Alunos:

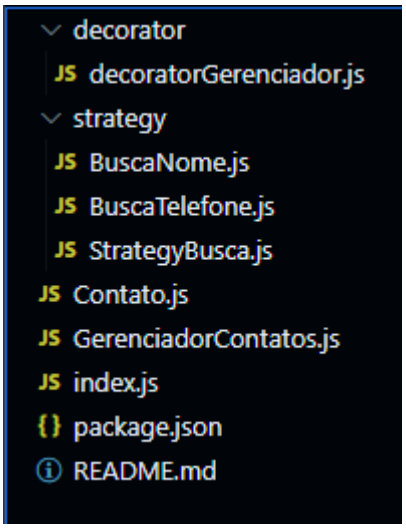
- **Marcos Vinicius Bueno Prestes - RA: 2465760**
- **Micael Ribeiro Rocha - RA: 2454424**

1) Padrões escolhidos:

- **Strategy:** Em relação ao padrão comportamental, escolhemos o Strategy por conta da possibilidade de opções que pode ser criada, como no algoritmo que foi passado para a implementação, ao usar a classe BuscaNome, o algoritmo irá retornar o usuário com tal nome que foi designado para a procura, já a classe BuscaTelefone irá retornar uma lista de todos os contatos com aquele telefone usado na busca. Portanto, esse padrão comportamental tem os benefícios de ajudar na organização do código, deixando mais intuitivo e durante a execução, o padrão Strategy permite realizar a troca das classes durante a execução do programa, tendo a possibilidade de adicionar estratégias e não alterando o contexto.

- **Decorator:** Por outro lado, para o padrão estrutural, concluímos que o mais adequado para nossa implementação seria o Decorator. Esse padrão de projeto, de forma simplificada, cria camadas de modificações sobre um determinado algoritmo sem alterar o código-fonte original.
  A implementação desse padrão no nosso projeto é realizada pela classe DecoratorGerenciador, que recebe a instância inicializada de GerenciadorContatos no index.js. Essa classe adiciona funcionalidades incrementais aos métodos existentes, garantindo assim a escalabilidade do sistema sem a necessidade de modificar o algoritmo original.

2) Estrutura do projeto:



3) Diagrama de classes: