias do Frontend.* O *frontend* foi implementado utilizando o React [5], uma biblioteca JavaScript de código aberto para criar interfaces de usuário. Essa é uma tecnologia baseada em componentes, que torna o desenvolvimento do sistema bastante flexível e modularizado, proporcionando boa reutilização de código.

O Redux foi utilizado para manipular os estados da aplicação, esse centraliza todo o estado em uma árvore de objetos dentro do *store*, que pode ser acessado por todos os componentes. Através da interação com o usuário são disparadas *actions*, essas são observadas pelos *reducers*, que vão alterar o estado da aplicação. Diante disso, o React irá renderizar novamente os componentes que são afetados com a mudança. Esse comportamento é representado na figura 6. A linguagem utilizada no frontend, foi o JavaScript.

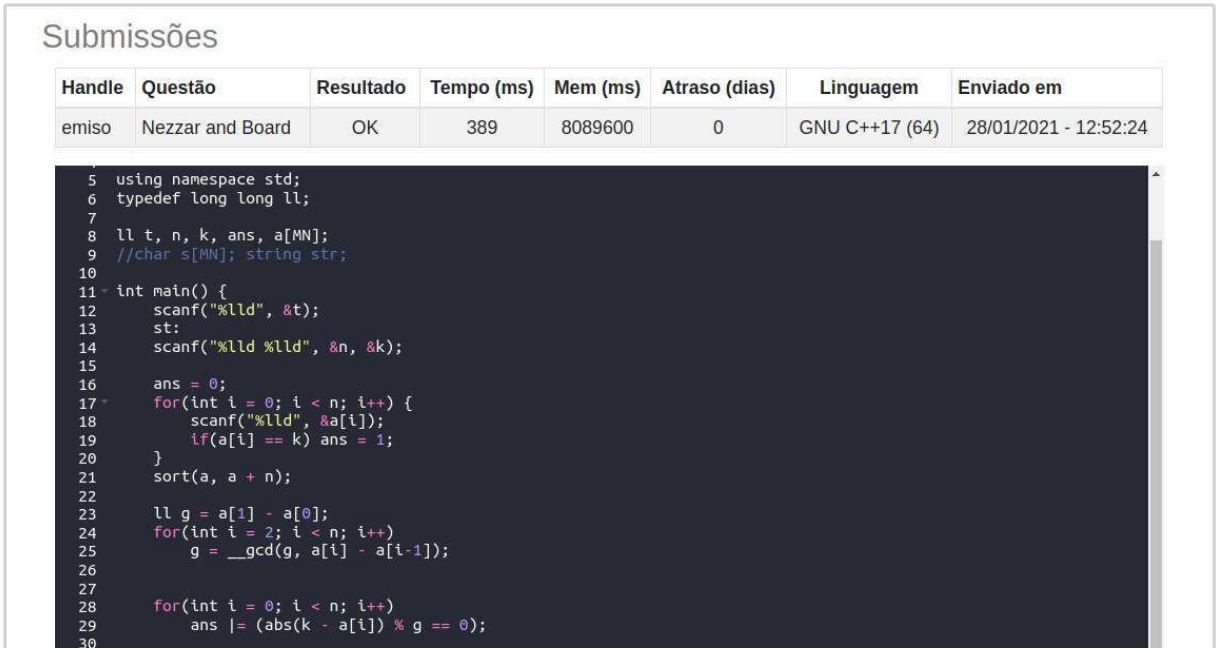


**Figura 1: Tela de detalhes de uma submissão.**

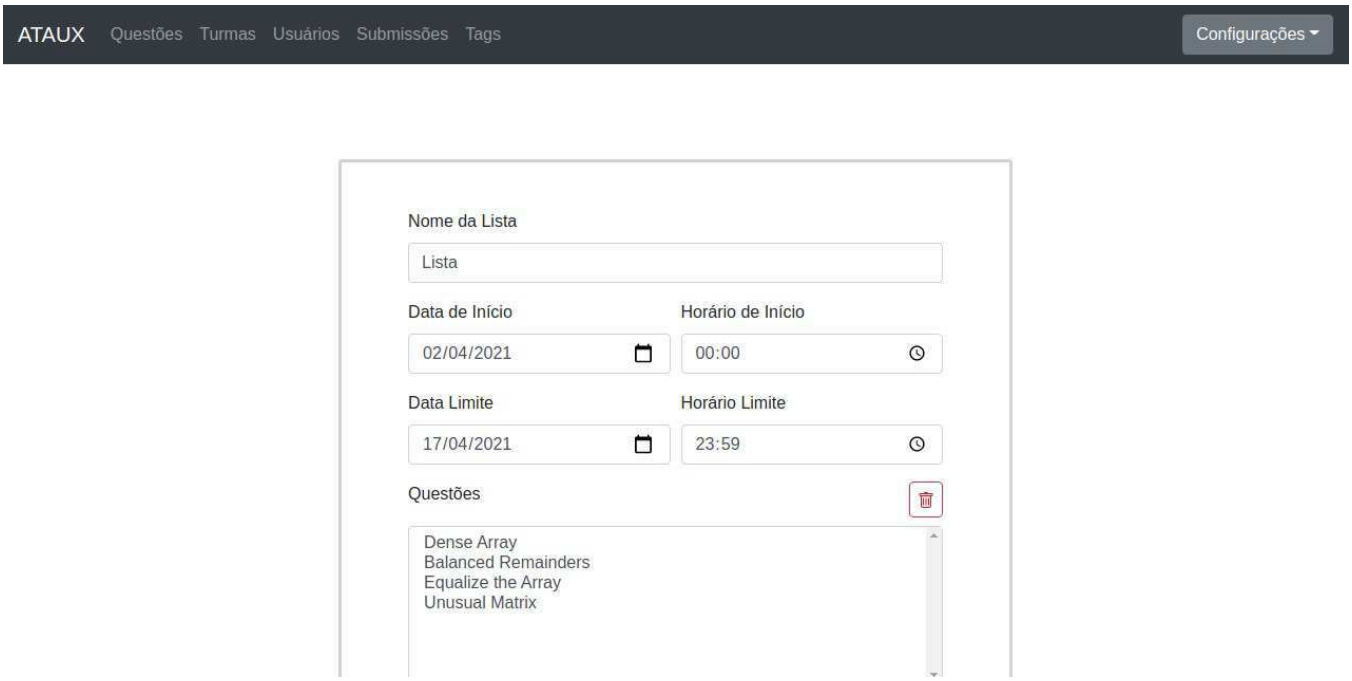


**Figura 2: Tela para criar uma lista de questões.**

O Bootstrap [10], um dos frameworks mais populares do mundo para construção de sites responsivos, é utilizado para tornar a interface do usuário mais agradável e facilitar o desenvolvimento. Nesse projeto, foi utilizado o React-Boostrap, que busca substituir o Bootstrap, mas com os componentes sendo construídos com React e sem dependências desnecessárias.

Para realizar a comunicação com o servidor foi utilizado o Axios, uma biblioteca de código aberto, que pode tanto ser utilizada no

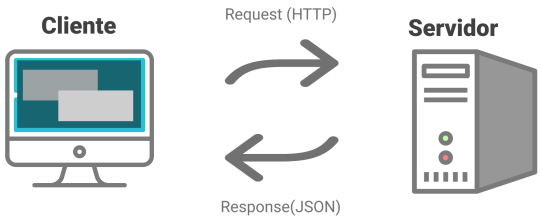


**Figura 3: Ilustração da comunicação entre cliente e servidor.**

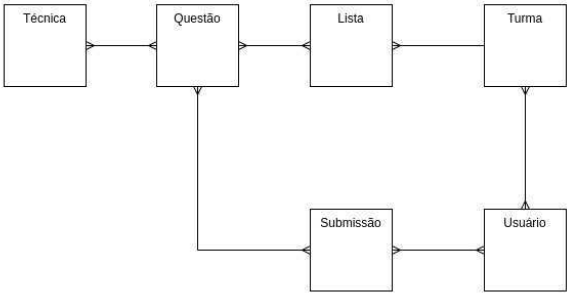


**Figura 4: Diagrama de entidades e relacionamentos do back-end.**

navegador quanto no NodeJS. Ela é um cliente HTTP (*Hypertext Transfer Protocol*) que consegue se comunicar com servidores REST através de requisições HTTP.

*3.2.2 Estrutura do Frontend.* Como pode ser visto na figura 7, a estrutura de dados do frontend foi dividido em 3 camadas principais: *pages*, *components* e *store*, e em 3 camadas auxiliares: os *assets*, que contém os arquivos estáticos e as imagens que são utilizadas no sistema, os *enums*, utilizado para centralizar as variáveis categóricas, e os *utils*, que serve para centralizar funções, as pastas de *enums* e *utils*, tem o intuito de centralizar código que é reutilizado em diversos lugares no sistema.

A pasta de *pages* contém todas as páginas da aplicação, onde cada uma é definida e atribuída a uma rota no arquivo App.js. Esta pasta, é responsável por se comunicar com o *store* e exibir a página web para o usuário com as informações necessárias. Com a finalidade de diminuir o acoplamento e reutilizar código é criado a pasta dos *components*.

O *store* é o que vai evoluir e manipular o estado da aplicação, esta pasta é divida em 2: *actions* e *reducers*. As *actions* são funções que podem receber entradas ou se comunicar com o servidor, através de requisições, para acionar mudanças no estado do *store*. Os *reducers* são funções puras que observam e esperam os resultados das actions para definir como será a mudança no estado da aplicação.


**Figura 5: Estrutura de pastas do backend.**

## 3.3 Autenticação

Para realizar a autenticação do sistema foi utilizado o JWT (*Json Web Token*) [3], um padrão definido para compartilhar de forma compacta e segura informações através de um objeto JSON. Através dessa biblioteca, é codificado um token com algumas informações, por exemplo, dados para a identificação do usuário e o seu tempo

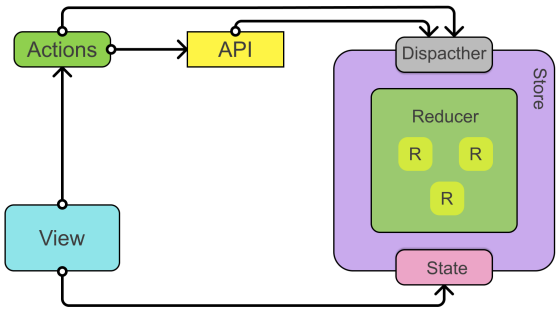


**Figura 6: Ilustração do fluxo de informação no cliente.**

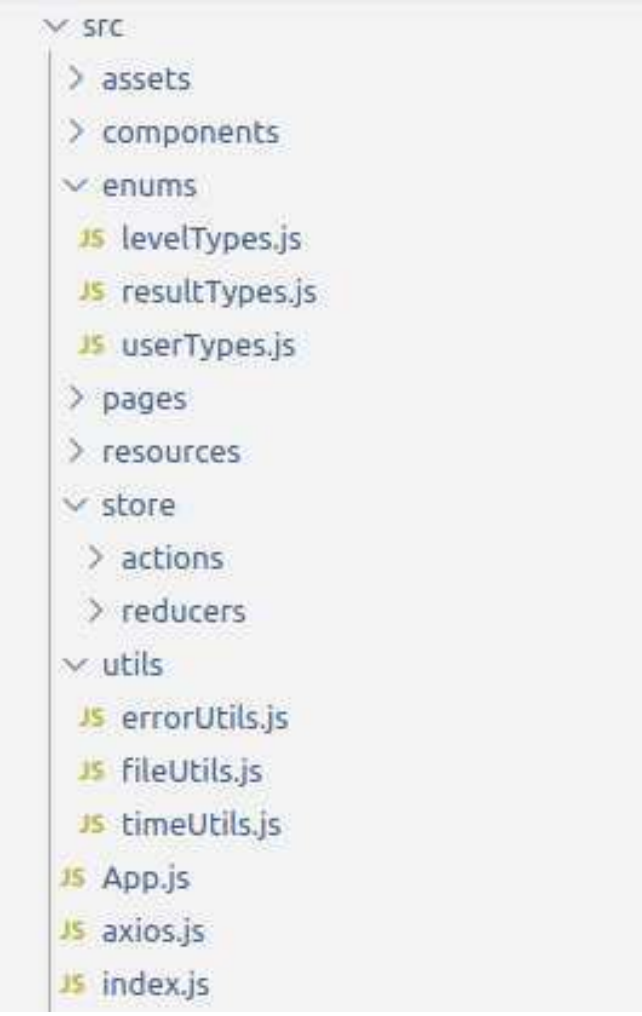


**Figura 7: Estrutura de pastas do frontend.**

limite de expiração. Essas informações podem ser decodificadas e verificadas através da biblioteca do JWT para o NodeJs.

O fluxo de autenticação segue da seguinte forma: o usuário, através do cliente da aplicação executa o login, que pode ser feito de duas maneiras: passando o email e a senha, ou através de uma conta Google, que irá gerar um token de acesso com informações do usuário que será decodificada na parte do servidor. O servidor recebendo as informações de login verifica se o usuário existe e valida as informações que foram recebidas, gerando assim um novo token e enviando como resposta. O cliente irá salvar o token recebido e enviar no campo *Authorization* dos *headers* das requisições que necessitam de autenticação. A autorização do sistema também é feita através do token que foi gerado, decodificando as informações do usuário e verificando se pela regra de negócio ele tem permissão para acessar determinado recurso.

## 4 AVALIAÇÃO

Nesta seção será descrito o ambiente de produção, a metodologia utilizada para a validação do sistema e os resultados obtidos.

### 4.1 Ambiente de Produção

Foi necessário colocar o sistema em um ambiente de produção para que os usuários pudessem testar e utilizar o Ataux. Para isso, foi utilizado o Heroku, uma plataforma de nuvem, que permite construir, integrar e monitorar aplicações. Por ser uma versão inicial de teste do sistema e por questões financeiras, essa plataforma foi escolhida por disponibilizar recursos gratuitos, porém, com algumas limitações. As principais limitações são: capacidade máxima do banco de dados de 1GB e que depois de algum tempo inativo, o sistema entra em estado de repouso, levando alguns segundos para ficar disponível novamente.

### 4.2 Metodologia de Avaliação

A avaliação do sistema foi feita utilizando como base o método *Post-Study System Usability Questionnaire* (PSSUQ) [9], uma maneira de avaliar o perfil de satisfação do usuário através de um questionário. Esse método foi desenvolvido pela IBM e fornece uma maneira de avaliar de forma geral a usabilidade do sistema. Atualmente, o PSSUQ está na terceira versão e consiste de alguns itens que abordam as seguintes características de um sistema: usabilidade, qualidade da informação e qualidade da interface. Além disso, a resposta para cada item está associada em uma escala de 1 a 7, onde os possíveis valores estão entre Concordo Fortemente e Discordo Fortemente.

O questionário utilizado não contém os mesmos itens que contém o PSSUQ, por causa do escopo e simplicidade do Ataux, foi feita uma adaptação do questionário. Sendo assim, foram adicionadas novas questões, removidas algumas e outras se fundiram para melhor se encaixar no contexto do sistema. As respostas para cada item seguem o padrão do PSSUQ normalmente.

Para responder o questionário de avaliação, a turma de ATAL do período letivo 2020.1 da UFCG, utilizou o sistema durante duas semanas, entre o dia 14 de abril até o dia 25 de abril. Nesse intervalo de tempo, foram lançadas duas listas de exercícios pelo monitor da disciplina e respondida pelos alunos que se cadastraram no sistema. A turma tem 71 alunos matriculados e 1 monitor, totalizando 72 usuários.

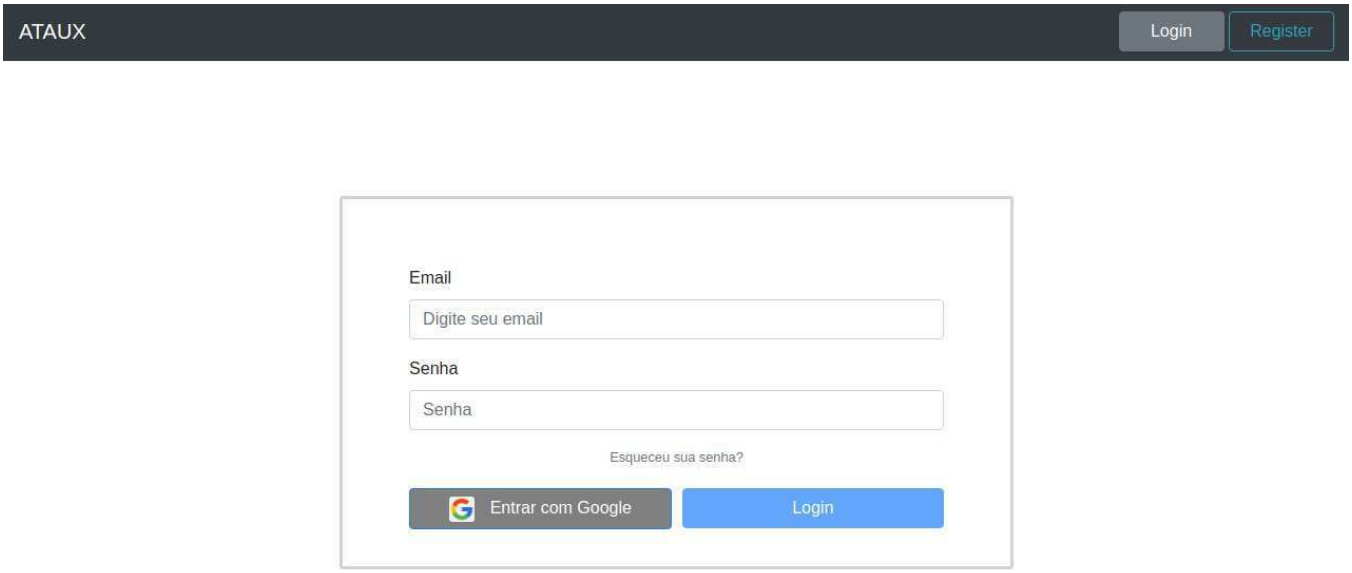


**Figura 8: Tela de login do sistema.**

O questionário (Apêndice - A) foi dividido em duas seções. Na primeira seção é avaliado fatores de usabilidade do sistema, bem como sua relação no contexto da disciplina, já na segunda seção foi realizado uma pergunta mais genérica, que pede para que os usuários darem uma nota ao sistema.

Após a utilização do sistema, os usuários de teste do sistema responderam o questionário para avaliar como foi a experiência. O questionário foi realizado utilizando o formulário online do Google Forms

## 4.3 Resultados

Para melhor análise dos resultado obtidos no questionário (Apêndice - A), a primeira seção, com as 11 primeiras questões, foram agrupadas para avaliar os seguintes aspectos do sistema (Tabela 1): satisfação geral do sistema (itens 1, 2, 10 e 11), qualidade da informação (itens 3 a 5), qualidade da interface (itens 6 e 7) e satisfação no contexto da disciplina (itens 8 e 9). Os resultados das questões podem ser vistos mais detalhadamente através do gráfico de barras na Figura 9.

**Table 1: Tabela da pontuação dos aspectos de usabilidade do sistema, em escala de 1 até 7**

| Aspectos do sistema | Pontuação |
|---|---|
| Satisfação geral do sistema | 6,06 |
| Qualidade da informação | 5,47 |
| Qualidade da interface | 5,46 |
| Satisfação no contexto da disciplina | 6,10 |

O formulário foi respondido por 42 pessoas, que representam 58,3% dos usuários que utilizaram o sistema na disciplina, onde destes, 41 são alunos e 1 monitor. Os resultados obtidos foram calculados através da média de valores de cada pergunta.

Os resultados apresentam valores próximos e consistentes, o valor que mais se distanciou dos outros está relacionado ao tratamento de erros, o que indica que é um ponto de melhoria. Todas as médias de valores foram superior à média da escala (4) adotada, além disso, 7 das 11 questões obtiveram nota maior ou igual a 6, muito próxima da nota máxima, o que representa um feedback bastante positivo do grau de satisfação dos usuários. Como pode ser visto em detalhes na Figura 10, a média da nota do sistema avaliado pelos usuários, em uma escala de 1 até 7, foi de 5.83.

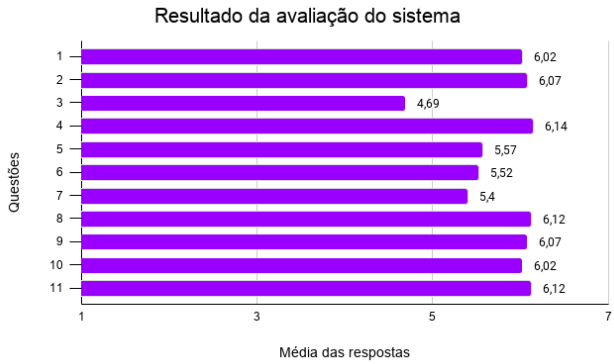


**Figura 9: Gráfico com os valores médios das respostas do questionário do Apêndice - A**


**Figura 10: Resposta do item 12 do questionário do Apêndice - A**

# 5 EXPERIÊNCIA

Nesta seção, será descrita a experiência de desenvolver uma aplicação de ponta a ponta até o *deploy*, focando no processo de desenvolvimento, nos principais desafios e limitações encontrados e nos trabalhos futuros.

## 5.1 Processo de desenvolvimento

Para a implementação do sistema foi adotado uma metodologia ágil que utiliza os princípios do Scrum. Esta metodologia é baseada em entregas rápidas de pequenas funcionalidades, que vão evoluindo de forma incremental e iterativa até atingir o produto final.

A ideia do Scrum [8] é composta por ciclos que são chamados de **Sprints**, intervalos curtos de tempo bem definidos, geralmente semanal, quinzenal ou mensal, onde as tarefas serão executadas. Todas as funcionalidades que foram desenvolvidas durante o projeto, tal como suas priorizações, foram definidas no **Product Backlog**. No **Sprint Backlog**, foi decidido quais tarefas do *Product Backlog* serão executadas em cada ciclo.

Sendo assim, para iniciar o desenvolvimento desse projeto, o *Product Backlog* foi definido com as funcionalidades do sistema e suas respectivas prioridades de implementação. As Sprints foram definidas com duração de 15 dias, de modo que a cada fim de ciclo fosse possível apresentar as novas funcionalidades implementadas. Refinando assim o *Sprint Backlog* para o próximo ciclo. Esse processo foi realizado até a finalização do projeto.

Inicialmente o desenvolvimento foi focado na parte do *backend*, com a implementação das principais funcionalidades e regras de negócio que foram definidas no *product backlog*. Após isso, iniciou-se a implementação do *frontend*, no qual foi criado a parte de interface do usuário, à medida que o desenvolvimento avançou, foi necessário melhorias no *backend*, isso aconteceu até obter o produto. Depois de finalizar esse processo, foi feito o *deploy* do sistema no Heroku, onde ele foi testado em um ambiente de produção, sendo assim observados possíveis melhorias que foram incrementadas.

## 5.2 Principais Desafios e Limitações

Um dos principais desafios foi criar uma interface amigável e agradável para o usuário, criar uma página web de forma responsiva é desafiador e trabalhoso. Para contornar isso foi utilizado o framework do React-Bootstrap. Outro desafio foi desenvolver o sistema em um tempo hábil, para que os usuários pudessem testar de forma tranquila durante o período determinado. A utilização de frameworks como o NestJS, ter um cronograma bem definido e a priorização das funcionalidades foi fundamental para superar isso.

A principal limitação está relacionada a consumir a aplicação do Codeforces, pois este tem uma limitação de 5 requisições por segundo, onde em alguns casos é preciso realizar várias chamadas nessa API, o que pode ocasionar lentidão no tempo de resposta ou ser retornado uma resposta de erro para o usuário. Além disso, o ambiente escolhido para o *deploy*, por ser a versão gratuita, tem limitações do tamanho do banco de dados, carga de desempenho que o sistema suporta e no tempo de resposta, caso o sistema entre em estado de repouso.

## 5.3 Trabalhos Futuros

O escopo atual do Ataux foi desenvolvido de forma simplista . No entanto, o sistema tem grande potencial de crescimento, sendo assim, podem ser adotadas em trabalhos futuros melhorias na interface do usuário, além disso, pode ser incorporado novas funcionalidades, tais como:

- Consumir questões de outras plataformas, por exemplo: HackerRank, LeetCode, URI, entre outros;
- Adicionar questões que poderão ser testadas na própria plataforma;
- Realizar as submissões diretamente da plataforma;
- Detecção de plágio.

# REFERÊNCIAS


[1] CanalTI. [n. d.]. Arquitetura cliente-servidor. Retrieved 2 de Abril de 2021 from https://www.canalti.com.br/arquitetura-de-computadores/arquitetura-cliente-servidor/#:~:text=Defini%C3%A7%C3%A3o,dos%20dados%20%28os%20clientes%29

[2] Thomas H. Cormen. 2009. *Introduction to Algorithms* (3rd. ed.). MIT Press, Massachusetts Londes, Inglaterra.

[3] Internet Engineering Task Force. 2010-2021. JWT. Retrieved 2 de Abril de 2021 from https://jwt.io/

[4] The PostgreSQL Global Development Group. 1996-2021. PostgreSQL. Retrieved 2 de Abril de 2021 from https://www.postgresql.org/

[5] Facebook Inc. 2017-2021. React. Retrieved 2 de Abril de 2021 from https://reactjs.org/

[6] Mikhail Mirzayanov. 2007. Guia do Codefores. Retrieved 2 de Abril de 2021 from https://codeforces.com/help

[7] Kamil Mysliwiec. 2017-2021. NestJS. Retrieved 2 de Abril de 2021 from https://nestjs.com/

[8] Andrew Stellma and Jill Alison Hart. 2014. *Learning Agile: Understanding Scrum, XP, Lean and Kanban.* MIT Press, Gravenstein Highway North, Sebastopol.

[9] Will T. [n. d.]. Post-Study System Usability Questionnaire (PSSUQ). Retrieved 2 de Abril de 2021 from https://uiuxtrend.com/pssuq-post-study-system-usability-questionnaire

[10] Bootstrap Team. 2011-2021. Bootstrap. Retrieved 2 de Abril de 2021 from https://getbootstrap.com/


# A QUESTIONÁRIO DE AVALIAÇÃO DO SISTEMA

Questões do questionário:

(1) Foi fácil usar esse sistema.
(2) Eu consegui entender como executar as funcionalidades do sistema rapidamente.
(3) O sistema mostrou mensagens de erro que me diziam como resolver os problemas claramente.
(4) O sistema sempre retornou uma resposta esperada.

(5) Foi fácil encontrar as informações que eu precisava para utilizar as funcionalidades do sistema.
(6) A interface me ajudou a entender como usar o sistema.
(7) A interface do sistema era agradável.
(8) O sistema supriu as funcionalidades esperadas para o escopo da disciplina.
(9) O sistema foi útil no contexto da disciplina.
(10) Estou satisfeito com o desempenho do sistema.
(11) No geral, eu estou satisfeito com esse sistema.
(12) Em termos gerais, numa escala de 1 até 7, onde 1 é a nota mínima e 7 é a nota máxima, qual nota você daria para o sistema?

As respostas para essas questões começam com 1 (Discordo totalmente) e vai até 7 (Concordo totalmente), onde quanto mais alta a pontuação média, melhor é satisfação dos usuários com o sistema. Essas perguntas buscam representar uma a satisfação do usuário quanto ao sistema no geral, satisfação do sistema relacionado ao contexto da disciplina, utilidade do sistema, qualidade da informação, qualidade da interface do sistema.