

UNIVERSIDADE FEDERAL DE MINAS GERAIS
PROJETO ORIENTADO EM COMPUTAÇÃO I

DIEGO PEREIRA DA SILVA

Álbum de figurinhas virtual dos objetos de Messier

Tipo: tecnológico

Orientador: Mário Sérgio Alvim

Belo Horizonte

2024

RESUMO

A astronomia é uma das ciências mais antigas da humanidade. A curiosidade acerca das estrelas e outros objetos astronômicos é observada mesmo em culturas pré-históricas. No entanto, mesmo após inúmeros avanços tecnológicos que permitiram ampliar muito o conhecimento acerca do universo, muitas pessoas ainda possuem um entendimento muito raso sobre o céu e os corpos celestes.

Neste projeto, o objetivo é criar um aplicativo que incentive o aprendizado de astronomia de forma lúdica, por meio de um álbum de figurinhas virtual em que os cromos são recebidos ao se responder perguntas sobre astronomia.

O álbum conta no total com 110 figurinhas correspondentes aos objetos do catálogo de Messier, uma lista de objetos observáveis do céu que poderiam ser confundidos com cometas, elaborada pelo astrônomo Charles Messier e publicada em 1774. Apesar da proposta inicial de servir como alerta àqueles que buscavam por novos cometas, ao passar do tempo, os objetos de Messier ganharam relevância por si próprios, pois incluem uma série de galáxias, nebulosas e aglomerados muito interessantes.

No aplicativo, o usuário deverá ser capaz de criar e acessar a sua conta, responder perguntas de astronomia para ganhar pacotes de figurinhas, então abrir esses pacotes e colar as figurinhas em seu álbum. Os usuários deverão também ser capazes de trocar figurinhas entre si, caso dois usuários se conheçam e desejem realizar essa troca.

Desse modo, o aplicativo apresenta um desafio que incentiva a busca de conhecimento em astronomia para responder as perguntas e ganhar novas figurinhas, além de promover a interação entre seus usuários através da troca de figurinhas.

Palavras-chave: Astronomia, Aplicativo, Álbum de figurinhas, Objetos de Messier, Charles Messier.

ABSTRACT

Astronomy is one of the oldest sciences in the history of humanity. Even prehistoric cultures were curious about stars and other astronomical objects. However, even after numerous technological advances that have greatly expanded the knowledge about the universe, many people still have a shallow understanding of the sky and celestial bodies.

In this project, the objective is to create a mobile application that encourages learning about astronomy in a playful way, through a virtual sticker album in which stickers are received when answering questions about astronomy.

The album has a total of 110 stickers corresponding to objects from Messier's catalogue, a list of observable objects in the sky that could be confused with comets, drawn up by astronomer Charles Messier and published in 1774.

Despite the initial proposal of serve as a warning to those looking for new comets, over time, Messier objects gained relevance on their own right, as they include a series of interesting galaxies, nebulae and clusters.

In the app, the user should be able to create and access their account, answer astronomy questions to win sticker packs, then open these packs and glue them into their album. Users should also be able to exchange stickers with one another, if two users know each other and wish to exchange stickers.

In this way, the application presents a challenge that encourages the search for knowledge in astronomy to answer questions and win new stickers, in addition to promoting interactions between its users through the exchange of stickers.

Keywords: Astronomy, Mobile application, Sticker album, Messier objects, Charles Messier

LISTA DE FIGURAS

Figura 1: Telas de autenticação e de registro	10
Figura 2: Fluxo de verificação de e-mail	11
Figura 3: Fluxo de redefinição de senha	11
Figura 4: Barra de navegação com a página de álbum (ícone de livro) selecionado	12
Figura 5: Telas do álbum de figurinhas, com figurinhas coladas de diferentes raridades e espaços vazios	12
Figura 6: Fluxo de abertura de pacote, ao clicar em um pacote, é possível visualizar mais detalhes, e ao abri-lo, revelar as figurinhas obtidas	13
Figura 7: Exemplos de visualização de figurinhas de um pacote, uma carta já possuída e uma carta nova	14
Figura 8: Fluxo de recebimento de pacotes	15
Figura 9: Esquema de funcionamento dos refresh tokens. Extraído de [3]	16
Figura 10: Diagrama entidade-relacionamento do banco de dados do sistema	17
Figura 11: Diretório fonte do backend e esquema de camadas da estrutura limpa extraída de [1]	18
Figura 12: Execução dos testes automatizados e cobertura	19

LISTA DE SIGLAS

1. POC – Projeto Orientado em Computação, matéria obrigatória do curso de Ciência da Computação e fonte deste trabalho
2. OBA – Olimpíada Brasileira de Astronomia
3. FIFA – Federação Internacional de Futebol
4. TA – Token de autorização
5. RT – Refresh Token
6. JWT – JSON Web Token
7. ORM – Mapeamento Objeto-Relacional

SUMÁRIO

1 Introdução	7
2 Referencial Teórico	8
3 Metodologia	8
4 Proposta	9
5 Interface de usuário	10
5.1 Fluxo de autenticação e registro	10
5.2 Barra de navegação	12
5.3 Álbum de figurinhas	12
5.4 Inventário	13
5.5 Recebimento de pacotes	14
6 Arquitetura	15
6.1 Repositórios e Sistema de Controle de Versões	15
6.2 Autenticação e Autorização	16
6.3 Banco de dados	17
6.4 Arquitetura Limpa	18
6.5 Internacionalização	18
6.6 Testes automatizados	19
7 Conclusão	20
Referências	20

1 Introdução

O objetivo do projeto é promover o interesse pela Astronomia de forma lúdica, pelo desenvolvimento de um álbum de figurinhas virtual, disponível como um aplicativo móvel, no qual cada figurinha corresponde a um dos objetos astronômicos do catálogo de Messier.

Este catálogo foi compilado pelo astrônomo Charles Joseph Messier, que tinha o propósito de listar objetos observáveis no céu que podem ser confundidos por cometas, embora não o sejam. Posteriormente, a compilação ganhou relevância e fama pelos objetos em si, os quais incluem uma série de aglomerados, galáxias e nebulosas, tornando-se muito mais do que um auxílio para a descoberta de cometas.

Além disso, o aplicativo visa contribuir com a aprendizagem de conceitos básicos da astronomia, através de um quiz, no qual o usuário deverá responder para ganhar os pacotes de figurinhas. As perguntas a serem exibidas no quiz serão extraídas das provas da Olimpíada Brasileira de Astronomia (OBA), que mantém o acervo de exames de suas edições disponível em seu site. Portanto, a aplicação poderá também ser utilizada como ferramenta de prática pelos competidores da olimpíada.

Para além dos objetivos relacionados à funcionalidade do aplicativo, pretende-se desenvolver um código que siga bons padrões de projeto a fim de garantir a qualidade do software e sua manutenibilidade. Ademais, foi escolhida a abordagem de desenvolvimento ágil, que prevê uma rápida entrega de valor.

Como objetivos para o POC II, definiu-se a implementação do quiz para o recebimento de pacotes, além da possibilidade de adicionar amigos e trocar figurinhas, bem como o objetivo final de publicar o aplicativo em uma ou mais lojas como a App Store ou Google Play.

A seguir destaca-se uma lista com os objetivos separados em uma categoria de impactos relacionados à Astronomia e uma categoria de objetivos relacionados à Engenharia de Software, com indicação sobre ao final de qual semestre (POC I ou POC II) pretende-se cumprir tal objetivo:

Objetivos relacionados ao impacto à Astronomia:

- Promover o interesse pela Astronomia (POC I)
- Contribuir para o aprendizado de conceitos fundamentais de Astronomia (POC I)
- Melhorar o entendimento sobre o céu e como ele é dividido (POC I)
- Servir como ferramenta de treino para a OBA nos níveis básicos (POC I)
- Promover interação entre os usuários do aplicativo e criação de grupos com o interesse em Astronomia (POC II)

Objetivos relacionados à Engenharia de Software:

- Aplicar bons padrões de projeto e melhores práticas de desenvolvimento (POC I)
- Seguir uma abordagem de desenvolvimento ágil (POC I)
- Utilizar ferramentas modernas de desenvolvimento (POC I)
- Publicar um aplicativo funcional para acesso ao público (POC II)

2 Referencial Teórico

A principal obra a ser utilizada como guia no que diz respeito às práticas adotadas durante o desenvolvimento é o livro **Engenharia de Software Moderna** [1], que contém explicações sobre o desenvolvimento ágil, princípios e padrões de projeto, arquitetura, testes, entre outros. A título de planejamento, o conceito utilizado é o de desenvolvimento ágil, cuja principal característica, segundo [1], é a adoção de ciclos curtos e iterativos de desenvolvimento, que serão detalhados na seção 5.

Além desta referência, é importante citar as fontes de inspiração para este projeto, neste caso ferramentas parecidas que executam parte da proposta do trabalho. Em primeiro lugar, é possível citar o aplicativo do álbum de figurinhas da Copa do Mundo FIFA, justamente uma versão virtual para o popular álbum de figurinhas dos jogadores do torneio, as funcionalidades do álbum como visualização, abertura de pacotes e colagem de figurinhas serão adotadas também neste projeto. Outra inspiração é um aplicativo chamado “Simulado OBA”, que consiste em um compilado de perguntas de edições passadas da olimpíada de astronomia apresentadas em formato de quis. Na proposta deste trabalho, esse tipo de quiz será o desafio a ser cumprido para obtenção dos pacotes de figurinhas. O diferencial do projeto consiste na junção destas duas ideias, além da distinta temática dos objetos de Messier.

3 Metodologia

O desenvolvimento do projeto será baseado no Manifesto Ágil [2], com foco no software em funcionamento e rápida resposta a mudanças. Por isso, serão definidas iterações (ou sprints), separando as funcionalidades mais importantes para serem implementadas de imediato e já validadas antes de iniciar a próxima iteração. Vale ressaltar que não se pretende seguir um método específico, como Scrum ou Kanban, pois essas propostas são pensadas para projetos em equipe, no caso deste trabalho, apenas o manifesto ágil será utilizado como referência, por ser mais abrangente.

A seguir estão definidos os passos previstos no desenvolvimento. Contudo, muitos desses passos podem não abranger todo o escopo do projeto logo de início, mas principalmente o que está previsto para uma iteração, e poderão ser executados múltiplas vezes ao longo do projeto. Por exemplo, a modelagem da Interface do Usuário vai incluir apenas as telas de uma sprint, e as telas previstas para uma sprint futura serão modeladas apenas quando a sprint correspondente iniciar.

1. Escolha das tecnologias
2. Preparação do ambiente de desenvolvimento
3. Modelagem do banco de dados
4. Organização das estruturas dos repositórios
5. Modelagem da Interface do Usuário
6. Definição das histórias de usuário
7. Execução das tarefas
8. Validação de qualidade

4 Proposta

A proposta do projeto inclui 4 iterações, cada uma delas com requisitos de um produto que já possui valor e atende parte dos objetivos, elas estão em ordem de prioridade, ou seja, as funcionalidades da iteração 1 são as mais importantes. Ressalta-se que a metodologia ágil prevê certa adaptabilidade, de modo que o escopo de funcionalidades de cada iteração pode ser alterado ao longo do desenvolvimento, então as descrições a seguir refletem a intenção inicial, mas estão sujeitas a alterações.

Funcionalidades da Iteração 1:

- Registro de usuário
- Verificação de e-mail do usuário
- Recuperação de senha
- Autenticação de usuário
- Login e registro utilizando a conta do Google

Funcionalidades da Iteração 2:

- Visualização do álbum de figurinhas
- Abertura de pacotes
- Recebimento de até um pacote por dia mediante resgate
- Visualização dos pacotes possuídos
- Colagem de figurinhas no álbum
- Diferentes raridades das figurinhas (comum, raras, épicas e lendárias)

Funcionalidades da Iteração 3:

- Possibilidade de responder a até um quiz por dia
- Recebimento de um pacote de figurinhas de acordo com o resultado do quiz (substitui o recebimento de um pacote por dia da iteração 2)

Funcionalidades da Iteração 4:

- Solicitar uma troca de figurinhas
- Aceitar uma troca de figurinhas
- Realizar a troca de figurinhas caso ambos os usuários tenham as figurinhas da troca

Para o primeiro semestre (POC I), planeja-se a execução até a iteração 2, e para o segundo semestre (POC II), a finalização, ou seja, a execução até a iteração 4.

A proposta deste trabalho sofreu alterações em relação à proposta inicial estipulada no começo do semestre, principalmente para possibilitar que a aplicação possua um sistema de registro e autenticação robusto e seguro. Desse modo, algumas funcionalidades estipuladas como visualização de perfil e adição de amigos foram descartadas para o escopo deste projeto, o que não as impede de serem implementadas como atualização após o lançamento do app.

5 Interface de usuário

As telas da aplicação foram cuidadosamente pensadas para possibilitar um fluxo intuitivo ao usuário. Cada tela tem o seu propósito específico.

5.1 Fluxo de autenticação e registro

Ao abrir o aplicativo pela primeira vez, o usuário encontrará a tela de autenticação, onde pode inserir suas credenciais para acessar a aplicação. Além disso, existe a possibilidade de entrar utilizando uma conta do Google, que dispensa o preenchimento de dados de registro. A tela inclui ainda um botão para realizar o cadastro no aplicativo através de um endereço de e-mail qualquer, bem como um botão para recuperação de senha caso o usuário não se recorde da mesma.

A tela de criação de conta exige o preenchimento de todas as informações indispensáveis ao usuário, são elas: nome, nome de usuário, endereço de e-mail e uma senha. O formulário é bem claro e aponta erros dinamicamente, por exemplo, caso o usuário tenha digitado um e-mail já cadastrado na plataforma, uma mensagem sobre este erro será exibida e o usuário não será capaz de realizar o registro até que altere o valor deste campo para um e-mail não registrado.

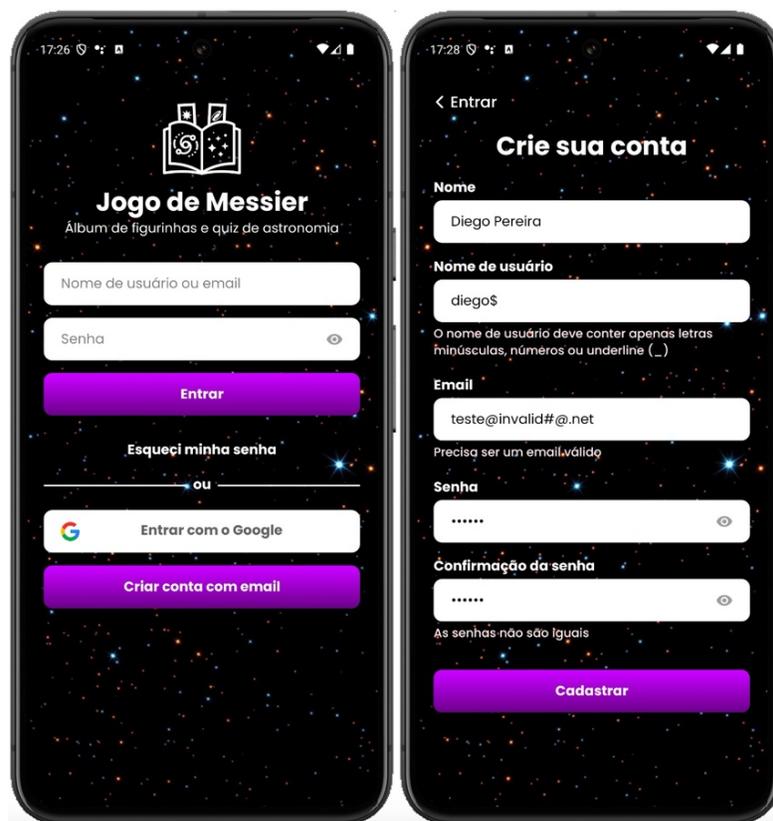


Figura 1: Telas de autenticação e de registro

Após realizar o cadastro utilizando um endereço de e-mail, o usuário deverá verificar que possui acesso àquele endereço, para confirmar que ele inseriu o endereço correto para um e-mail em que tem acesso. Esta verificação é importante para garantir que o usuário terá a possibilidade de alterar sua senha no futuro caso a tenha esquecido, uma vez que o processo de recuperação

depende do envio de um e-mail. A verificação de e-mail e redefinição de senha são feitas através de uma página da web, com um link contendo um código para validar aquela requisição.

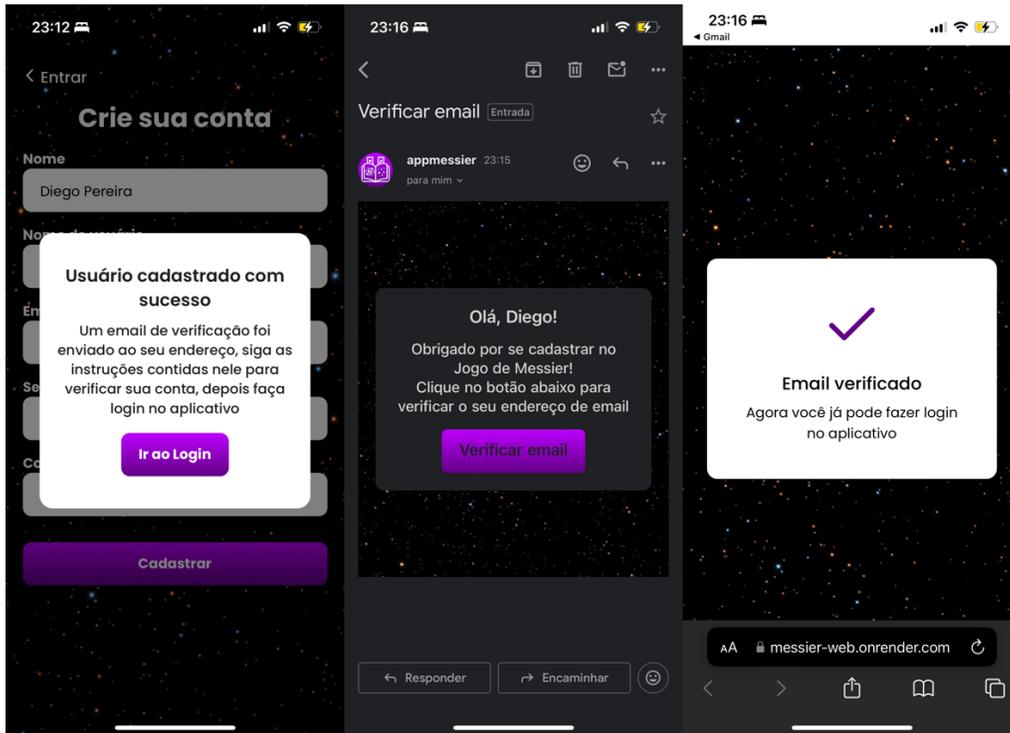


Figura 2: Fluxo de verificação de e-mail

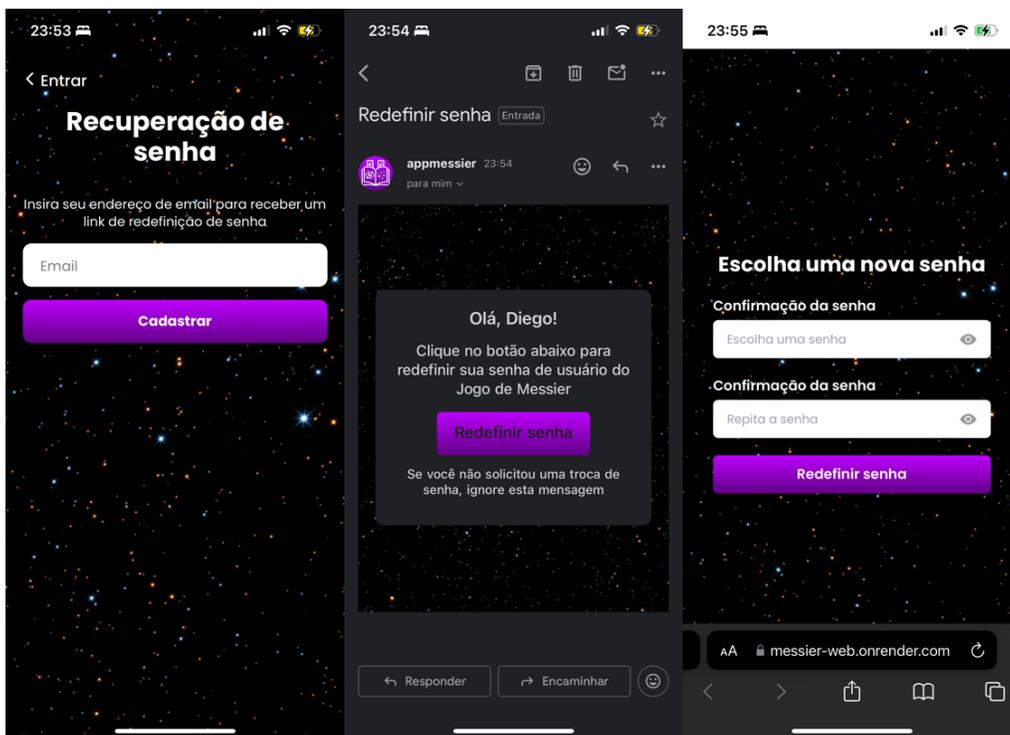


Figura 3: Fluxo de redefinição de senha

5.2 Barra de navegação

Para organizar as diferentes possibilidades do aplicativo, as telas são acessáveis através de uma barra horizontal com ícones indicativos de alguma característica daquela tela. Foram criadas 3 diferentes telas, uma para o recebimento de pacotes, representada por um ícone de presente, uma tela para o álbum de figurinhas, representada por livro aberto, e uma tela de inventário, onde o usuário pode ver e abrir seus pacotes de figurinhas.

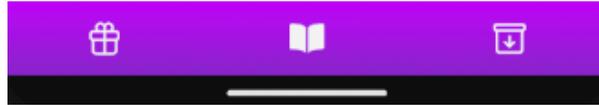


Figura 4: Barra de navegação com a página de álbum (ícone de livro) selecionado

5.3 Álbum de figurinhas

A tela de álbum de figurinhas possui diversas páginas, que podem ser trocadas pelo gesto de deslizar na horizontal. Além disso, na parte inferior existe a indicação de todas as páginas e é possível clicar na indicação de uma página para deslizar automaticamente até ela. Cada tela contém vários espaços para figurinhas, que podem estar em três estados:

1. Figurinha não colada e não possuída pelo usuário
2. Figurinha não colada, mas possuída pelo usuário
3. Figurinha colada



Figura 5: Telas do álbum de figurinhas, com figurinhas coladas de diferentes raridades e espaços vazios

No estado 1, o espaço da figurinha estará com uma forma tracejada com o número de Messier no centro. No estado 2, o espaço estará da mesma forma que no estado 1, porém colorido com um fundo banco translúcido, para indicar que o usuário possui a figurinha correspondente, de modo que basta que o usuário clique nesse espaço para colá-la no álbum. Já no estado 3, é mostrado uma versão pequena da figurinha, contendo a imagem do objeto de Messier e seu número, a cor da figurinha indica a sua raridade, as figurinhas cinzas são as comuns, as amarelas são raras, as roxas são épicas e as coloridas são lendárias, em grau crescente de raridade.

5.4 Inventário

No inventário, o usuário pode visualizar os pacotes que possui para então abri-los. Ao clicar em um pacote, aparecem quantas figurinhas há dentro do mesmo bem como as probabilidades de cada tipo de figurinha (raridade) para cada figurinha do pacote. Ao abrir um pacote, o usuário pode ver quais figurinhas ele ganhou imediatamente, com uma versão maior da figurinha em relação àquela presente no álbum. Caso não possua a figurinha sorteada no pacote, haverá uma marcação indicando que ela é nova, então o usuário sabe que poderá colá-la em seu álbum e não se trata de uma figurinha repetida.

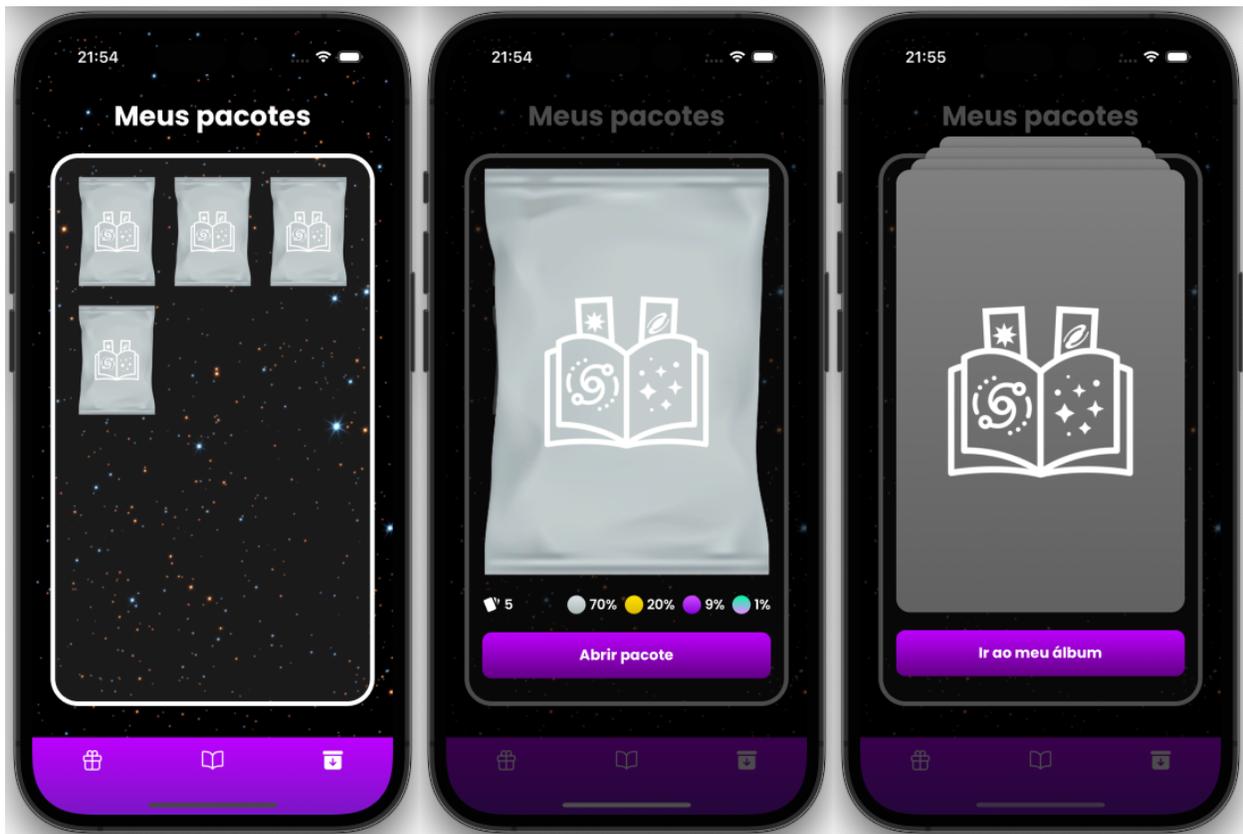


Figura 6: Fluxo de abertura de pacote, ao clicar em um pacote, é possível visualizar mais detalhes, e ao abri-lo, revelar as figurinhas obtidas

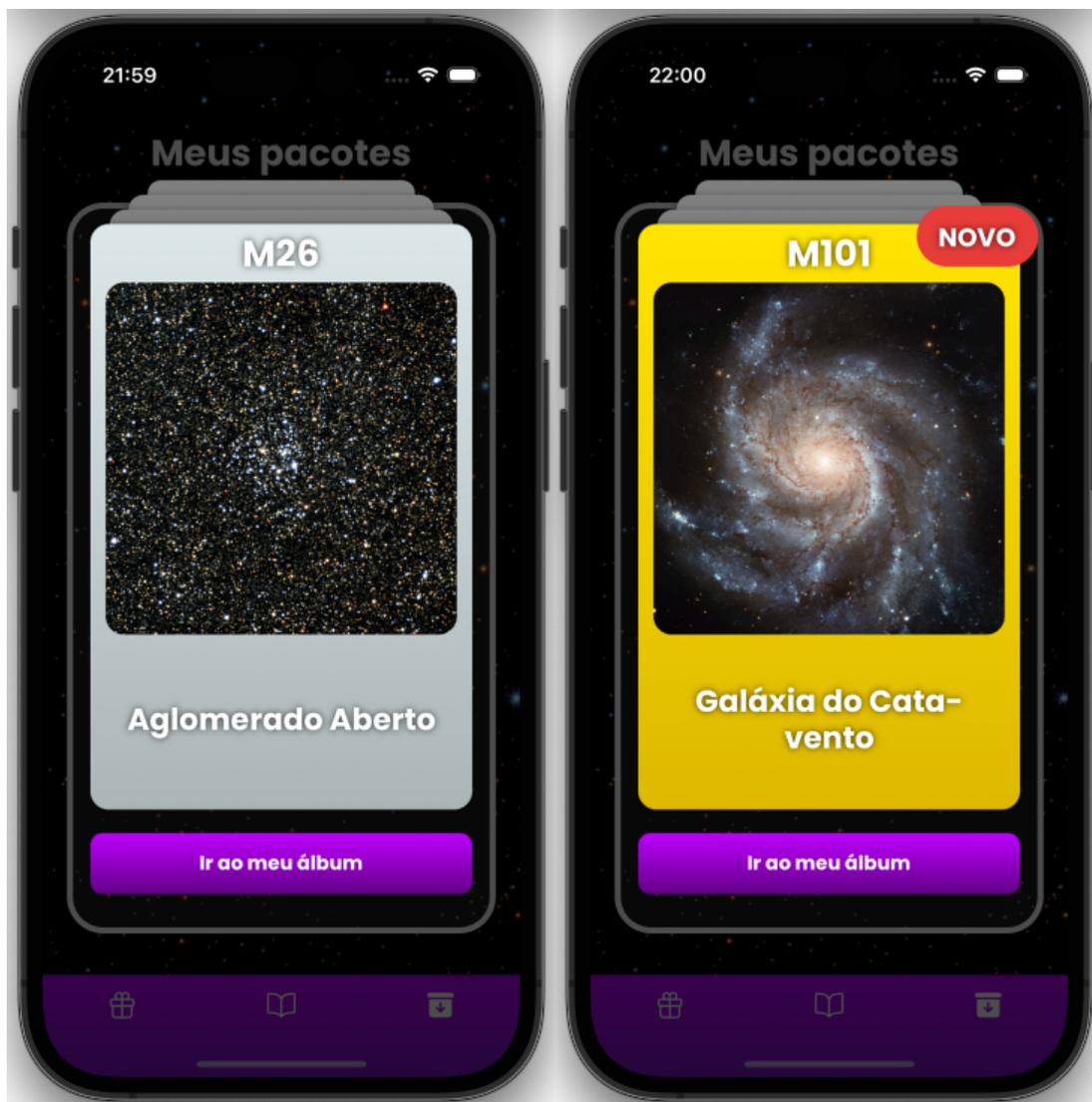


Figura 7: Exemplos de visualização de figurinhas de um pacote, uma carta já possuída e uma carta nova

5.5 Recebimento de pacotes

Ao clicar no ícone de presente, o usuário tem acesso à tela de recebimentos de pacotes. Cada usuário pode receber um pacote com 5 figurinhas a cada 24 horas. Caso o usuário possa receber um pacote, ele poderá reivindicá-lo através de um botão, caso contrário, será mostrada uma contagem regressiva em uma barra de progresso indicando o tempo até o recebimento do próximo pacote.

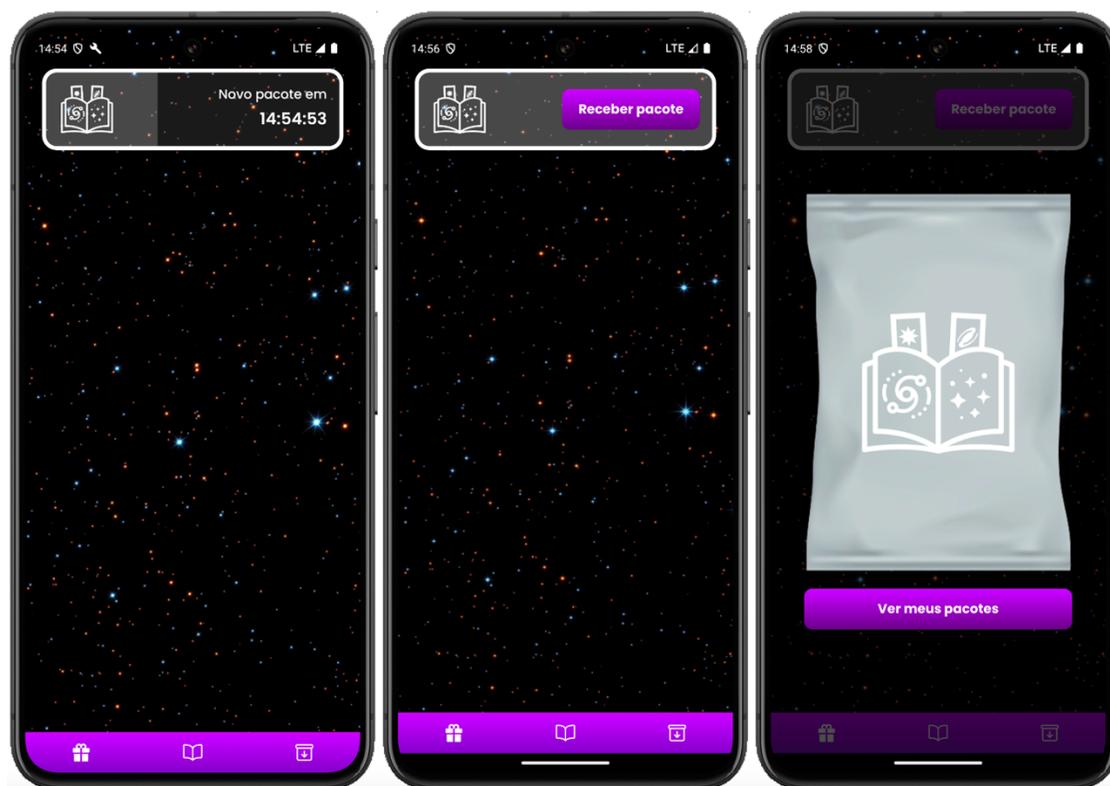


Figura 8: Fluxo de recebimento de pacotes

6 Arquitetura

Esta seção busca detalhar estrutura de desenvolvimento do aplicativo de forma mais técnica, incluindo as tecnologias utilizadas e princípios de desenvolvimento, bem como a estruturação do projeto.

6.1 Repositórios e Sistema de Controle de Versões

O projeto utiliza o Git como Sistema de Controle de Versões, que já é uma tecnologia considerada padrão para a maioria dos projetos de software. Foram criados três repositórios, um contendo o backend do projeto, ou lado do servidor, desenvolvido em Node.js, nele é implementada a API Rest da aplicação e é feita a conexão com o banco de dados, neste caso feito em PostgreSQL. Outro repositório contém o aplicativo React Native, feito através da plataforma Expo, nele é desenvolvido a aplicação móvel propriamente dita, que interage com o servidor via API. Por último existe um repositório da parte Web feito em React, este repositório é responsável pela interface web da verificação de e-mail e redefinição de senha. Cada um destes subprojetos possui seu repositório próprio e todos possuem em comum o uso da linguagem Typescript, além disso, o mesmo padrão de estilo de código é forçado para os três subprojetos utilizando uma combinação das tecnologias ESLint e Prettier. Os três estão hospedados na plataforma Github e possuem uma configuração básica de Integração contínua implementada com o Github Actions, a qual verifica se as regras de estilo estão sendo atendidas.

6.2 Autenticação e Autorização

Para promover a segurança da aplicação, optou-se por um sistema de tokens de atualização ou refresh tokens, de modo que a aplicação possui tantos tokens de autorização (TA), como refresh tokens (RT). Um token de autorização é necessário para fazer requisições protegidas ao servidor, porém eles estão sendo implementados através de JSON Web Token (JWT) e por isso possuem uma data de expiração fixa, sem possibilidade de inativação do token antes de sua expiração. Neste caso, caso alguém tenha acesso a um TA indevidamente, ele poderá utilizá-lo até sua expiração. Por isso, os tokens de autenticação possuem curta duração, e caso tenha expirado, o usuário precisará obter outro TA, através do envio de seu RT. Ao contrário os TA, os refresh tokens estão armazenados no banco de dados, e podem ser excluídos antes de sua expiração, desse modo, caso um ataque seja identificado, é possível impedir o acesso excluindo do refresh token obtido indevidamente. Devido a este maior controle, os tokens de atualização podem possuir vidas maiores, nessa aplicação, eles expiram após 30 dias. Toda vez que o usuário realiza sua autenticação, ele recebe tanto um RT como um TA, e essas informações são salvas em seu armazenamento local, a obtenção de um TA a partir de um RT e feita sem que o usuário perceba.

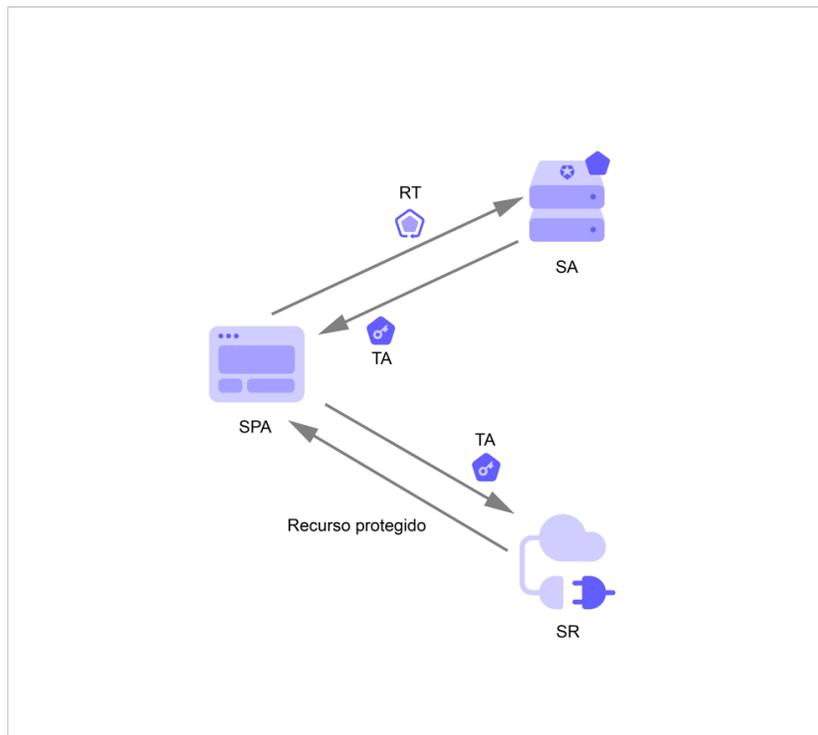


Figura 9: Esquema de funcionamento dos refresh tokens. Extraído de [3]

Como os refresh tokens duram 30 dias, os usuários seriam então requisitados a autenticarem-se novamente após este prazo, mas para evitar essa inconveniência, foi implementada a renovação dos referesh tokens, de modo que cada vez que um RT é utilizado para obter um novo TA, este RT é deletado e um novo RT é gerado e enviado ao cliente, é como se aquele RT fosse “renovado” e enquanto o usuário utilizar o sistema, fazendo requisições com TA, ele terá seu RT sempre renovado e não precisará fazer login novamente. Ele só precisará refazer sua autenticação caso passe mais de 30 dias sem utilizar a aplicação.

6.3 Banco de dados

Optou-se por uma estrutura relacional no banco de dados, por ser um tipo de modelagem já consolidado no desenvolvimento de software. Foram criadas 6 tabelas ao todo:

1. Usuário (User): Armazena informações básicas do usuário como nome e e-mail.
2. Objeto de Messier (MessierObject): Contém informações sobre um objeto de Messier.
3. Pacote (Pack): Pertence a um usuário e possui informação de quantas figurinhas possui, seu tipo (que determina a distribuição de probabilidade da raridade), além de carregar a informação se já foi ou não aberto e quando esta abertura aconteceu.
4. Figurinha (Sticker): Possui o número de Messier e de qual pacote veio, bem como qual usuário possui esta figurinha no momento.
5. Token de e-mail (EmailToken): Armazena um código que o usuário tem acesso recebendo um link por e-mail, guarda este código de forma criptografada e pode ser de dois tipos: Token de verificação de e-mail ou token de redefinição de senha. Um token possui um prazo de validade, e a informação de se já foi ou não usado pelo usuário ao qual está relacionado.
6. Token de atualização (RefreshToken): Este token pertence a um usuário é utilizado para gerar novos tokens de autorização.

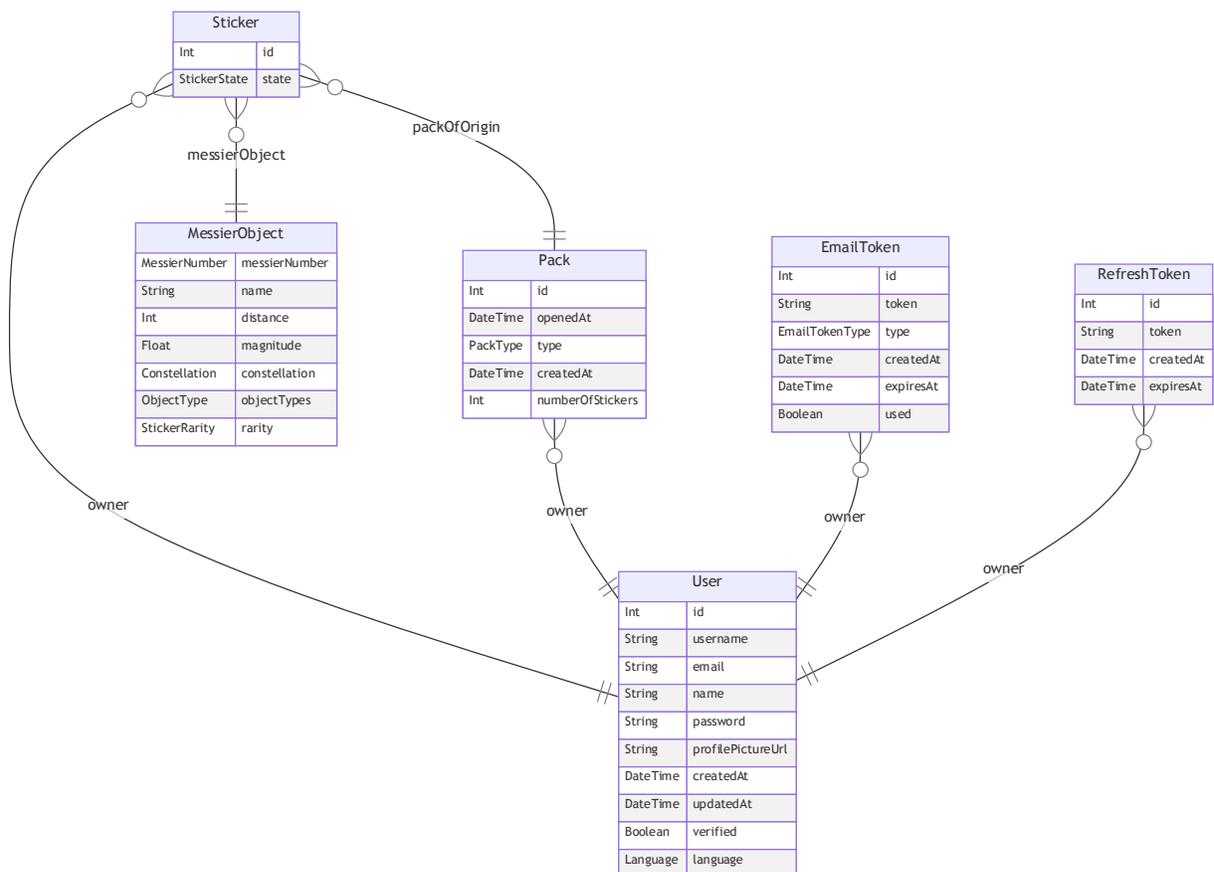


Figura 10: Diagrama entidade-relacionamento do banco de dados do sistema

6.4 Arquitetura Limpa

A estrutura de arquivos utilizada no backend busca promover uma arquitetura limpa, favorecendo a reusabilidade do código, sua coesão e uma maior independência das tecnologias adotadas, como por exemplo o ORM, sigla em inglês para Mapeamento Objeto-Relacional. Neste projeto o ORM adotado é o Prisma, que realiza as operações no banco de dados. A arquitetura limpa prevê a divisão de camadas, em que as mais centrais se relacionam ao núcleo do sistema, ou seja, suas funcionalidades primordiais e suas regras de negócio, enquanto as camadas mais externas tratam mais da implementação do sistema, com o uso de ferramentas externas como o banco de dados e a interface gráfica. A imagem a seguir mostra a estrutura do diretório fonte do lado do servidor, e as camadas da arquitetura limpa.

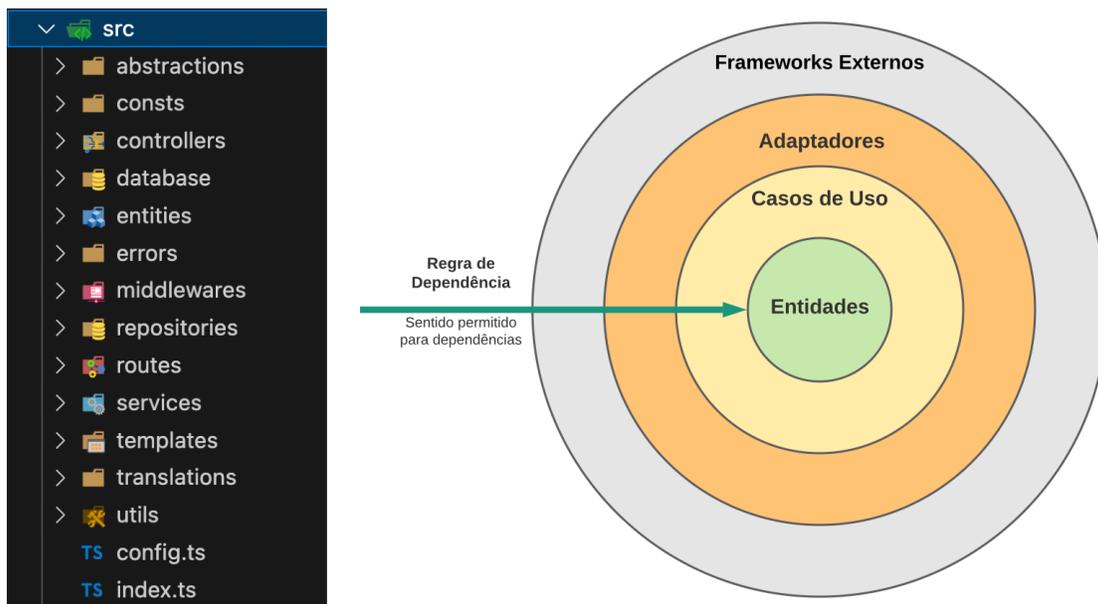


Figura 11: Diretório fonte do backend e esquema de camadas da estrutura limpa extraída de [1]

Relacionando a estrutura do diretório às camadas, podemos considerar a pasta “entities” como a camada mais central, com os elementos principais do sistema, em seguida, a pasta “services” contém os casos de uso, com implementação das regras de negócio. Dentro dos “controllers” é feita a adaptação dos serviços ao formato convencional da API, implementado através da biblioteca Express.js, enquanto a pasta “repositories” e “database” utilizam o framework do Prisma, já mencionado, para manipular o banco de dados, configurando um exemplo da camada mais externa. Observe que, caso o Prisma deixe de ser mantido, é possível substituir essa tecnologia por outra apenas alterando os diretórios correspondentes, sem interferir no resto da aplicação, favorecendo a manutenibilidade.

6.5 Internacionalização

Outro ponto importante da arquitetura do projeto é o seu suporte a múltiplos idiomas, cada repositório possui uma pasta “translations”, contendo um arquivo de tradução para cada idioma suportado. Neste momento, a ideia é suportar o português, inglês e espanhol. A separação dos textos visualizados pelos usuários em um arquivo separado é muito vantajosa, pois permite que

erros sejam localizados mais facilmente, e caso decida-se adicionar suporte a um novo idioma, basta adicionar um arquivo novo com as traduções nessa nova língua. Toda a interface apresentada até o momento está traduzida para cada um dos três idiomas mencionados. No entanto, a tradução das perguntas do quiz na próxima interação não é tão simples, pois requer que as traduções sejam armazenadas no banco de dados, além de consistir em muito mais texto, portanto, não está garantido o suporte total a qualquer idioma além do português para a aplicação completa.

6.6 Testes automatizados

Criar testes automatizados é uma boa prática e permite verificar a cada atualização no código se algum comportamento foi alterado de forma inesperada. Neste projeto, foram escritos tanto testes de unidade, para classes mais simples, como testes de integração, feitos em um banco de dados de teste. Os principais serviços estão sendo testados, como por exemplo as regras de criação de um pacote de figurinhas, abertura de pacotes, entre outros. Os testes estão inseridos dentro da estrutura do projeto próximo às respectivas implementações, ou seja, não foi criado um diretório específico com os testes, ao invés disso, dentro dos serviços é possível encontrar uma pasta chamada “PackServices” por exemplo, que contém um arquivo `index.ts` de implementação e um arquivo `index.int.test.ts` com os testes de integração correspondente. A ferramenta utilizada para escrever os testes automatizados foi o Jest, e todos os arquivos de teste, bem como a cobertura final do código pode ser conferida na imagem a seguir:

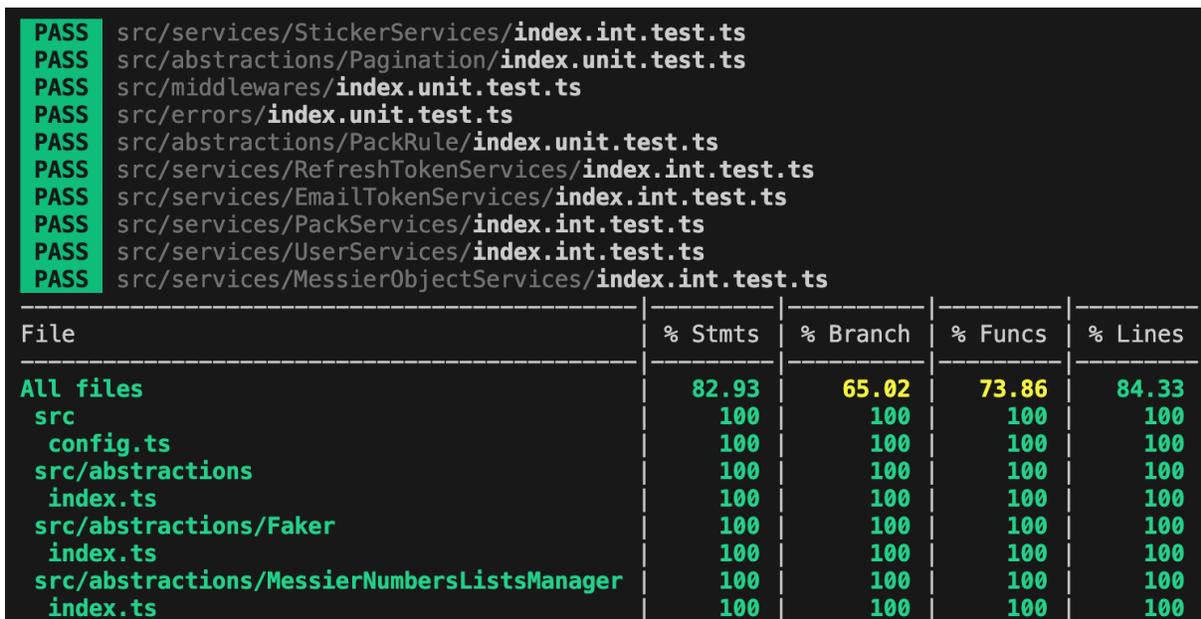


Figura 12: Execução dos testes automatizados e cobertura

Por enquanto os testes foram desenvolvidos apenas para o backend, e sua execução é feita automaticamente após cada modificação publicada no Github, através da execução do script de testes configurada no Github Actions.

7 Conclusão

Ao final do semestre, foi possível concluir os todas as funcionalidades planejadas até a iteração 2. A principal característica do aplicativo: ser um álbum de figurinhas dos objetos de Messier, já está desenvolvida e funcional. Além disso, foi possível utilizar boas práticas como a arquitetura limpa e testes automatizados na execução do projeto. Os próximos passos consistem em seguir com o desenvolvimento das próximas iterações, com as funcionalidades de quiz e troca de figurinhas a serem implementadas no POC II.

Referências

[1] Marco Tulio Valente. *Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade*. Independent edition, 2020.

[2] Manifesto para Desenvolvimento Ágil de Software. Disponível em: <<https://agilemanifesto.org/iso/ptbr/manifesto.html>>. Acesso em: 3 de abril de 2024.

[3] O que são refresh tokens e como usá-los com segurança. Disponível em: <<https://auth0.com/blog/pt-refresh-tokens-what-are-they-and-when-to-use-them/>>. Acesso em: 30 de julho de 2024.

Guilherme de Almeida. *Charles Messier e a sua obra*. Disponível em: <https://www.apaa.co.pt/GA/Charles_Messier.pdf>. Acesso em: 3 de abril de 2024.

Prisma to ER Diagram. Disponível em: <<https://prisma-erd.simonknot.de>>. Acesso em: 30 de julho de 2024.