Design Doc:
CRU Pokemon App

For simplicity both the backend and frontend will be contained in the same github. If this was a larger project I would most definitely separate the Front and Backend.

Initial Data will be generated by seeds.rb file

Project Notes:
https://arridoarfiadi.notion.site/Samaritan-Full-stack-Take-Home-Assignment-41eedfb48fd842589727efd87023232e

Backend

Database:

- There will be a join table to the pokemon and types because they have a many-many relationship. This will let us access all the types of the pokemon easily.
- Tables
    - Pokemon
        - ID
        - Name
        - Image
    - Types
        - ID
        - Name
    - Pokemon_types

How to store images?

- https://www.youtube.com/watch?v=_rLMRd676-I&ab_channel=Deanin
- Rails Active Storage

API:

Endpoints:

- Using gem rack-cors so the frontend can request data from the api (some security)
    - This can be set in config/initializers/cors.rb
    - Right now we set it to “ * ” which means anyone can access it but ideally we would set a specific address

*I didn't want to change the routes too much because that's what was specified but if it was up to me I would have added versions to the API routes (ex: /api/v1/pokemon) so you can access different versions in the future if you were to make changes

- GET /api/pokemon (example response)
    - Accepts
        - Page number
    - Returns
        - Pagy information
        - For each pokemon
            - ID
            - Name
            - Image
            - Types
                - ID
                - Name
- POST /api/pokemon/
    - Accepts
        - Name
        - Image
        - Types
- PUT /api/pokemon
    - I set it up one or many of the attributes can be included in the request and it will still work
    - Accepts
        - Name
        - Image
        - Types
            - Remove all types and re-add in case they want different order of types on the card
- GET /api/types
    - Returns
        - For each type
            - ID
            - Name

Pagination on backend

- Using the pagy gem
- https://ddnexus.github.io/pagy/how-to.html#gsc.tab=0

Frontend

- Axios for api requests
- Cards for pokemon
    - https://stackoverflow.com/questions/40853161/change-text-colour-depending-on-if-condition-react
    - Colors for cards will just be css classes that will be assigned to a card based on response.types
    - first original type decides the color
- “Edit” and “Add” accordion pop ups
    - Modal
        - https://mui.com/material-ui/react-modal/
    - Required fields
    - button that can only be selected when all required fields are filled
    - Should types be retrieved when the modal is opened or when the initial page loads?
        - Currently whenever the modal is opened the types are loaded each time but this could definitely be improved with more time
    - Type select
        - React-select library
    - edit form has prefilled textfield
        - Takes in props of clicked pokemon card
- Add new pokemon button
    - Form validation
- How many pokemon should be on one page? (pagination)
    - 20
    - https://www.npmjs.com/package/react-paginate