Universidade Federal do Estado do Rio de Janeiro
Centro de Ciências Exatas e Tecnologia
Escola de Informática Aplicada

CoStylist: Uma rede social para recomendação coletiva de estilo

Kevin Crystian Botelho da Costa

Orientador
Márcio de Oliveira Barros

Rio de Janeiro, RJ – Brasil
Novembro de 2024

Catalogação informatizada pelo autor

C838 Costa, Kevin Crystian Botelho da CoStylist: Uma rede social para recomendação coletiva de estilo / Kevin Crystian Botelho da Costa. -- Rio de Janeiro : UNIRIO, 2024.
60

Orientadora: Márcio de Oliveira Barros.
Trabalho de Conclusão de Curso (Graduação) Universidade Federal do Estado do Rio de Janeiro, Graduação em Sistemas de Informação, 2024.

1. Aplicação Web. 2. Rede Social. 3. Recomendação de Estilo. I. de Oliveira Barros, Márcio, orient. II. Título.

CoStylist: Uma rede social para recomendação coletiva de estilo

Kevin Crystian Botelho da Costa

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Aprovado por:

_____

Márcio de Oliveira Barros (UNIRIO)

_____

Paulo Sérgio Medeiros dos Santos (UNIRIO)

_____

José Ricardo Cereja (UNIRIO)

Rio de Janeiro, RJ – Brasil.

Novembro de 2024

Agradecimentos

Primeiramente, gostaria de agradecer aos meus pais, Darlei Duarte da Costa e Joana Darc Botelho da Costa, por terem incentivado a mim e aos meus irmãos, a sempre trilharmos o caminho da educação. Deste modo sempre fui ensinado por eles a ter a obtenção do diploma de nível superior como objetivo, mesmo que eles mesmos não tenham tido essa oportunidade.

Também agradeço a todos os meus colegas de curso, que por muitas vezes me auxiliaram na realização das tarefas e em momentos de estudos. Em especial destaco: William Machado, Renan Hatherly, Luca Loyolla e Victor Oliveira.

Agradeço também a meus colegas que cursam graduação em outras áreas, porém, mesmo em muitos momentos ouviram minhas divagações sobre o desenvolvimento desse trabalho e contribuíram para o mesmo. Além disso, estes contribuíram para minha formação como um todo, sendo possível destacar: Lara Dorileo, Tacio Monteiro, Leonardo Amorim, Nayara Guedes, Raira Lima.

Além dos citados anteriormente, agradeço a meus colegas de trabalho, que me auxiliaram, ouvindo sobre o tema deste trabalho, e inclusive, dando sugestões que foram muito úteis para elaboração do mesmo, se destacando: Rodolpho Lipovscek, Diordi Yamada, Gabriela Mendes e Gabriel Souza.

Finalizo agradecendo a todos os professores que me auxiliaram nessa jornada, bem como meu orientador Márcio Barros, que me deu todo apoio necessário para a conclusão deste trabalho. Não obstante, também agradeço a todos os funcionários que trabalham diariamente pelo bom funcionamento da Universidade Federal do Estado do Rio de Janeiro.

RESUMO

A afirmação do indivíduo por meio do estilo é algo presente na sociedade a décadas e se intensificou com o advento da *internet*. As subculturas, agora com alcance global, se utilizam dessa lógica como vínculo e forma de distinção. Mediante a este cenário, este trabalho visa o desenvolvimento de uma plataforma *web*, na forma de uma rede social, que sirva como ferramenta na busca de referências na construção do seu estilo.

Esta plataforma, possui um *frontend* responsivo utilizando o *framework React.js*. Este *frontend* se conecta a um *backend* construído com o framework *Nest.js*. Além disso, os dados são armazenados em um banco de dados PostgreSQL, e todas as imagens e textos passam por validação através de integração com o *Gemini API*.

Palavras-chave: estilo, rede social, reactjs, nodejs, plataforma *web*

ABSTRACT

The individual's assertion through style has been a part of society for decades and has intensified with the advent of the internet. Subcultures, now with a global reach, use this approach as a bond and form of distinction. Given this scenario, this work aims to develop a web platform, in the form of a social network, which serves as a tool for seeking references in constructing one’s style.

This platform features a responsive frontend using the React.js framework. The frontend connects to a backend built with the Nest.js framework. Additionally, data is stored in a PostgreSQL database, and all images and texts are validated through integration with the Gemini API.

Keywords: style, social network, reactjs, nodejs, web platform.

Índice

Índice de Figuras

1 Introdução

1.1 Motivação

A principal motivação para a criação dessa plataforma é o entendimento da importância do estilo como afirmação do indivíduo. Ao mesmo tempo, o estilo também serve como elemento de identificação entre membros de grupos sociais e subculturas, ajudando a criar um sentimento de pertencimento a estes grupos. Essa afirmação individual foi algo que me ajudou a criar uma identidade pessoal e melhorar minha autoestima e interação social.

A importância do estilo de vestimentas para subculturas também se torna relevante, além do senso de pertencimento. Esses grupos procuram se distanciar da cultura, da mesma forma que o estilo muitas das vezes se distancia da moda. Portanto, muitas vezes, esses coletivos e a moda andam juntos.

Em minha busca por autoafirmação através do estilo, percebi que me identificava com um tipo muito específico de roupa, muitas vezes usado por subculturas urbanas, como *Streetwear* ou *Skatewear*, que são culturas ligadas a movimentos marginalizados, como *hip-hop*, *skate* e grafite.

Achar roupas que se adequassem a essa categoria em *sites* de compra convencional se provou algo difícil. A alternativa que muitas vezes acaba sendo adotada por pessoas do meu círculo social é utilizar a rede social *Pinterest* para achar referências, buscando por palavras-chave. Apesar disso, esta rede não é voltada somente a roupas, não possui uma classificação inerente ao tema e nem sempre possui *links* para compra dos itens apresentados.

Outra alternativa seriam as ferramentas de recomendação por categorias, como o proposto por SHIMIZU (2023), onde as roupas são classificadas por etiquetas (*tags*) definidas através de *Machine Learning*. Essa alternativa, porém, não contempla o aspecto de rede social, onde é possível se espelhar em outros usuários e também acompanhar a mutabilidade dessa categoria.

Tendo em vista essas opções, a melhor alternativa que se demonstrou foi de desenvolver uma rede social para facilitar a busca de referências de roupa seguindo um estilo específico. Além de ser uma forma de busca, também foi notado os benefícios de orientar os filtros pela lógica de subcultura, fortalecendo assim o senso de pertencimento e identidade presente na organização desses grupos.

Dessa forma, a ideia do Costylist foi nutrida por dois anos, contemplando visões de outras plataformas e estudos acerca dos temas relacionados. Por fim, esse trabalho se tornou o local oportuno para o desenvolvimento de uma plataforma com essas características, bem como a fundamentação dos possíveis impactos sociais de uma rede com esse aspecto.

1.2 Objetivos

O objetivo deste trabalho é o desenvolvimento de uma aplicação *web* com a estrutura de rede social. Esta rede permite que usuários exponham peças de roupa que sigam um determinado alinhamento de estilo e acompanhem *publicações* de outros usuários com estilos similares aos seus. Desta forma, se torna possível a construção de estilos através da interação coletiva dos usuários, além da construção da categorização em subculturas por meio das postagens dos usuários.

Essa categorização permite que, além de ferramenta de busca, essa plataforma se torne uma ferramenta que possa facilitar a identificação através de grupos. Acreditamos que alguns recursos disponíveis no projeto, como a possibilidade de seguir outros usuários e de comentar nas postagens que estão presentes na aplicação, serão de grande utilidade para alcançar esse objetivo.

Outro aspecto relevante visado para este projeto é o fato de ser uma construção coletiva de categorização, com os próprios usuários definindo em qual categoria a sua postagem se enquadra no momento da criação. Esse aspecto se torna algo relevante e coerente devido à volatilidade das categorias, que são definições abstratas construídas pela sociedade.

1.3 Organização do texto

O presente trabalho está estruturado em capítulos e, além desta introdução, será desenvolvido da seguinte forma:

- Capítulo II: apresenta o cenário atual com outras soluções usadas na busca por referências de vestuário.
- Capítulo III: percorre pelas tecnologias utilizadas no desenvolvimento da aplicação, bem como os casos de uso e diagrama de classes que orientaram o desenvolvimento.
- Capítulo IV: descreve e apresenta visualmente o funcionamento da aplicação através das telas que compõem a interface de interação com o usuário da mesma.
- Capítulo V: finaliza o texto, com uma breve análise do trabalho como um todo, contendo as limitações encontradas e os possíveis trabalhos futuros.

2 Contextualização

Primeiramente, para entender o aplicativo proposto, é necessário diferenciar moda e estilo e entender ambos os conceitos. Segundo a estilista Titta Aguiar:

> "A moda é o reflexo da cultura do momento, são as tendências difundidas pelos estilistas em todo o mundo. O estilo é a expressão pessoal de quem somos, a expressão do caráter, a relação com o mundo à nossa volta; tem conteúdo pessoal, pois, tiramos da moda somente aquilo com o que nos identificamos. " (AGUIAR, 2004 p. 39 - 40).

Ou seja, a moda reflete a cultura vigente, alinhada com as normas sociais daquele local e época, enquanto o estilo é algo que permite uma expressão maior. Em muitos casos, inclusive, ele vai contra a moda ou acaba absorvendo aspectos pontuais dela.

Outra definição cujo entendimento se faz necessário, é a de subcultura. Como descrito no Dicionário de Cambridge (TURNER, 2006), o termo se refere a qualquer grupo que, dentro de uma cultura local, possui valores e comportamentos distintos. De maneira mais aprofundada, o termo representa uma rejeição simbólica da ordem vigente e da cultura local, se tornando uma forma de se destacar por oposição e se unir a outros que se rebelam da mesma maneira.

A moda reflete a cultura vigente e, nesse contexto, os membros de subculturas, ao se oporem à ordem estabelecida, criam uma distinção clara ao adotarem estilos pessoais. Dessa forma, esses estilos se tornam uma das principais formas de expressão utilizadas por diversos grupos para desafiar o consenso estético vigente na sociedade.

Os estilos pessoais, apesar de se diferenciarem da moda vigente, podem se aproximar entre si dentro de um mesmo grupo. Esse tipo de assimilação acaba criando subculturas que são ligadas através de conceitos e símbolos a serem representados. Um comentário pertinente acerca desse tópico é da historiadora Mara Rúbia Sant'Anna:

> “As roupas, por serem signos que carregam em si uma série de significados atrelados à beleza, à juventude, à feminilidade ou masculinidade, à riqueza e distinção social ou à marginalidade, à alegria ou tristeza etc., imprimem ao seu portador uma escolha diária de posicionamento no conjunto maior das teias de significados compostos como cultura. A apropriação desse signo permite desde a exaltação dele próprio à sua contestação irônica. ” (SANT’ANNA, 2009 p.76)

Ou seja, as roupas são elementos que carregam signos e diversas subculturas adotam as vestimentas para passarem suas ideias. É possível concluir que por meio desses signos podemos agrupar determinados tipos de roupa, que são adotados por algumas subculturas. Um exemplo dessa aproximação dentro de um mesmo grupo, como descrito por CAETANO (2020), pode ser encontrado nos góticos. O grupo conta com um forte e distinto apelo visual que se manifesta por meio das roupas. Tal importância da vestimenta na construção de uma identidade torna a busca por peças de roupas específicas algo necessário.

Estilos pessoais e subculturas estão se tornando tópicos em crescimento, principalmente entre os jovens. O que já era algo presente na sociedade com grupos como os *hippies* nos anos 60 e os *punks* dos anos 70, se tornou algo mais plural e global com o advento da Internet. Diversos novos movimentos similares são criados e popularizados, com diversas pessoas adotando suas representações.

Essa pluralidade advinda da imersão digital, como explicada por OLIVEIRA (2012), se dá muitas vezes a partir da interação em ambientes virtuais, chamados de ciberespaço. Porém, essa dinâmica e influência acelerada, particular das redes, não se limita ao *online*. Ela também perpassa os ambientes urbanos e a vida cotidiana nos centros, com esses estilos únicos estando presente em todos os ambientes.

A influência comportamental chega através de redes sociais como o *Tiktok*, influenciadores no Instagram ou no *Youtube*, por amigos ou parentes. Apesar da forma fácil que essa influência é propagada, obter novas lojas de vestimentas e referências diferentes às vezes se torna algo complicado. Por isso, encontrar vestimentas relacionadas a um estilo específico se tornou algo presente na vida de muitos jovens, ainda que não seja tarefa simples.

A opção escolhida para a adequação de vestiário varia bastante: alguns escolhem se focar em lojas ou marcas específicas ao nicho que fazem parte, enquanto outros usam

plataformas online para se inspirarem. É justamente nesse espaço que a rede social proposta neste trabalho pretende atuar e plataformas como Pinterest e, em menor escala, o *Whering* são atualmente utilizadas. Além destes, também há propostas de ferramentas de busca utilizando *Machine Learning* para categorização das postagens realizadas pelos usuários.

2.1 **Pinterest**

O Pinterest é uma das plataformas de rede social mais populares, focada em possibilitar através de imagens que o usuário consiga buscar novas ideias e inspirações sobre diversos tópicos. Ele é usado para diferentes fins, como receitas, decoração de ambientes, reformas e também para busca de novas referências de vestuário.

O foco principal do Pinterest é possibilitar a descoberta de novas referências através dos *Pins*, como são chamados os *posts*, e também de salvar essas referências para serem consultadas novamente. Por esse motivo se escolheu o termo *Pin*, que significa alfinete, para representar suas publicações. Porém, ele também possui características de rede social, possibilitando o compartilhamento e a adição de comentários.

Existem diferentes tipos de *Pins*, destacando-se os de imagens, de vídeos, os avançados e os de produto. Os *pins* de imagem e vídeo são simples de entender e se referem somente ao tipo de mídia presente. Já os avançados, por sua vez, não possuem um arquivo de mídia atrelado: ao invés disso, assimilam informações de um site externo, como artigos, receitas e produtos, e contém *links* para esses sites.

Se aprofundando nos *posts* de produtos, eles possibilitam a adição de imagens do item a ser vendido, descrição textual, *hashtags* relevantes e o *link* para a loja. Esse tipo de publicação tem se tornado uma opção para se encontrar novas peças de roupas a serem adquiridas. A chegada desses *pins* ao usuário pode ser por meio do algoritmo que disponibiliza em seu *feed* ou por meio de uma busca ativa na plataforma, utilizando termos ou *hashtags*. A Figura 1 apresenta um exemplo de *post* de produto.

Figura 1 – Exemplo de Post de produto no *Pinterest*, com foto, descrição e link para compra do produto.

A página inicial, também chamada de *feed*, exibe diversas recomendações de publicações obtidas através de algoritmos de recomendação com base nas pastas criadas, dos *pins* interagidos e das buscas realizadas. Ela fornece uma forma simples de obter contato com novas postagens, mas por ser uma plataforma de muitas aplicações, pode conter vários elementos diversos. A Figura 2 apresenta um exemplo de página inicial de um usuário hipotético.

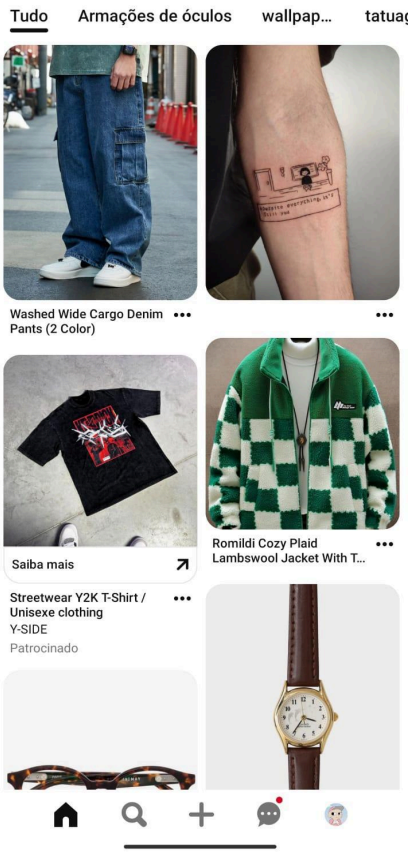

Figura 2 – Exemplo de *feed* de um usuário no *Pinterest*.

Outra forma de contato com novas postagens é a barra de busca, onde, ao clicar no ícone de busca, abre-se a barra superior que permite pesquisar por palavra chave. Essa é uma forma mais direcionada, que possibilita encontrar publicações relacionadas ao termo inserido. No entanto, dada a natureza da rede, nem todos os resultados de uma busca vão possuir referência daquela palavra específica. A Figura 3 apresenta a tela de busca do *Pinterest*.

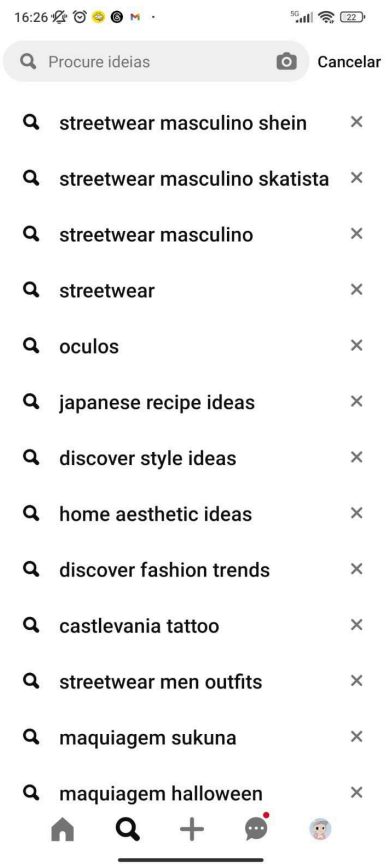

Figura 3 – Exemplo de busca no *Pinterest*.

Tendo em vista a estrutura do Pinterest, é possível entender sua popularidade, devido a grande gama de possibilidades e facilidade de seu uso. Porém, o aplicativo proposto neste trabalho se diferencia do *Pinterest*. Isso se deve ao fato do CoStylist orientar todas as publicações em relação a termos específicos de subculturas no momento da sua criação, focando em postagens com *links* de compra e focado em itens de vestuário.

Outro ponto de comparação são os elementos de rede social que, assim como no *Pinterest,* estão presentes no CoStylist. O ponto de divergência, porém, é de que na primeira, as postagens advindas de quem o usuário segue são misturadas às demais, enquanto na segunda, são separadas por abas específicas.

2.2 **Whering**

Whering é uma aplicação que vem começando a ganhar espaço recentemente. Apesar de possuir um número de usuários total menor que o *Pinterest*, ele vem ganhando força e sendo comentado dentro do nicho da moda.

A proposta do aplicativo é se tornar um guarda-roupa virtual, possibilitando que usuários tirem fotos de suas roupas e o adicionem no seu perfil, podendo combinar os itens de modo a criar *outfits* por integração com o *Canvas*. Também é possível criar uma lista de desejo de itens que não possui, mas pretende adquirir.

Além disso, é possível visitar outros usuários, e além de seguir esse perfil, é permitido visualizar tanto itens quanto *outfits*, e adicionar estes itens à sua própria conta. É possível visualizar os itens filtrando pelo seu tipo, além de visualizar os itens que o perfil visitado possui na lista de desejos. A Figura 4 apresenta a tela de uma usuária do *Whering*.

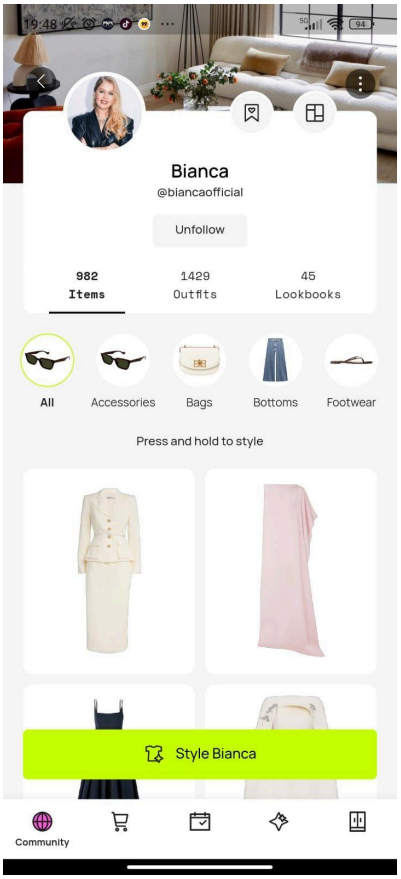

Figura 4 – Exemplo de página de usuário no *Whering*.

Outro recurso é a aba de comunidade, único setor de busca e interação com novas ideias. Nesta seção, são exibidas sugestões de perfil, mas o que fica em evidência é a barra de busca, onde somente é possível buscar nomes de usuários, sendo, portanto, uma busca direcionada a perfis e não a roupa em si. A Figura 5 apresenta uma página de comunidade.

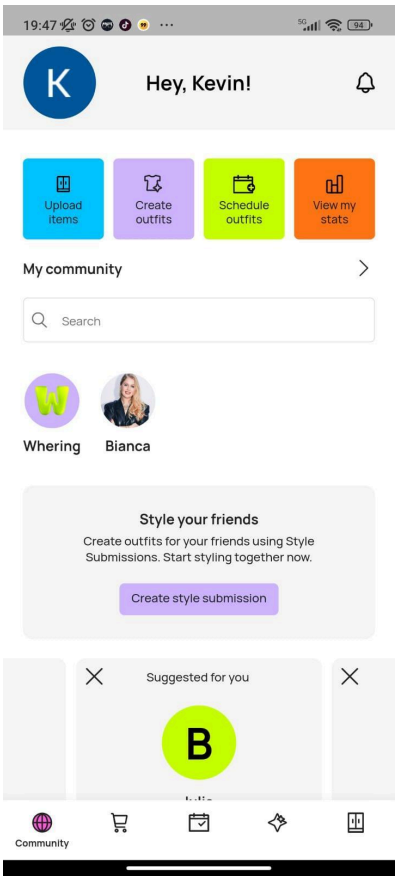

Figura 5 – Exemplo de página de comunidade no *Whering*.

O último recurso de interesse presente na plataforma é a aba de *marketplace*, onde é possível encontrar e buscar roupas disponíveis para compra em plataformas específicas. Porém, o aspecto dessa parte do aplicativo se assemelha a um site de compra convencional, com os produtos não sendo relacionados a nenhuma postagem ou usuário e possuindo um sistema de busca simples e pouco refinado.

Outro ponto notável é que é possível adicionar as roupas dessa sessão ao guarda-roupas. Porém, a roupa salva não carrega a referência para a parte de compra ou para a loja, perdendo assim parte da experiência do usuário. Também não é possível adicionar novos itens à loja e a quantidade de lojas também parece ser bem reduzida, somente abrangendo lojas muito específicas. A Figura 6 apresenta a página de *marketplace* do *Whering*.

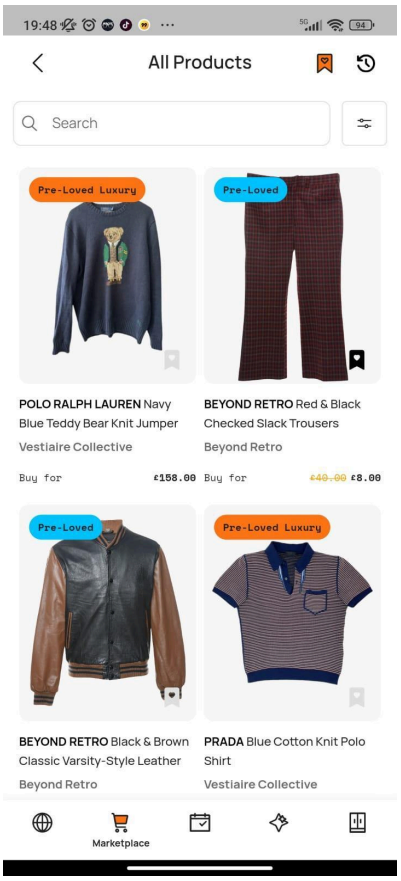

Figura 6 – Exemplo de página de *marketplace* no *Whering*.

Acreditamos que essa plataforma, apesar de ter funcionalidades inovadoras, têm características que fazem o CoStylist se tornar uma melhor opção no que se refere a auxílio em compras. A falta de uma referência direta nos itens a lojas de compra específicas, ou de uma ligação do setor de compras com a parte de perfis, torna a experiência de comprar um item visto em um perfil algo trabalhoso para o usuário.

Outra característica que evidencia essa diferença é o setor de comunidade ser a única forma de ter contato com outras idlevin13éias. Isso se torna um fator limitante, porque somente é possível visualizar e buscar outros usuários, e não por palavras-chave, categorias, ou mesmo visualizar novas roupas e itens que talvez possam interessar.

Além desse aspecto, a ausência de uma sessão de comentários é algo que se torna notório e que talvez pudesse ajudar na interação entre os usuários e na formação de um sentimento de comunidade. Também não é possível adicionar aos itens ou *outfits* nenhum tipo de palavra-chave ou *hashtag*, o que auxiliaria na busca.

2.3 **Machine Learning**

Existem artigos publicados onde é proposta a elaboração de aplicações que auxiliam na compra por categorização de itens. Dentre estes, no presente trabalho, foi escolhido o artigo de autoria de Ryotaro Shimizu intitulado “Fashion intelligence system: An outfit interpretation utilizing images and rich abstract tags” SHIMIZU (2023).

No artigo, o autor propõe um modelo que automatiza a interpretação de imagens e as categoriza a partir de *tags* abstratas, facilitando uma busca posterior. O objetivo do artigo se foca na aplicação desse modelo a ser utilizado para auxílio de compras a peças de roupas e vestuários. A aplicação proposta utiliza uma tecnologia chamada *Single Shot MultiBox Detector* para identificação de objetos e de suas devidas características. Após isso, através de *machine learning* ele relaciona essas características observadas a *tags* pré-definidas.

É possível se observar através dos dados obtidos no trabalho, que de fato a plataforma proposta consegue diminuir a ambiguidade em relação à categorização no mundo da moda. Além disso, o autor também destaca a aplicação da proposta em estratégias de *marketing* e em buscas feitas por usuários para compras.

Apesar de se mostrar eficiente e ter uma aplicação clara para auxílio na comercialização de itens de vestuário, o trabalho de Ryotaro se diferencia do CoStylist. Os pontos de maior relevância estão nos aspectos de rede social que essa proposta não possui. O CoStylist se baseia em um aspecto forte de rede social, com a categorização sendo criada a partir da atribuição dos próprios usuários. Acreditamos que esse aspecto permite criar categorizações mais dinâmicas e que se adequem melhor ao nível de abstração das categorias.

Além disso, ser uma rede social permite que haja interação entre os usuários, podendo seguir uns aos outros e se espelharem em postagem para que haja novas compras. Outro aspecto a ser mencionado é a parte de adição de comentários nas publicações, o que cria maior engajamento do usuário com a plataforma, fazendo ele passar mais tempo no aplicativo e ter contato com mais produtos.

Outra parte relevante é de que o CoStylist permite a criação de uma base autoalimentada de postagens contendo peças de vestuário disponíveis para compra. Essa possibilidade não é presente em uma ferramenta de busca comum, que somente utilize

outra fonte de dados para realizar as buscas e possibilitar a compra, como diretamente os *sites* de compra.

Por fim, um aspecto valioso de se criar uma rede social com essa finalidade é a possibilidade de se manter o histórico de gostos e preferências dos usuários. Essa informação pode ser valiosa e possibilitar recomendações e anúncios personalizados.

2.4 Considerações Finais

Tendo em vista essas outras propostas que recorrem a uma área similar de atuação, ainda se evidencia a necessidade da criação de uma rede social, na forma de uma aplicação web, que permite a busca, compra e categorização de roupas através de categorias pré-definidas. No próximo capítulo, as características desta aplicação serão detalhadas e uma solução técnica será proposta.

3 Desenvolvimento

O início do desenvolvimento se deu com a identificação dos casos de uso que orientaram as funcionalidades da aplicação projetada. Após esse momento, foi elaborado um diagrama de classes para modelagem dos objetos retratados, que auxiliou na criação de classes, interfaces, DTOS e do banco de dados. Outro ponto relevante foi a definição das tecnologias a serem utilizadas. Esses três pontos serão abordados neste capítulo.

3.1. Diagrama de classes

O diagrama abaixo é um modelo UML projetado para orientar o desenvolvimento do Costylist.

Ele possui seis classes, onde *User* representa o usuário, logo após a criação de conta na plataforma, tendo passado todos os dados necessários. Por sua vez, *Follow* representa a interação de seguir entre dois usuários, registrando o id do usuário que começou a seguir outro e o id do usuário que está sendo seguido.

Post representa as postagens criadas pelos usuários, contendo todas as informações relevantes necessárias para essa representação. Uma das informações presente em *Post* é o estilo no qual aquela peça de vestiário se relaciona. Todos os estilos presentes na plataforma estão representados em *Styles.*

Outra representação presente que se relaciona com post são *Like* e *Comment,* que representam respectivamente, curtidas e comentários em uma publicação. Ambos contêm o id da publicação e do usuário que realizou a ação, com a diferença de que um usuário só pode curtir uma vez a mesma publicação, mas pode comentar diversas vezes, além de em *Comment* ser necessário o conteúdo da mensagem.

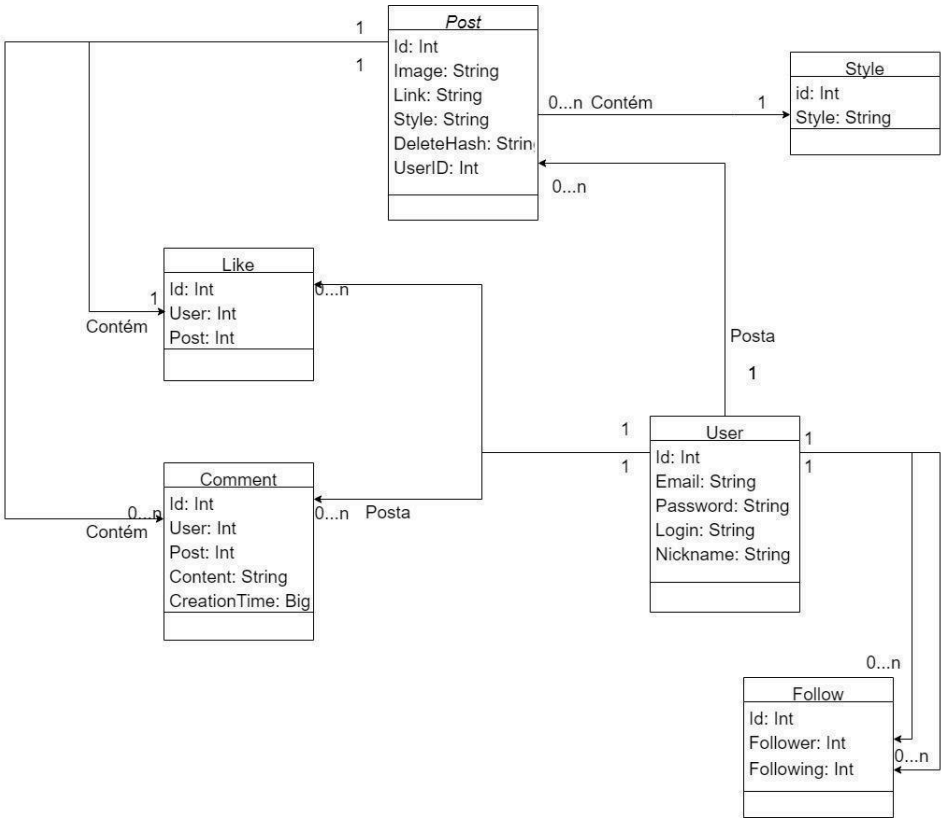

Figura 7 – Diagrama de classes do Costylist.

3.2. Casos de usos

Inicialmente foram identificados nove casos de uso que orientaram o desenvolvimento, porém, ao longo da execução do projeto, surgiu a possibilidade de adição de novos casos de uso, totalizando dezessete. Tanto os casos de uso iniciais, quanto os identificados posteriormente estão listados abaixo sem distinção.

3.2.1. Caso de uso: Inserir postagem

Descrição geral: Permitir que o usuário consiga adicionar novas postagens.
Atores: Usuário.
Pré-condições: Usuário está logado, e na sessão de closet.
Pós-condições: Usuário visualiza a nova postagem em seu closet.
Fluxo principal:

1. O usuário clica em adicionar.
2. O sistema exibirá um modal de título adicionar postagem contendo input de imagem, *select* para seleção do estilo e input de texto para inserção de link.
3. O usuário preenche todos os dados e clica em seguir.

4. O sistema exibe um *spinner* de carregamento e ao final apresenta a postagem.

Fluxo alternativo:

A.1. O usuário insere uma imagem que é inadequada para a rede social.

A.1.2. O sistema exibe o erro “foto não permitida”.

3.2.2. Caso de uso: Visualizar postagens

Descrição geral: Permitir que o usuário visualize suas postagens.

Pré-condições: Usuário estar logado.

Pós-condições: Usuário com visão dos *posts* adicionados, podendo editar ou excluir postagens.

Fluxo principal:

1. O usuário clica em “Meu Closet”.

2. O sistema redireciona o usuário a uma página onde poderá visualizar em forma de *cards* as postagens correspondentes a esse usuário.

3.2.3. Caso de uso: Atualizar postagem

Descrição geral: Permitir que o usuário consiga editar as postagens adicionadas em seu closet.

Pré-condições: Usuário estar logado, e na sessão de closet.

Pós-condições: Usuário visualiza a postagem editada.

Fluxo principal:

1. O usuário clica em editar na postagem que deseja editar.

2. O sistema exibirá um modal de título atualizar postagem contendo input de imagem, *select* para seleção do estilo e input de texto para inserção de link.

3. O usuário preenche todos os dados e clica em seguir.

4. O sistema exibe um *spinner* de carregamento e ao final apresenta a postagem editada.

Fluxo alternativo:

A.1. O usuário insere uma imagem que é inadequada para a rede social.

A.1.2. O sistema exibe o erro “foto não permitida”.

3.2.4. Caso de uso: Excluir postagem

Descrição geral: Permitir que o usuário consiga excluir as postagens adicionadas em seu closet.

Pré-condições: Usuário estar logado, e na sessão de closet.

Pós-condições: Usuário visualiza as postagens sem a postagem removida.

Fluxo principal:

1. O usuário clica no ícone de x na postagem que deseja excluir.
2. O sistema exibe um *spinner* de carregamento e ao final remove a postagem da base e da tela.

3.2.5. Caso de uso: Visualizar *feed*

Descrição geral: Permitir que o usuário consiga visualizar as postagens de outros usuários.

Pré-condições: Usuário estar logado.

Pós-condições: Usuário visualiza as postagens correspondentes.

Fluxo principal:

1. O usuário clica em selecione um estilo.
2. O sistema exibe uma lista com todos os estilos disponíveis.
3. O usuário seleciona um estilo.
4. O sistema exibe todas as postagens relacionadas com esse estilo.

Fluxo alternativo:

A.1. O usuário já possui postagens.

A.1.2. O sistema mede qual estilo está mais presente nas postagens do usuário.

A.1.3. O sistema exibe as postagens relacionadas com esse estilo.

3.2.6. Caso de uso: Cadastrar usuário

Descrição geral: Permitir que o usuário possa se cadastrar na plataforma.

Pré-condições: Entrar na plataforma.

Pós-condições: Usuário visualiza o *feed* da página.

Fluxo principal:

1. O usuário clica em “Se não possui conta, clique aqui”.
2. O sistema exibirá além do campo de *login* e senha já exibidos, os campos de *nickname* e e-mail.
3. O usuário preenche os dados e clica em seguir.
4. O sistema valida os dados e se correto salva os dados na base e redireciona para o *feed*.

Primeiro fluxo alternativo:

A.1.1. O usuário clica em “Se não possui conta, clique aqui”.

A.1.2. O sistema exibirá além do campo de *login* e senha já exibidos, os campos de *nickname* e e-mail.

A.1.3. O usuário clica em seguir sem preencher os dados.

A.1.4. O sistema exibe em cada campo não preenchido, uma mensagem com o nome do campo seguido de Inválido.

Segundo fluxo alternativo:

A.2.1. O usuário clica em “Se não possui conta, clique aqui”.

A.2.2. O sistema exibirá além do campo de *login* e senha já exibidos, os campos de *nickname* e e-mail.

A.2.3. O usuário preenche os dados, insere um usuário já cadastrado por outro usuário e clica em seguir.

A.2.4. O sistema exibe uma mensagem informando “Já possui cadastro para esse *login*”.

3.2.7. Caso de uso: Atualizar cadastro

Descrição geral: Permitir que o usuário possa atualizar seus dados na plataforma.

Pré-condições: Usuário estar logado.

Pós-condições: Usuário visualiza o *feed* da página.

Fluxo principal:

1. O usuário clica no nome do usuário no menu superior.
2. O sistema exibirá um menu *dropdown* com a opção atualizar;
3. O usuário clica em atualizar.
4. O sistema exibe os campos: senha, confirmar senha, *nickname* e e-mail.

5. O usuário preenche os dados e clica em seguir.

6. O sistema atualiza os dados e redireciona o usuário de volta para o *feed*.

Fluxo alternativo:

A.1.1. O usuário clica no nome do usuário no menu superior.

A.1.2. O sistema exibirá um menu *dropdown* com a opção atualizar.

A.1.3. O usuário clica em atualizar.

A.1.4. O sistema exibe os campos: senha, confirmar senha, *nickname* e e-mail.

A.1.5. O usuário clica em seguir sem preencher os dados.

A.1.6. O sistema exibe em cada campo não preenchido, uma mensagem com o nome do campo seguido de Inválido.

3.2.8. Caso de uso: Logar usuário

Descrição geral: Permitir que o usuário possa se logar na plataforma.

Pré-condições: Entrar na plataforma.

Pós-condições: Usuário visualiza o *feed* da página.

Fluxo principal:

1. Ao entrar no link o sistema exibe os campos de *login* e senha.

2. O usuário preenche os dados e clica em seguir.

3. O sistema valida os dados e se correto redireciona o usuário para o *feed*.

Fluxo alternativo:

A.1.1. Ao entrar no link o sistema exibe os campos de *login* e senha.

A.1.2. O usuário preenche os dados e clica em seguir.

A.1.3. O sistema valida os dados e se forem incorretos, exibe uma mensagem informando “Senha inválida”.

3.2.9. Caso de uso: Logar com o Google

Descrição geral: Permitir que o usuário possa se logar na plataforma pelo Google.

Pré-condições: Entrar na plataforma.

Pós-condições: Usuário visualiza o *feed* da página.

Fluxo principal:

1. Ao entrar no link o sistema exibe a tela de *login* e abaixo um botão escrito “Google”.
2. O usuário clica no botão escrito “Google”.
3. O sistema se conecta com o *Google API* que exibirá na tela a lista de contas salvas no dispositivo.
4. O usuário escolhe a conta.
5. O sistema valida os dados e se correto redireciona o usuário para o *feed*.

3.2.10. Caso de uso: Visitar perfil de outro usuário

Descrição geral: Permitir que o usuário visualize o perfil de outro usuário

Pré-condições: Estar logado e no *feed*.

Pós-condições: Usuário visualiza a página de perfil do usuário.

Fluxo principal:

1. Ao visualizar uma postagem, o usuário clica no nome do usuário que postou.
2. O sistema exibe a página do perfil do usuário, com nome do usuário, botão de seguir e closet com postagens do usuário.

3.2.11. Caso de uso: Seguir usuário

Descrição geral: Permitir que o usuário siga outro usuário.

Pré-condições: Estar visualizando o perfil de outro usuário.

Pós-condições: Usuário visualiza a página de perfil do usuário.

Fluxo principal:

1. O usuário clica em seguir.
2. O sistema armazena essa informação no banco de dados e troca o texto do botão para seguindo.

3.2.12. Caso de uso: Deixar de seguir usuário.

Descrição geral: Permitir que o usuário possa deixar de seguir outro usuário

Pré-condições: Estar visualizando o perfil de outro usuário.

Pós-condições: Usuário visualiza a página de perfil do usuário.

Fluxo principal:

1. O usuário clica em seguindo.
2. O sistema remove essa informação no banco de dados e troca o texto do botão para seguir.

3.2.13. Caso de uso: Visualizar seguidos

Descrição geral: Permitir que o usuário consiga visualizar as postagens de usuários que ele segue.

Pré-condições: Usuário estar logado.

Pós-condições: Usuário visualiza as postagens correspondentes.

Fluxo principal:

1. O usuário clica em seguindo.
2. O sistema exibe todas as postagens postadas por todos os perfis seguidos pelo usuário.

3.2.14. Caso de uso: Curtir postagem

Descrição geral: Permitir que o usuário consiga curtir postagens.

Pré-condições: Estar logado e no *feed*.

Pós-condições: Usuário visualiza as postagens correspondentes.

Fluxo principal:

1. Ao visualizar uma postagem, o usuário clica no ícone de coração da mesma.
2. O sistema armazena essa informação e altera o ícone para um coração preenchido.

3.2.15. Caso de uso: Visualizar postagens curtidas

Descrição geral: Permitir que o usuário consiga visualizar as postagens curtidas por ele.
Pré-condições: Usuário estar logado.
Pós-condições: Usuário visualiza as postagens correspondentes.
Fluxo principal:

1. O usuário clica em curtidos.
2. O sistema exibe todas as postagens curtidas pelo usuário

3.2.16. Caso de uso: Comentar em postagem

Descrição geral: Permitir que o usuário consiga comentar postagens.
Pré-condições: Estar logado e no *feed*.
Pós-condições: Usuário visualiza as postagens correspondentes.
Fluxo principal:

1. Ao visualizar uma postagem, o usuário clica em ver comentários.
2. O sistema retorna um modal, com campo de texto e um botão de comentar.
3. O usuário preenche o campo com seu comentário e clica em comentar.
4. O sistema irá validar e registrar essa informação.
5. O sistema exibe o comentário no mesmo modal.

3.2.17. Caso de uso: Visualizar comentários em postagem

Descrição geral: Permitir que o usuário consiga acessar comentários.
Pré-condições: Estar logado e no *feed*.
Pós-condições: Usuário visualiza as postagens correspondentes.
Fluxo principal:

1. Ao visualizar uma postagem, o usuário clica em ver comentários.
2. O sistema retorna um modal, com todos os comentários, contendo o nome do usuário que postou, conteúdo e horário de postagem.

A representação dos casos de uso citados, bem como suas interações, podem ser observados no diagrama de casos de uso representado abaixo.

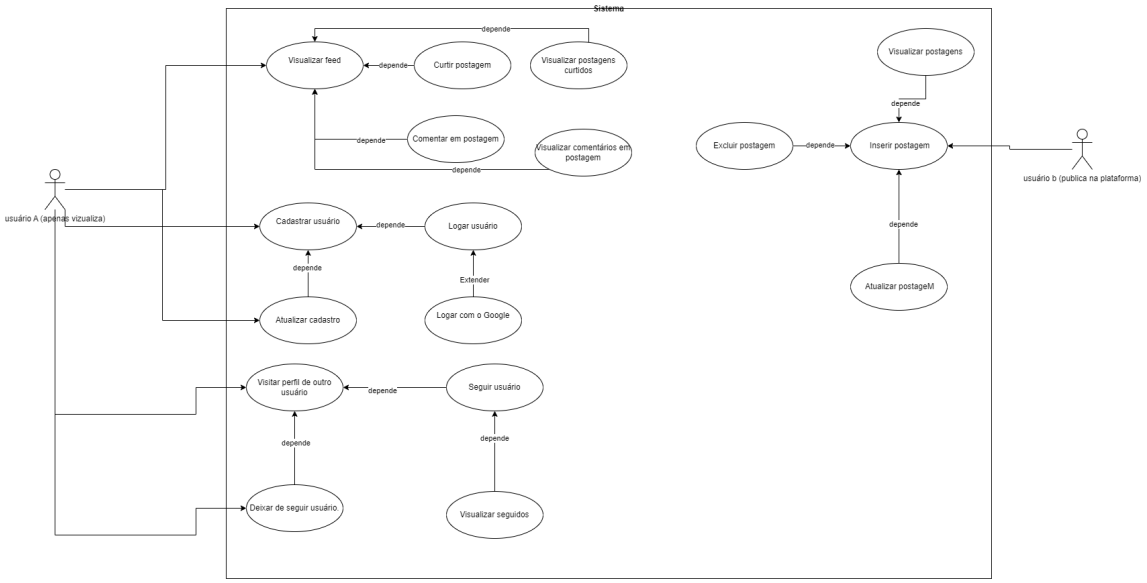

Figura 8 – Diagrama de casos de uso do Costylist

3.3. Tecnologias utilizadas

3.3.1. Javascript

“Todas as informações sobre o *Javascript* foram retiradas diretamente do site oficial da Amazon (AMAZON, 2024)”.

É uma linguagem de programação utilizada na criação de páginas *web* interativas. Ela permite que sejam adicionadas funcionalidades como animações, mapas interativos, menus suspensos, dentre outros eventos interativos que podem ser adicionados.

Ele é utilizado tanto do lado cliente quanto do lado do servidor, sendo extremamente versátil. Um dos pontos que ressaltam sua versatilidade é de poder abordar múltiplos paradigmas. Outra característica é o fato de possuir uma tipagem dinâmica fraca, o que significa que é possível desenvolver sem atribuir tipo para as variáveis criadas.

Além disso, é possível executar o *JavaScript* em qualquer página *web*, o que torna ele uma escolha segura em relação ao funcionamento do ambiente. Outro ponto

que corrobora positivamente para o uso do *JavaScript* é a facilidade com que é possível aprender e codificar.

A linguagem *JavaScript* foi escolhida para ser utilizada devido a sua versatilidade e facilidade no uso. Apesar disso, na maior parte do *backend* foi utilizado *Typescript* para ter uma segurança maior de consistência dos dados, através da validação de tipagem.

3.3.2. Typescript

"Todas as informações sobre o *Typescript* foram retiradas diretamente do site oficial (TYPESCRIPT, 2024) ".

É uma linguagem, criada pela *Microsoft* com a finalidade de adicionar uma sintaxe no *JavaScript* que permite a tipagem na declaração de variáveis e validação de tipo na execução da aplicação. Essa adição possibilita uma melhor representação de objetos e auxilia na aplicação do paradigma de programação orientada a objetos, com validação de tipo inclusa.

Apesar de conter mais paradigmas e ser considerada uma linguagem diferente, o *Typescript* possui a mesma flexibilidade de execução que o *JavaScript*. Isso se deve ao fato de que ele converte o código para *JavaScript* antes de executá-lo, permitindo com que seja possível a execução em qualquer ambiente que suporte *JavaScript*.

A escolha pelo seu uso se deu devido à necessidade de uma validação segura dos tipos de objetos presentes no *backend*. Outro ponto relevante é que o *Nest.js* foi projetado para ser usado em projetos *Typescript.*

3.3.3. Node.JS

"Todas as informações sobre o *Node.JS* foram retiradas diretamente do site oficial (NODE.JS, 2024) ".

É um ambiente de execução para *Javascript*, de código aberto, que permite a execução do *Javascript* no *backend*. Possui uma arquitetura assíncrona orientada a eventos. Isso possibilita o uso de funções assíncronas e de que múltiplas conexões possam ser tratadas sem usar um *thread* de processamento adicional.

Além disso, apesar de ser single-thread, *Node.js* permite o uso de múltiplos núcleos de processamento, possibilitando balanceamento de carga e comunicação entre

processos, garantindo a escalabilidade em ambientes com múltiplos núcleos. Isso torna *Node.js* uma escolha robusta para o desenvolvimento de servidores, aplicações *web* e automação de tarefas.

Por esse motivo, ele foi escolhido para ser o ambiente de execução no qual o backend do projeto foi projetado.

3.3.4. NestJS

"Todas as informações sobre o *NestJS* foram retiradas diretamente do site oficial (NESTJS, 2024) ".

É um *framework* desenvolvido para auxiliar na construção de aplicações *backend* facilmente escaláveis usando *Node.js* como ambiente de execução. Apesar de ser desenvolvida para ter suporte nativo ao *Typescript*, permite também o uso de *Javascript*.

O ponto de destaque no *Nest.js* é a possibilidade de criar aplicações modulares com cada módulo tendo responsabilidades definidas e podendo ser chamados e utilizados por outros módulos. Essa modularização permite uma estrutura sólida, organizada e simples de se entender e de realizar manutenção em pontos focais.

O *NestJs* foi escolhido justamente pela sua modularidade que facilita a organização, divisão de responsabilidade e entendimento.

3.3.5. React

"Todas as informações sobre o *React* foram retiradas diretamente do site oficial (REACT, 2024) ".

É uma biblioteca *Javascript* para construção de interfaces de usuário para páginas web. O conceito principal em torno do *React* é o fato de possuir componentes reutilizáveis por outros componentes, evitando a duplicidade, facilitando a divisão de responsabilidade, padronizando e agilizando o desenvolvimento.

Através do uso de uma extensão sintaxe chamada *JSX*, é possível através de *JavaScript* manipular o layout em HTML, inclusive adicionando valores e variáveis oriundos do *JavaScript*. Essa possibilidade se torna vantajosa devido à capacidade do *React* em atualizar a interface depois de alterada.

O *React* pode ser usado em diferentes plataformas *web*, além de possuir uma curva de aprendizado curta. A escolha por essa biblioteca se deve à possibilidade de reutilização dos seus componentes e ampla possibilidade de execução, que abrange todos os ambientes onde o *Javascript* pode ser executado.

3.3.6. Jest

"Todas as informações sobre o *Jest* foram retiradas diretamente do site oficial (JEST, 2024) ".

É um *framework* para criação de testes unitários automatizados, no qual possui uma boa gama de recursos disponíveis e uso simples. É compatível com vários *frameworks* que utilizam *JavaScript*, sendo importante destacar seu uso no *React* e no *Node.js*, onde seu uso é difundido.

Um recurso notável é o uso de *snapshots*, que facilita a validação de grandes objetos, garantindo que o estado atual corresponda ao esperado. Além disso, os testes são executados de forma isolada e paralela, maximizando o desempenho.

O *Jest* também possui uma documentação robusta, tornando a busca por conteúdo que irá auxiliar na escrita e manutenção de testes mais rápida. Outro ponto de destaque é a capacidade de gerar relatórios de cobertura de código automaticamente, oferecendo uma visão abrangente sobre o quanto do código foi testado.

Ele foi usado na criação de testes unitários desse projeto, devido à facilidade de se realizar testes com ele no *Typescript* e *Javascript*, e da documentação robusta.

3.3.7. Gemini

"Todas as informações sobre o *Gemini* foram retiradas diretamente do site oficial (GOOGLE, 2024) ".

É uma ferramenta de inteligência artificial avançada desenvolvida pelo *Google*, com a finalidade de auxiliar em atividades como escrita, planejamento e aprendizado. Seu foco se dá no reconhecimento de padrões linguísticos e na busca por informações em diferentes fontes, a fim de gerar respostas adequadas.

Além de texto, o *Gemini* consegue receber imagens como entrada e avaliar elas podendo ser consumido via *API* fornecida pelo *Google*.

No contexto do Costylist, o *Gemini* foi usado em dois momentos: o primeiro para avaliar as imagens que os usuários postam, evitando imagens pornográficas ou que fujam da temática de moda. Já o segundo uso foi para validar os comentários postados pelos usuários e evitar que sejam postados comentários ofensivos.

3.3.8. PostgreSQL

"Todas as informações sobre o *PostgreSQL* foram retiradas diretamente do site oficial (POSTGRESQL, 2024) ".

É um sistema de gerenciamento de banco de dados objeto-relacional (ORDBMS) que se baseia no *POSTGRES*. Ele foi desenvolvido pelo Departamento de Ciência da Computação da Universidade da Califórnia, em Berkeley.

O *PostgreSQL* suporta grande parte do padrão *SQL* e oferece diversas funcionalidades. É possível elencar consultas complexas, chaves estrangeiras, gatilhos, visões atualizáveis, integridade transacional e controle de concorrência de múltiplas versões.

Além disso, o *PostgreSQL* permite extensões personalizadas por parte do usuário, incluindo a adição de novos tipos de dados, funções, operadores, funções agregadas, métodos de índice e linguagens de programação. Graças ao fato de ser código aberto, o *PostgreSQL* pode ser utilizado, modificado e distribuído por qualquer pessoa, sem custos, para fins privados, comerciais ou acadêmicos.

Ele foi escolhido como banco de dados devido à facilidade de uso para este projeto académico. Além disso, a grande gama de recursos e o fato de ser código aberto e permitir o uso sem custos adicionais também foram fatores relevantes.

3.3.9. TypeORM

"Todas as informações sobre o *TypeORM* foram retiradas diretamente do site oficial (TYPEORM, 2024) ".

TypeORM é um ORM (*Object-Relational Mapping*), ou seja, uma ferramenta para mapeamento de objetos em elementos de banco de dados relacionais. Ele pode ser executado em diversas plataformas, mas seu nome se deve ao fato de, durante seu desenvolvimento ter sido pensado para ter seu uso no *Typescript*.

Ele suporta uma boa gama de padrões de mapeamento de dados, além de possibilitar a criação de aplicações escaláveis e de fácil manutenção. A facilidade e escalabilidade se deve ao fato de extinguir a necessidade do uso de queries de *SQL*, podendo se ter a interação com o banco de dados somente por meio do *Typescript* e do uso do *TypeORM*.

A escolha nesta aplicação se deve à necessidade de possuir um *ORM* na aplicação e o *TypeORM* ser o mais popular. Esse fato facilita na busca de informações que podem auxiliar o desenvolvimento.

3.3.10. Passport

"Todas as informações sobre o *Passport* foram retiradas diretamente do site oficial (PASSPORT, 2024) ".

É um *middleware* para *Node.js* que simplifica a implementação de autenticação e autorização em aplicações web. Apesar de sua facilidade de uso, possui abrangência, mas possibilidades de autenticação.

O *Passport* oferece um conjunto abrangente de estratégias, permitindo autenticações variadas, como nome de usuário e senha, bem como através de *logins* em redes sociais, como *Facebook*, *Twitter*, *Google*, entre outras. Essa abrangência torna o *Passport* uma boa escolha para desenvolvedores que desejam implementar soluções de identidade de forma eficiente, independentemente do nível de experiência.

Outro ponto importante é a possibilidade de utilizar de maneira simples, soluções seguras, com estratégias que incluem a validação da presença de *tokens* criptografados.

Ele foi utilizado no contexto da aplicação para criação de autenticação com usuário e senha padrão e *login* com o *Google*. Além disso, a facilidade de integração com a API do *Google* e a segurança do seu processo de autenticação via *token* criptografado também foram fatores relevantes.

3.3.11. Bootstrap

“Todas as informações sobre o uso do *Bootstrap* foram retiradas diretamente do site oficial (BOOTSTRAP, 2024) ”.

É um kit de ferramentas para o desenvolvimento *frontend* que permite o desenvolvimento rápido de sites responsivos e com design consistente e moderno. Nas versões mais recentes, o *Bootstrap* possui uma arquitetura modular, que permite a seleção dos componentes necessários.

A instalação e uso do *Bootstrap* é simples e pode ser usado com integração via CDN ou via gerenciador de pacotes. Além disso, ele recebe atualizações constantes nos seus recursos e na sua documentação abrangente.

A combinação das classes presentes no *Bootstrap* permite a criação rápida de componentes únicos. Além da criação e personalização de componentes, uma boa gama de componentes e elementos estão disponíveis na sua documentação, bem como instruções sobre como implementar.

O *Bootstrap* foi escolhido devido a sua vasta documentação e comunidade engajada. Esses fatores possibilitam a obtenção de exemplos que auxiliam no desenvolvimento.

3.3.12. Imgur

“Todas as informações sobre o uso do *Imgur* foram retiradas diretamente do site oficial (IMGUR, 2024) ”.

É uma plataforma *online* de hospedagem e compartilhamento de imagens, criada em 2009 por Alan Schaaf. O *Imgur* possui uma base sólida de usuários devido ao uso simples, tendo atingido em 2019 a marca de 300 milhões de usuários por mês.

Além do uso facilitado, o *Imgur* oferece poucas limitações de uso e conta com serviço de *API* para integração rápida com a plataforma. Esses fatores foram relevantes para a escolha de usar o *Imgur* como principal forma de hospedagem de imagens do *Costylist*.

Ao realizar o *upload* de qualquer imagem no *Costylist*, o sistema após avaliar esse arquivo, através de integração com a *API* do *Imgur*, realiza o upload dessa imagem.

Após isso, é retornado pela *API* a *URL* da publicação no *Imgur* e o *token* de identificação necessário para exclusão do arquivo posteriormente da plataforma.

A vantagem dessa integração se dá no fato de somente ser necessário manter salvo no banco de dados *strings* curtas, a *URL* já encurtada e o *token*. Além da facilidade desse processo, isso possibilita com que sejam gastos menos recursos de armazenamento do que se fossem armazenados os arquivos de imagem.

A integração com *apis* externas como o Gemini e o Imgur, e as interações entre componentes internos, podem ser observados no diagrama de componentes disponível abaixo.

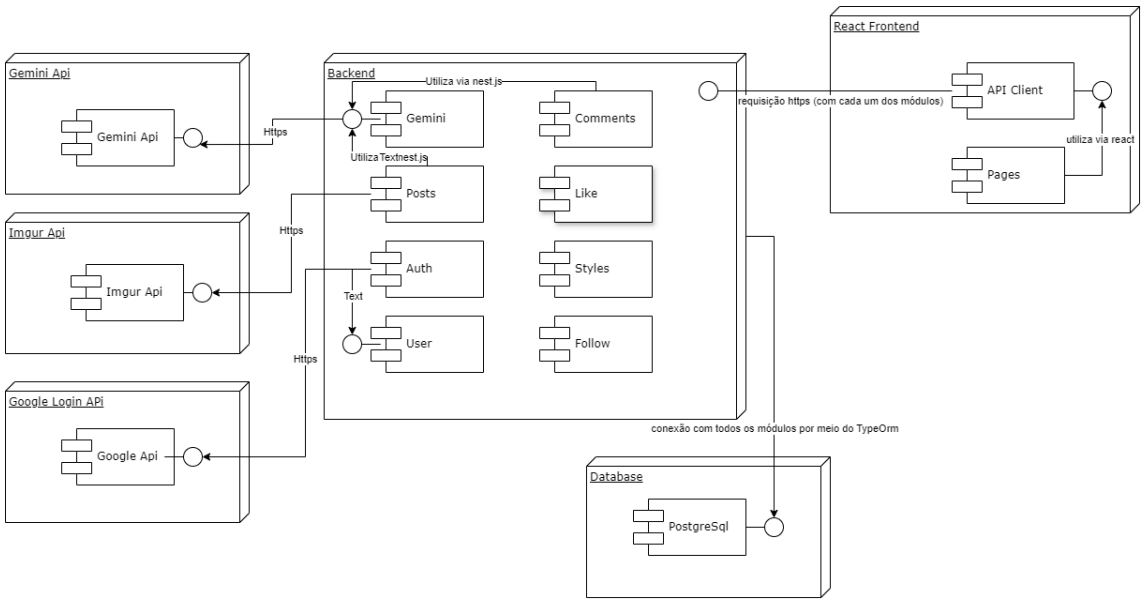

Figura 9 – Diagrama de componentes do Costylist.

3.4. Considerações Finais

Neste capítulo foram apresentadas as tecnologias que foram utilizadas no desenvolvimento, bem como os casos de uso e o diagrama de classe. Essas informações são importantes para evidenciar as etapas do fluxo de desenvolvimento de uma aplicação. Além disso, esse tipo de visão, auxilia no entendimento técnico da mesma.

Apesar disso, é necessária uma demonstração prática do funcionamento da plataforma, percorrendo cada uma das telas e funcionalidades da mesma. Isso será feito

no capítulo abaixo, contendo descrição textual e a partir de imagens de todos os casos de uso listados acima.

4 Apresentação da aplicação

Neste capítulo, a fim de exemplificar as funcionalidades da aplicação, será feita uma demonstração das telas do sistema. Serão visitados todos os casos de uso citados no capítulo anterior.

4.1. Autenticação

4.1.1. *Login* com usuário e senha

A primeira tela exibida quando se entra em qualquer *link* correspondente ao Costylist, é a tela de *login*. Mesmo acessando algum *endpoint* específico, o usuário é redirecionado para o *login* se não houver a presença do *token* de autenticação.

Nessa tela, o sistema apresenta os campos de *login* e senha, a opção de clicar em criar uma conta e o botão para autenticar com o Google. Após preencher os campos, o usuário pode clicar em seguir e prosseguir com a autenticação. Um exemplo desta tela está presente na Figura 8.

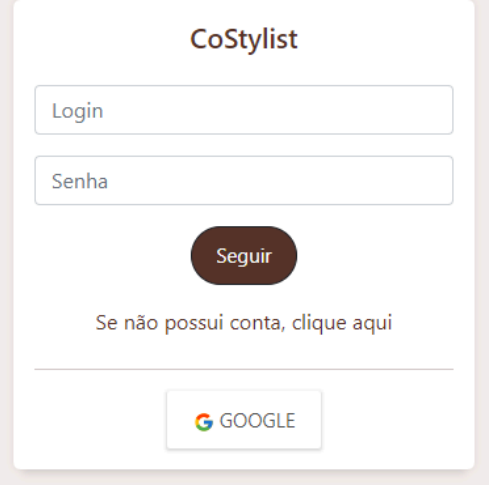

Figura 10 – Tela de *login*.

Caso o usuário clique em seguir, sem preencher os campos um aviso será inserido acima de cada campo não preenchido do formulário. O aviso, como mostrado na Figura 9, corresponde ao nome do campo, seguido da palavra inválido.

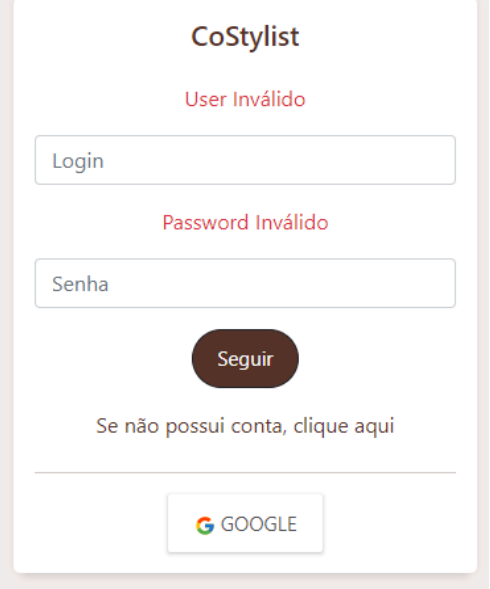

Figura 11 – Validação de campos do *login*.

Além dessa validação, caso o usuário insira um *login* errado ou uma senha errada ele receberá erros correspondentes em forma de alerta. Exemplos dessa validação de *login* inexistente e senha errada podem ser observados nas Figuras 10 e 11, respectivamente.

Figura 12 – Alerta de usuário não encontrado.

Figura 13 – Alerta de senha inválida.

Caso o usuário digite usuário e senha corretos, o *backend* irá gerar um *token* JWT, que será armazenado no *frontend*. Após o armazenamento desse *token* em sessão ele deverá ser repassado em todas as chamadas subsequentes a esta.

Se a presença desse *token* não for identificada em alguma chamada, ou o *token* for inválido, o *frontend* redireciona o usuário novamente a tela de *login*. Após o *frontend* receber e armazenar esse *token*, ele irá redirecionar o usuário para o *feed*

principal.

4.1.2. Login com o Google

Uma outra opção para autenticação, é o *login* social via *Google*, esse processo se inicia ao usuário clicar no botão escrito *Google*. Após clicar no botão, o usuário é redirecionado para uma página de *login* do próprio *Google* onde poderá escolher uma conta para se autenticar.

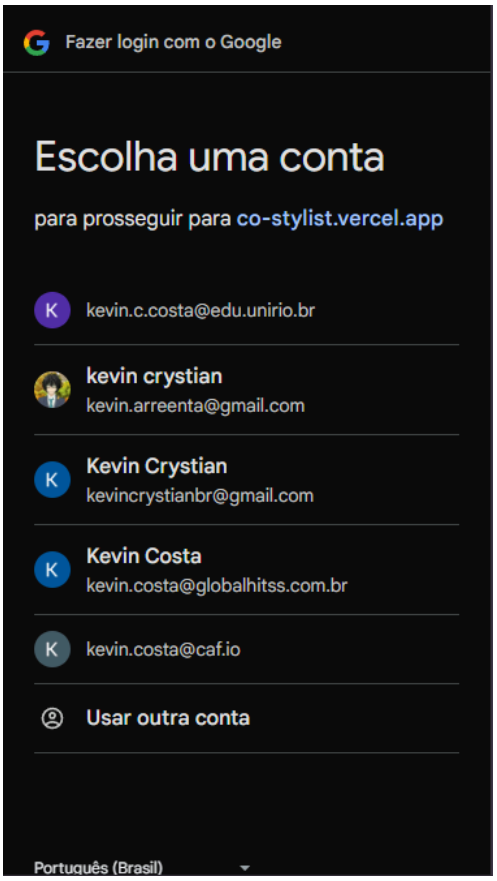

Figura 14 – Seleção de conta Google.

Após realizar o *login*, o *backend* valida o id do Google deste usuário. Se já possuir uma conta com esse id como *login*, a *API* irá retornar para o *frontend* o *token* de autenticação necessário para ele prosseguir. Porém, caso ele não possua, o *backend* criará um perfil do usuário usando o id dessa conta como *login* e o e-mail do Google como e-mail e, a partir desse ponto, prossegue com a criação do *token*.

4.1.3. Cadastro

Caso o usuário clique em “Se não possui conta, clique aqui”, o sistema exibe

mais dois campos, *nickname* e e-mail. Os campos possuem validação caso não sejam preenchidos. A tela pode ser vista na imagem abaixo.

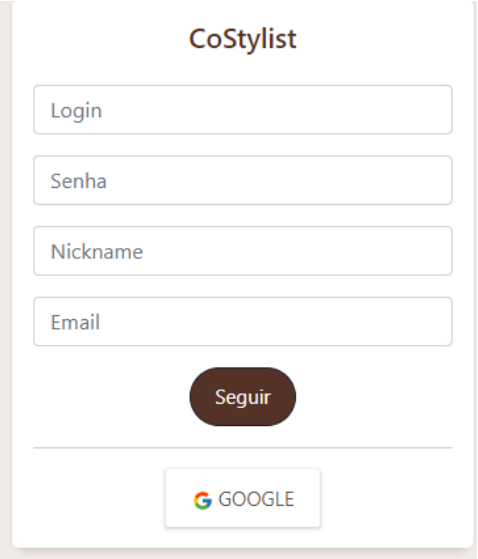

Figura 15 – Cadastro de usuário.

Se um *login* já utilizado for inserido o sistema exibirá um alerta com a mensagem de erro "já possui cadastro para esse *login*". Esse erro pode ser observado na Figura 14.

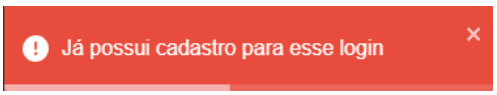

Figura 16 – alerta em caso de usuário já cadastrado.

4.2. Homepage

Logo após se autenticar, a primeira tela que o usuário verá será a *homepage*, onde poderá visualizar postagens de outros usuários. Assim como as outras telas da área logada, no topo da página há um menu com a logo do site, "Meu closet" e o nome do usuário, nesse caso "teste".

Cada sessão será abordada separadamente, e nesse primeiro momento será a *home page*. Além de ser a primeira página a realizar o *login*, ela pode ser acessada

clicando no logotipo. Ela possui, além do menu superior, um campo de seleção de estilos, um menu de seleção de *feed* e *cards* de postagens que aparecerão abaixo.

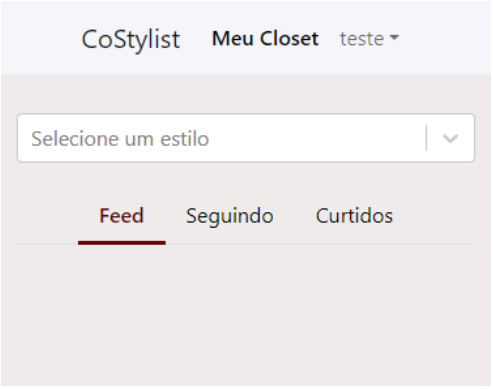

Figura 17 – Menu de seleção de *feeds*.

4.2.1. *Feed*

No *feed* principal, chamado "*Feed*", caso esse usuário ainda não tenha nenhuma postagem própria, o primeiro passo para visualizar novas postagens é selecionar um estilo. Essa ação pode ser realizada ao clicar em "selecione um estilo", abrindo assim, um *input* selecionável com diversos estilos, como visto na imagem abaixo.

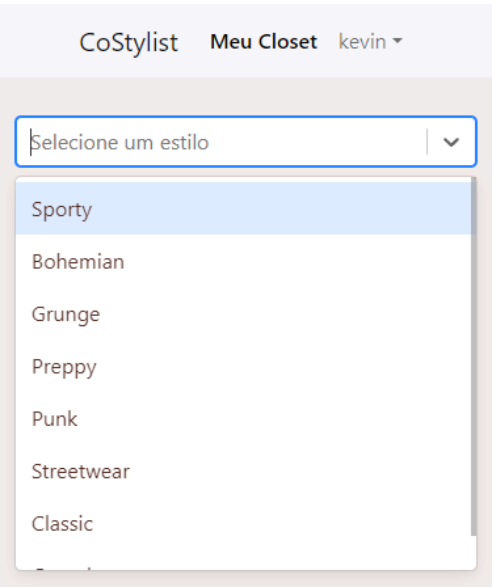

Figura 18 – Seleção de estilo.

Através desse *input*, é possível selecionar um ou mais estilos, bem como remover um estilo já selecionado. Após a seleção do estilo, aparecerão postagens abaixo relacionadas com os estilos selecionados.

Neste primeiro momento, a ordem com que as postagens aparecerão segue apenas o critério de tempo de publicação. Apesar disso, há o desejo de implementação de algoritmos de recomendação mais complexos em trabalhos futuros.

As postagens aparecem em formato de *card*, com imagem em cima e em baixo dos botões e estilo correspondente. Além disso, há a presença de um botão para curtir, no topo, que serve para adicionar essa postagem a aba de postagens curtidas.

Na parte de baixo do *card*, existem duas seções. Na primeira, se encontram dois botões e ao meio o estilo correspondente. Primeiro, à esquerda se encontra o botão com o nome do usuário que realizou a postagem, e ao clicar redireciona para a página pessoal dele. À direita, se encontra o botão “Loja”, que redireciona o usuário para loja cadastrada no momento de publicação.

Na segunda sessão, há o botão de visualização de comentários, com a fraseologia “Ver comentários”. Ao usuário clicar nele, será exibido um modal para visualização e inserção de comentários. Esse tópico será melhor detalhado posteriormente. Um exemplo do visual de uma postagem, pode ser observado na imagem abaixo.

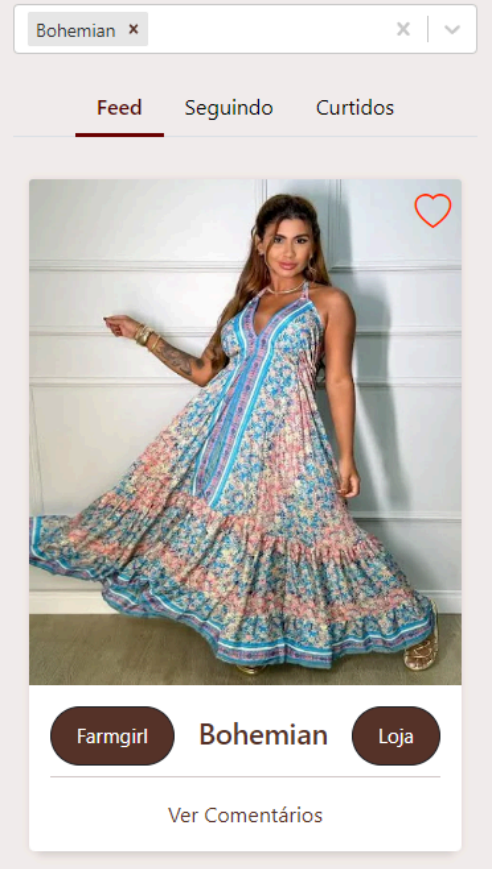

Figura 19 – *Card* de postagem.

Ao clicar em “Ver Comentários”, o sistema irá abrir um modal onde o usuário terá a possibilidade de visualizar e inserir novos comentários. Os comentários são exibidos acompanhados do nome do usuário que postou e, antes de serem salvos pelo sistema, passam por integração com o *Gemini Api.*

O *prompt* de integração como o *Gemini* inserido, além de contextualizar, serve para definir alguns critérios importantes. Essa validação se embasa na premissa de que os comentários devem ser respeitosos, não ofensivos e não podem conter conteúdo inapropriado.

Caso não atenda algum critérios, será exibido um alerta de erro com os dizeres “comentário não permitido”. O modal de adição e visualização de comentários pode ser visualizado na Figura 18, e o alerta para comentários inapropriados, na Figura 19.

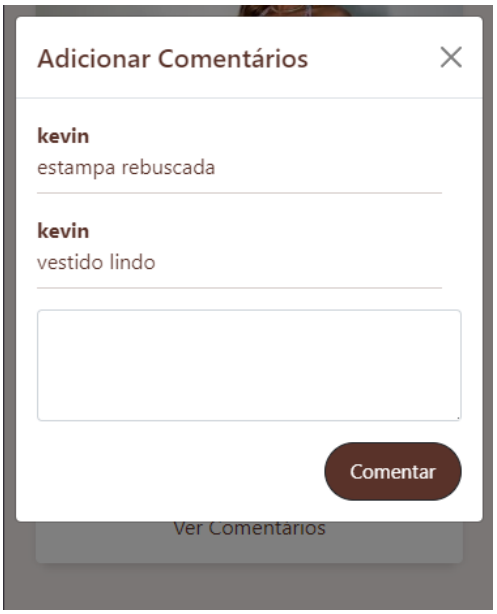

Figura 20 – Modal de adição de comentários.

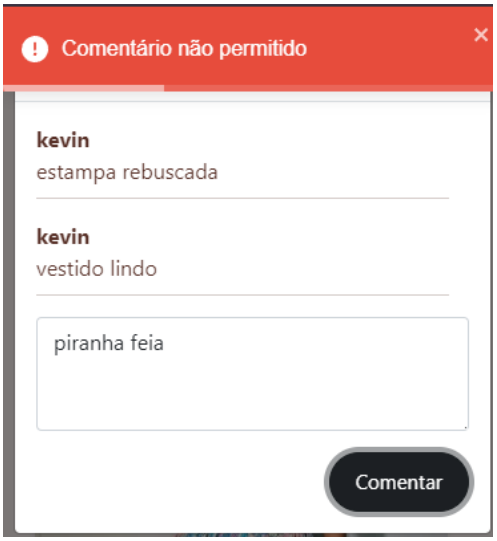

Figura 21 – Alerta de comentário não permitido.

4.2.2. Seguindo

Ao clicar em seguindo, o sistema exibirá apenas postagens publicadas pelos usuários os quais o usuário que realiza a operação seguir. Seguir um usuário consiste em acessar o perfil pessoal deste usuário, ao clicar no botão com o nome dele em alguma postagem, e dentro dessa página, clicar em seguir. Essa operação será melhor detalhada quando for feita a explanação acerca do perfil pessoal de outros usuários.

O comportamento e visual da página é similar ao *feed* principal, apenas

diferenciando no fato de só permitir postagens de usuários seguidos, isso pode ser visto na Figura 20. A presença dessa aba e do botão seguir se deve à possibilidade do usuário selecionar quais pessoas deseja que apareçam para ele, personalizando assim sua busca por referências.

Figura 22 – *Feed* de usuários seguidos.

4.2.3. Curtidos

A aba “Curtidos” também possui aparência e comportamento similar às outras abas da página inicial. Ao clicar nela, o sistema exibe apenas postagens curtidas pelo usuário. O principal ganho dessa implementação se dá na possibilidade do usuário curtir postagens que deseja salvar para visualizar novamente, podendo acessar de maneira simplificada essas postagens.

Além da função de salvar postagens, funcionalidades de curtir, normalmente em redes sociais, exercem o papel de alimentar algoritmo de recomendação. Apesar disso,

não houve nesse trabalho tempo hábil para a criação de um algoritmo mais complexo que considerasse as características das postagens curtidas.

É possível remover uma postagem da aba “Curtidos”, bastando clicar novamente no ícone de coração nas postagens, o que o fará ficar transparente e removerá a postagem dela. Um exemplo visual dessa aba se encontra na Figura 21.

Figura 23 – *Feed* de publicações curtidas.

4.3. Perfil

Caso o usuário clique no botão com o nome de outro usuário, o sistema exibirá a página deste usuário, somente com postagens dele. No topo da página se tem o nome do usuário a esquerda e à direita aparece o botão seguir, que adiciona todas as postagens deste usuário na aba seguindo do *feed*.

Um pouco mais abaixo, existe a presença do “*closet”* desse usuário, que consiste em todas as postagens publicadas pelo mesmo. O usuário pode navegar pelas postagens

usando os botões laterais presentes em forma de seta. Os *cards* presentes na página do usuário são similares aos cards presentes na página inicial, tendo os mesmos recursos, apenas se diferenciando na ausência do botão para a página do usuário.

Um exemplo visual desta página, bem como seus elementos, pode ser encontrado na Figura 22. Nela é possível observar a página pessoal do usuário “kevin”, criado para fins de demonstração.

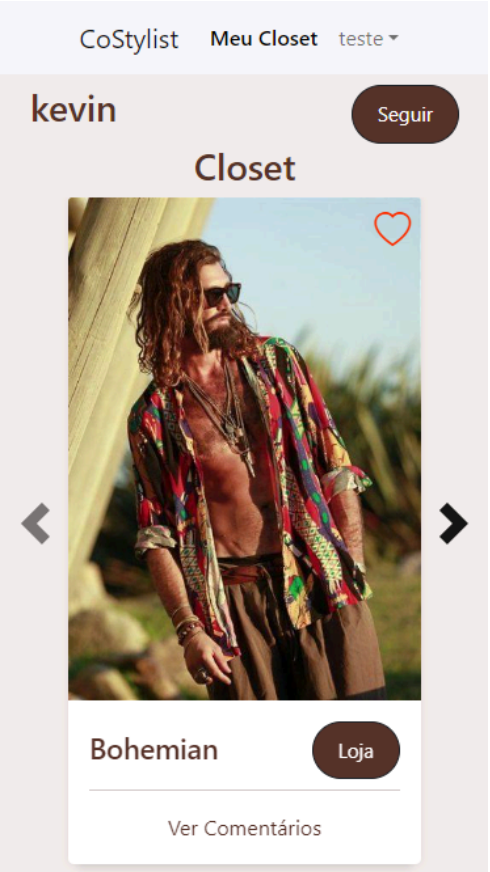

Figura 24 – Perfil pessoal do usuário Kevin.

4.4. Meu Closet

Caso o usuário clique em “Meu closet”, no menu superior, ele será redirecionado para o local onde poderá gerenciar suas postagens. Essa página é composta por um título indicando o estilo com mais postagens, um botão para adicionar postagens, e cards de postagem com opções de excluir ou editar. Um exemplo desta página pode ser visto na Figura 23.

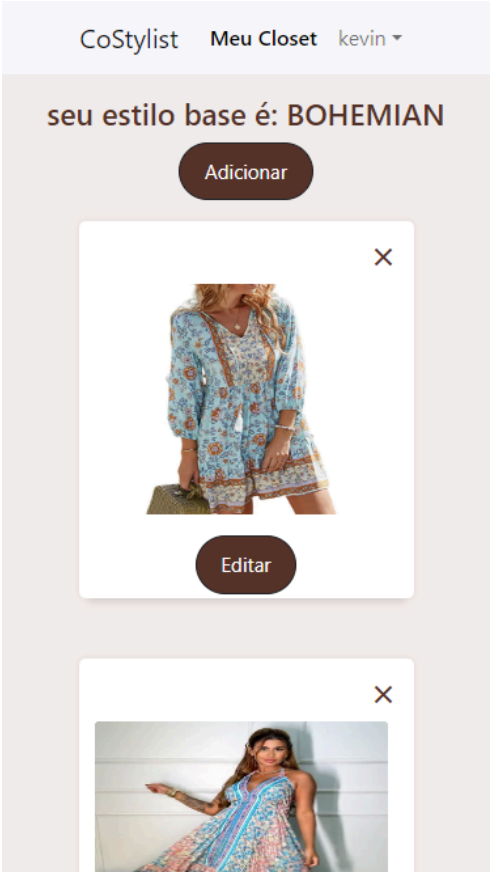

Figura 25 – página Meu Closet do usuário Kevin.

A indicação do estilo base é feita contabilizando o estilo com mais incidência nas postagens deste usuário. Essa informação é novamente retomada na página inicial, para automaticamente, logo após realizar o *login*, preencher a seleção de estilo automaticamente com esse estilo mais publicado.

Abaixo do indicador de estilo, existe a presença do botão adicionar. Após o usuário clicar neste botão o sistema exibe um modal com o título de "Adicionar postagem" e possibilita a adição de novas postagens.

Os campos presentes nesse modal são, "escolher arquivos", "selecionar estilo" e "*Link*". Além desses campos, o modal possui botão para fechar e um botão enviar que prossegue com a adição de uma nova postagem. Um exemplo visual desse modal, pode ser observado na Figura 24.

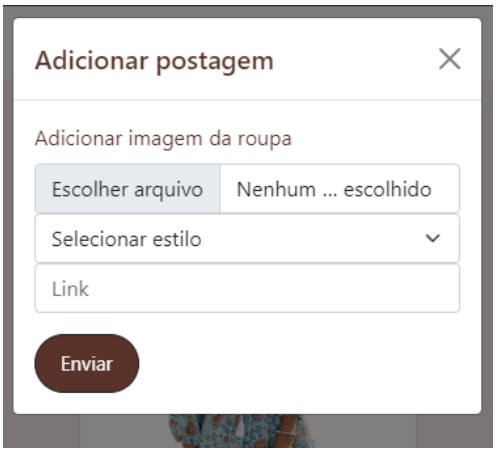

Figura 26 – Modal de adição de postagens.

O campo “escolher arquivo” permite os usuários a realizarem o upload de imagens, desde que possuam como formato JPEG ou PNG. Além disso, todas as fotos adicionadas passam pela validação do *Gemini API*. O critério utilizado para aprovação é se adequar ao contexto de uma rede social sobre vestuário e não possuir conteúdo lascivo e inadequado.

Para as situações onde a imagem inserida não se adequa aos padrões definidos e elencados anteriormente, foi criado um alerta para realizar essa sinalização ao usuário. Esse alerta pode ser observado na Figura 25.

Figura 27 – Alerta no caso de foto não permitida.

Outro campo presente neste modal é o de seleção de estilo, que mostra todos os estilos cadastrados na base de dados e que podem ser usados pelo usuário. Além deste, também há o campo chamado “*Link*”, onde o usuário deve inserir o *link* da loja onde ele comprou o produto, ou uma loja confiável que a venda.

Depois de preencher e clicar no botão “Enviar” as informações são enviadas

para o sistema, onde ele realiza o *upload* dessa imagem na rede social Imgur. Essa integração é feita para que seja possível armazenar apenas a *URL* da imagem e dessa forma, seja possível economizar espaço no banco de dados.

Após o registro desses dados no banco de dados, essa postagem aparece na página "Meu Closet" apresentando a imagem, um ícone de x e um botão de editar. Caso o ícone de x seja clicado a postagem será permanentemente deletada, com a imagem também sendo removida do *Imgur*.

Caso o usuário clique em editar, será aberto o modal de edição de dados, onde o sistema exibirá um modal similar ao de adição de postagem, porém, com outro título. Funcionalmente, esse modal atua substituindo na base de dados as informações da postagem antiga, removendo a imagem publicada anteriormente do *Imgur*, bem como adicionando a nova imagem. A similaridade visual pode ser conferida na Figura 26.

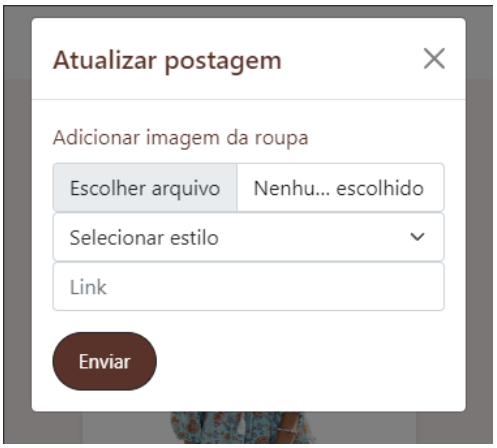

Figura 28 – Modal de atualização de postagem.

4.5. Menu cadastral

Ao clicar na sessão do menu superior contendo o nome do usuário, o sistema exibe um menu *dropdown* contendo duas opções relacionadas a informações cadastrais, "Deslogar" e "Atualizar". Esse menu pode ser observado na Figura 27. Caso "Deslogar" seja pressionado, o *token* de autenticação é removido do navegador e o usuário é redirecionado para a tela de *login*.

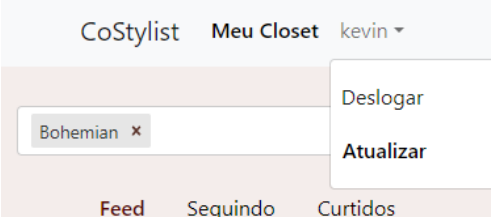

Figura 29 – Menu de opções cadastrais.

Outra opção no menu aberto ao clicar no nome de usuário é o “Atualizar” que ao ser pressionado redireciona o usuário para a tela de atualização de dados cadastrais. Nesta tela, ilustrada na Figura 28, o sistema exibe os campos: senha, confirmar senha, *nickname* e e-mail. Ao preencher os campos e clicar em “Atualizar” o sistema irá atualizar os dados cadastrais deste usuário.

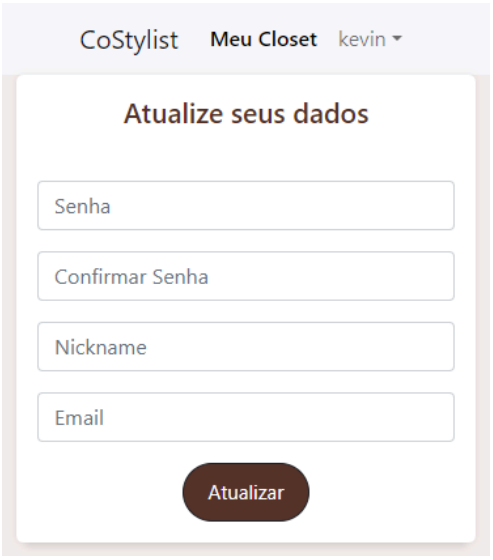

Figura 30 – Tela de atualização de dados.

5 Conclusão

5.1. Considerações finais

Este trabalho descreveu o desenvolvimento de uma plataforma *web* com característica de rede social, para compartilhamento de sugestões relacionadas à moda pessoal. Esta plataforma demonstrou potencial devido à facilidade do seu uso de se tornar uma ferramenta para aquisição de peças de vestuário. Além disso, existe a expectativa de fomento das relações dentro de subculturas, sendo intermediadas através da busca por um estilo pessoal similar.

O *frontend* do Costylist foi construído usando *React* para atender a estrutura de *Single page aplication*. Apesar de ser responsivo, graças ao uso do *Bootstrap*, o *design* da plataforma foi pensado prioritariamente para dispositivos móveis. Já o *backend* utiliza *Nest.js* como *framework*, gerando uma *API* eficaz, de manutenção simplificada e modular, o que ajuda na escalabilidade e implementação de novos recursos.

A experiência de desenvolvimento deste trabalho foi enriquecedora por possibilitar a exploração de anseios pregressos de uma plataforma com esse escopo. Outro ponto de satisfação foi fazer parte de toda a etapa de criação da plataforma, desde a elucidação dos requisitos até a plataforma á funcional, podendo ser acessada pelo *link*: https://costylist-front.vercel.app/

5.2. Limitações

Durante o desenvolvimento da plataforma houveram algumas limitações. É possível destacar a falta de recursos financeiros disponíveis, pouca quantidade de tempo disponível e falta de conhecimento em alguns tópicos específicos.

A falta de recursos financeiros foi determinante na escolha de ferramentas e soluções. Foram escolhidas soluções gratuitas para hospedagem e armazenamento, como uma base de dados em *Postgresql* com limite bem baixo de requisições simultâneas. Outra escolha pautada no custo foi a integração com o *Imgur*, que possui limitações de escalonamento, ao invés de uma solução própria de armazenamento de imagem.

A quantidade limitada de tempo dificultou a inserção de novas funcionalidades, como a criação de um *feed* infinito e outras melhorias na experiência do usuário. O motivo principal do tempo escasso se deve ao fato do único desenvolvedor da plataforma e autor do trabalho precisar se dividir entre múltiplas atividades. Essas atividades são este projeto, emprego em tempo integral como desenvolvedor *fullstack* e matérias da graduação de sistemas de informação.

A falta de conhecimento foi limitante para a projeção do projeto em alguns aspectos. Havia o desejo de incluir, já no escopo atual, o uso de *machine learning* para auxiliar na definição das categorias, porém, não havia conhecimento na área de ciência de dados. Outro ponto que foi dificultado pela falta de conhecimento nessa área foi a criação de um sistema de algoritmo para direcionamento das postagens nos *feeds*.

5.3. Trabalhos Futuros

Para trabalhos futuros, as sugestões se dividem em melhorias que poderiam ser implementadas e estudos de estratégias a serem abordadas e de impactos causados. Cada uma das propostas será detalhada a seguir.

5.3.1. Implementações

Um dos pontos de maior ganho para a plataforma seria a criação de um algoritmo de recomendação para postagens nos diferentes *feeds*. A partir das postagens curtidas, seria possível ter um mapeamento das características que o usuário mais procura nas roupas e exibir as que mais se enquadram nesse perfil. Essa melhoria traria uma usabilidade melhor para o usuário e agregaria mais valor comercial ao produto, com a possibilidade de propaganda direcionada.

Outra possibilidade de implementação técnica que geraria benefícios para a plataforma é o uso de *machine learning* para auxiliar os usuários a escolher a categoria. Essa implementação ajudaria a facilitar o uso da plataforma por usuários novos que ainda não estão familiarizados com o conceito das subculturas. A ideia seria exibir, no momento da criação de uma nova postagem, uma sugestão de qual categoria que representa subculturas, aquela postagem se enquadra mais.

Uma frente de atuação necessária para o crescimento da plataforma seria o uso de plataformas que possibilitem escalonamento rápido. As ferramentas que mais se

adequam a esse contexto são as disponibilizadas pela plataforma *Amazon Web Services*. Esses serviços possibilitam facilmente o escalonamento de acordo com o valor contratado.

As imagens poderiam ser armazenadas em *Buckets* do *Amazon S3* e para o banco de dados poderia ser utilizado o *DynamoDB*. Além disso, a plataforma também fornece serviço de *Gateway* para *Apis*. Também existe a possibilidade de monitorar, implantar, gerenciar e escalar aplicações escaláveis pelo *Amazon Elastic Container Service*.

Outro aspecto que necessita de melhorias é a experiência de usuário e o visual da plataforma, visto que a mesma foi implementada sem os conhecimentos necessários nesta área. A forma mais adequada de melhorar esse aspecto seria através da contratação de um profissional da área de design com conhecimento em experiência do usuário.

5.3.2. Análises

Um possível trabalho futuro posterior ao uso do Costylist por uma gama suficiente de usuários seria o estudo do impacto de uma rede social com esse aspecto na organização social. Como foi abordado ao longo do texto, a moda tem impacto na interação dentro de subculturas e afirmação pessoal. Um estudo dos impactos de uma rede social com foco nesse aspecto traria mais lucidez a esse potencial previsto.

Um aspecto que deve ser analisado é a vulnerabilidade do sistema, que necessita de uma análise de formas de como mitigar essa vulnerabilidade. O ponto de preocupação, é a possibilidade de inserir qualquer link na criação das postagens, inclusive links fraudulentos. Seria necessário ponderar as diferentes formas de validar os links inseridos e de como manter uma lista de domínios inseridos de maneira segura, sem possibilitar inserções de links contendo *malwares* ou golpes.

Outro ponto ainda sem definição é sobre qual seria a melhor estratégia de monetização para tornar a plataforma sustentável. O potencial de uso para propagandas direcionadas é evidente, e pode se tornar valioso para, por exemplo, marcas de roupas. Apesar de evidente, a forma de captação de possíveis anunciantes e de inserção dos anúncios ainda precisa ser analisada.

Referências Bibliográficas

SANT'ANNA, Mara. Teoria de moda: Sociedade, imagem e consumo. 2 Ed. São Paulo: estação das letras e cores, 2009.

MOURA, Larissa Leal. Moda como expressão de identidade no mundo contemporâneo. 2018. 97 f. Dissertação (Mestrado em Psicologia Social) - Universidade Federal de Sergipe, São Cristóvão, SE, 2018

AGUIAR, Titta. Personal stylist: Guia para consultores de imagem. 2 Ed. São Paulo: Editora Senac, 2004.

TURNER, Bryan S. (Ed.). *The Cambridge dictionary of sociology*. Cambridge University Press, 2006.

CAETANO, Stella Mendonça. Indumentária, pertencimento e diferenciação: o papel das roupas na construção de uma identidade coletiva gótica. *Revista Ensaios*, v. 16, p. 176-192, jan./jun. 2020.

OLIVEIRA, José Reinaldo. Juventude e ciberespaço: implicações do uso da internet na constituição da sociabilidade juvenil. 2012. 98 f. Dissertação (Mestrado em Educação) — Programa de Pós-Graduação Stricto Sensu em Educação, Universidade Católica de Brasília, Brasília, 2012.

SHIMIZU, R.; SAITO, Y.; MATSUTANI, M.; GOTO, M. Fashion intelligence system: an outfit interpretation utilizing images and rich abstract tags. *Expert Systems With Applications*, v. 213, p. 119167, 2023.

SANTOS, F. Whering: conheça o app que promete ajudar a planejar seus looks. Terra, 4 ago. 2024. Disponível em: <https://www.terra.com.br/vida-e-estilo/whering-conheca-app-que-promete-ajudar-a-planejar-seus-looks,e43b7624a11c155c2fed48bd691509f9uwetdb8g.html>. Acesso em: 4 ago. 2024.

PINTEREST. Help Center. Disponível em: <https://help.pinterest.com/en>. Acesso em: 4 ago. 2024.

AMAZON WEB SERVICES. O que é JavaScript? Disponível em: https://aws.amazon.com/pt/what-is/javascript/. Acesso em: 21 set. 2024.

TYPESCRIPT. TypeScript: From Scratch. Disponível em: https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html. Acesso em: 21 set. 2024.

NODE.JS. Sobre o Node.js. Disponível em: https://nodejs.org/pt/about. Acesso em: 21 set. 2024.

NESTJS. Documentação do NestJS. Disponível em: https://docs.nestjs.com/. Acesso em: 21 set. 2024.

REACT. React Documentation: Learn React. Disponível em: https://react.dev/learn. Acesso em: 21 set. 2024.

JEST. Documentação do Jest. Disponível em: https://jestjs.io/pt-BR/. Acesso em: 21 set. 2024.

GOOGLE. Perguntas frequentes sobre o Gemini. Disponível em: https://gemini.google.com/faq?hl=pt-BR. Acesso em: 21 set. 2024.

BOOTSTRAP. Bootstrap. Disponível em: https://getbootstrap.com/. Acesso em: 22 set. 2024.

PASSPORT. Passport.js. Disponível em: https://www.passportjs.org/. Acesso em: 22 set. 2024.

IMGUR INC. *Imgur*. Disponível em: https://imgurinc.com/. Acesso em: 29 set. 2024.