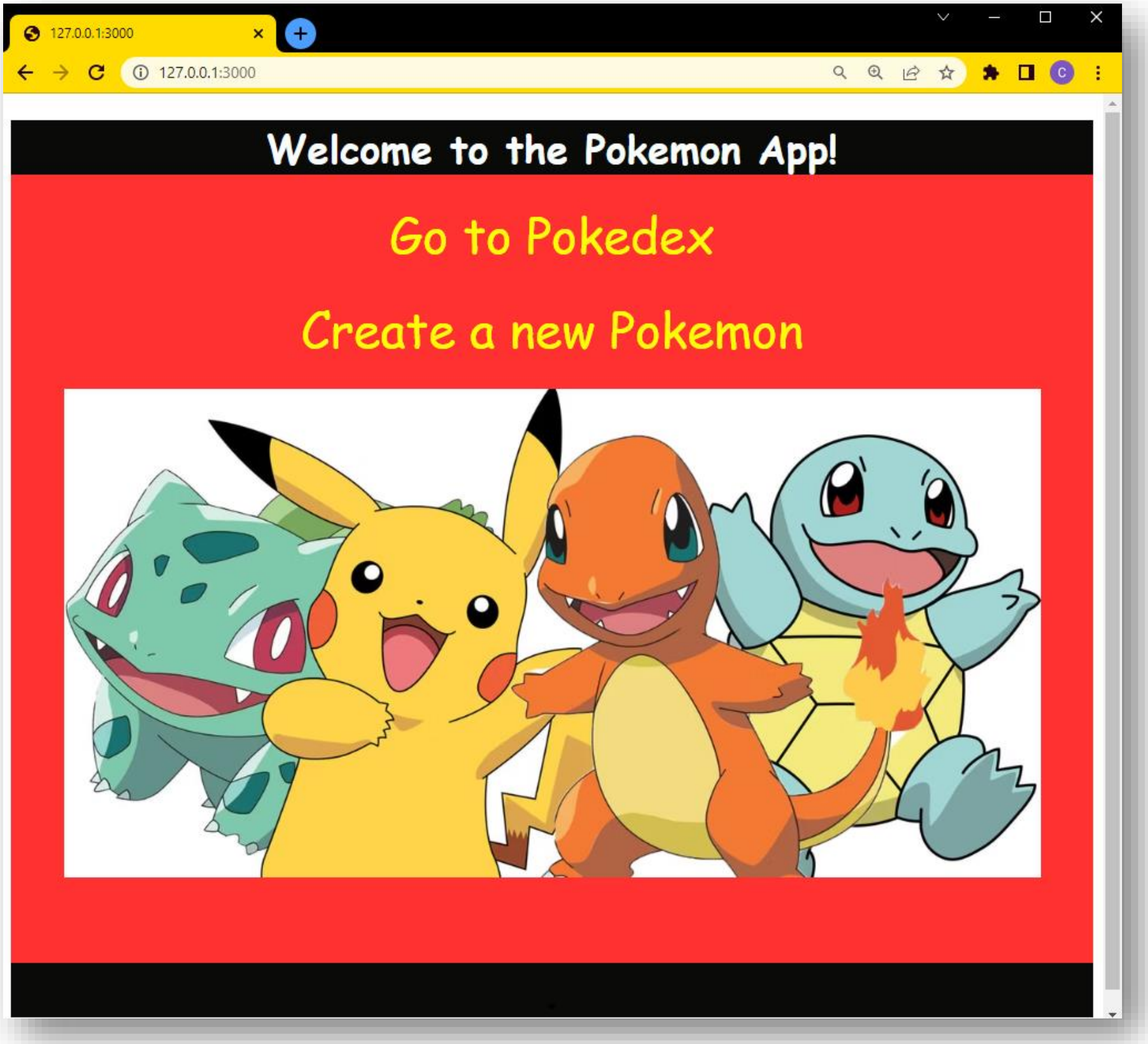
The User Stories

I created a landing page (first page) that user then have option of getting to Pokémon list (“Pokedex”) or create a Pokémon if the user knows the name and the URL address of the image that is associated with that name.



Welcome to the Pokemon App!

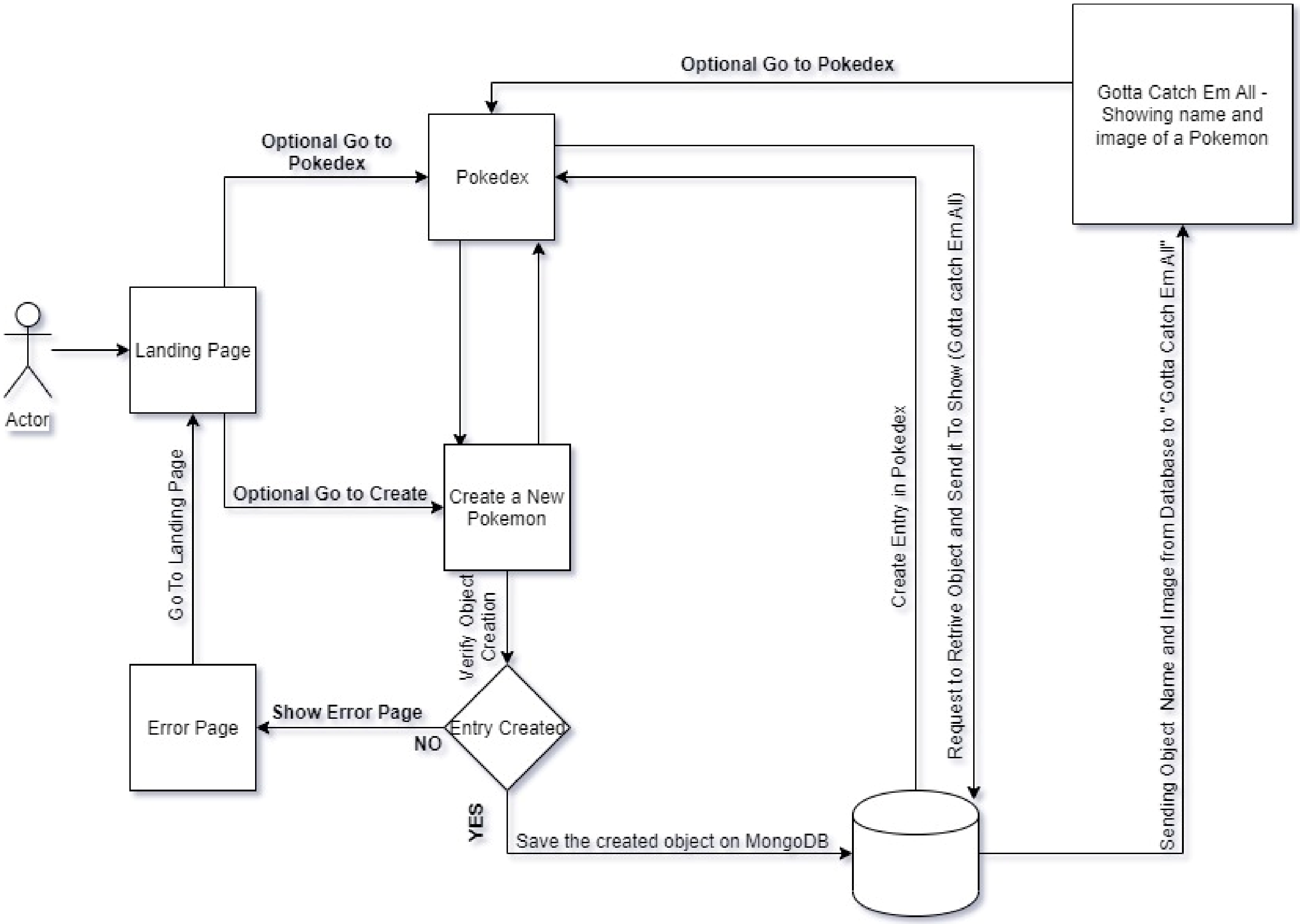
Go to Pokedex

Create a new Pokemon

This is located at the root, as you can see from Image above.

The user will go to any of these options and the have link to each other and cannot get back to this page normally. The only way to get to this page if an error detected by the code.

This can be depicted as follow:


Optional Go to Pokedex
Gotta Catch Em All - Showing name and image of a Pokemon
Optional Go to Pokedex
Pokedex
Actor
Landing Page
Go To Landing Page
Optional Go to Create
Create a New Pokemon
Verify Object Creation
Create Entry in Pokedex
Request to Retrive Object and Send it To Show (Gotta catch Em All)
Sending Object Name and Image from Database to "Gotta Catch Em All"
Error Page
Show Error Page
NO
Entry Created
YES
Save the created object on MongoDB

How did I get here:

Followings are steps that have been taken in creating this apps. The goal is to make a Pokemon app that displays data inside server-side rendered views.

Objectives
Practicing index and show, new and create routes with express

Prerequisites
JavaScript
Express
Node
JSX

When a user goes to the /pokemon route they will see an index of pokemon: the names of each pokemon rendered to the page.
When a user clicks on the name of the pokemon, they will be taken to that pokemon's showpage, and will see the pokemon's name and image.

Let's Start Catching 'em All!
When a user goes to the /pokemon route they will see an indexof pokemon: the names of each pokemon rendered to the page.

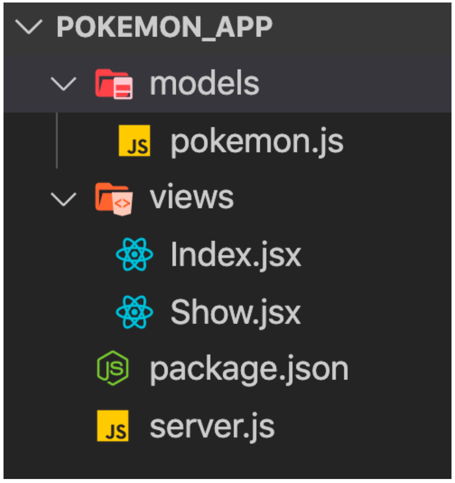
When a user clicks on the name of the pokemon, they will be taken to that pokemon's showpage, and will see the pokemon's name and image.

When a user goes to /pokemon/new a user sees a form that allows them to create a brand new pokemon, and then redirects the user back to /pokemon

Instructions to create the app:
Below We will walk you through setting up your app making your first commits and getting started with INDEX and SHOW, NEW and CREATE Will be totally up to you, you won't find any instructions below about how to set them up, it's up to you.

Use your notes from today and your fruits and vegetables app to make the New and Create Routes Correctly.
Set up the file structure In terminal:

- inside your homework folder:
- mkdir pokemon_app
- cd pokemon_app
- mkdir views
- touch views/Index.jsx
- touch views/Show.jsx
- mkdir models
- touch models/pokemon.js
- touch server.js
- touch .gitignore
- npm init -y
- Approximate File Structure

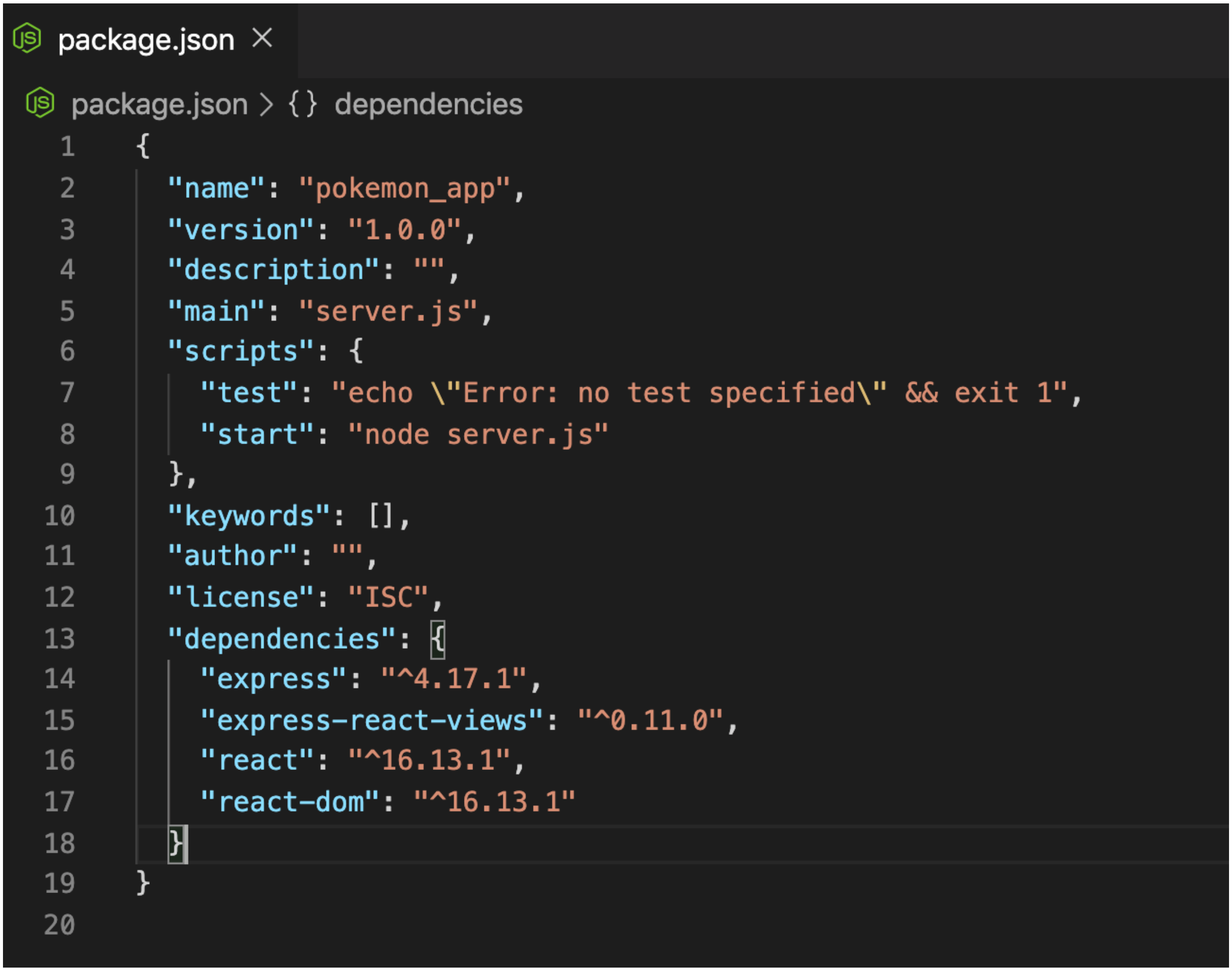


```
POKEMON_APP
  models
    pokemon.js
  views
    Index.jsx
    Show.jsx
  package.json
  server.js
```

▨ Commit:
"All my files are created"

Install NPM packages
In terminal:
Make sure that you are on the same directory level as your package.json.

- npm i express
- npm i express-react-views react@16 react-dom@16
- check your package.json
- Approximate package.json



```json
{
  "name": "pokemon_app",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.1",
    "express-react-views": "^0.11.0",
    "react": "^16.13.1",
    "react-dom": "^16.13.1"
  }
}

```

▨ Commit:
"All my npm packages are added"

Set up your server
-In your server.js file, set up your server

- require express
- set express() to a variable
- set a variable of port to 3000
- set your app to listen to the port and include a console.log(), so that you can tell when your server is running
- include a get route /that will res.send('Welcome to the Pokemon App!');so that when you got to localhost:3000, you will see Welcome to the Pokemon App!

In terminal
Nodemon or nodemon server.js (if you set your package.json to start server.jsyou don’t need to put it after nodemon)
GOTCHA! : nodemon will only work if you run it from the same location as your package.json.

In the browser
go to localhost:3000
check that you have your Welcome to the Pokemon App! message displaying

Commit:
"My server is set up and running"

Set up your *'database'*
You have created a file called pokemon.js
Inside of this file, put the following array of pokemon objects. This is your 'database' for tonight's homework, copy and paste it **as is**! You'll notice the image url's are missing something, this is *intentional*, do not edit anything directly inside the 'database'!

```
const pokemon = [
                {name: "bulbasaur", img: "http://img.pokemondb.net/artwork/bulbasaur"},
                {name: "ivysaur", img: "http://img.pokemondb.net/artwork/ivysaur"},
                {name: "venusaur", img: "http://img.pokemondb.net/artwork/venusaur"},
                {name: "charmander", img: "http://img.pokemondb.net/artwork/charmander"},
                {name: "charizard", img: "http://img.pokemondb.net/artwork/charizard"},
                {name: "squirtle", img: "http://img.pokemondb.net/artwork/squirtle"},
                {name: "wartortle", img: "http://img.pokemondb.net/artwork/wartortle"}
            ];
```

Set up your 'database' so that it can be exported to your server.js and then be required by your server.js
Set this 'database' to a variable called pokemonin your server.js file
Create a get route /pokemon that will res.send(pokemon), which will display your pokemon data as json in the browser

---

Commit:
"Connected my database, can see json in the browser"

Set up your index view
Instead of displaying json at your /pokemon route, you should serve the Index.jsx file you created that will display your pokemon

- You will have to set up your jsx file
- Start with your html boilerplate code
- Add an <h1>that describes this page, i.e. 'See All The Pokemon!'

Try adding some inline styles:
We can add inline CSS, which are specified as attributes and are passed to the elements. These are specified as an object with a key as camelCased style name & value being the actual style value (and not as a string).

```
const myStyle = {
color: '#ffffff',
backgroundColor: '#000000',
};

class MyFirstComponent extends React.Component() {
return (
  <div style={myStyle}>My First React Component!</div>;
}
}
```

Change your /pokemon route to res.renderyour Index.jsx file
In your browser, go to localhost:3000/pokemon and be sure to see your Index.jsx view with an h1tag

---

Commit:
"index.jsx view rendered at pokemon route"

Set up your index view to show your pokemon data
Continue working on your Index.jsx view so that you can see:

The name of each pokemon, as a list item, inside an unordered list
This list should be dynamically rendered by jsx based on your data from your 'database'
You'll notice the pokemon names aren't properly capitalized! Let's fix that! Manipulate the data **programmatically** to capitalize the first letter of their names

---

Commit:
"I can view a list of all my pokemon in the browser"

Setting up your show route
Inside your server.js, add a new get route /pokemon/:id
That will res.send(req.params.id);
So, when you go to localhost:3000/pokemon/whatever
whateverwill show up in the browser

Commit:
"Show view shows req.params.id "

Link your Index.jsx to your Show.jsx
Inside your Index.jsx,
For each pokemon, add an <a>tag that will have an href that goes to the route /pokemon/x, where x is the array position of the pokemon in the data array. This should be set dynamically with jsx
When you click the link, you should go to your show route and the index number corresponding to the pokemon's array position should be displayed

---

Commit:
"Added dynamic anchor tags to index.jsx "

Render your individual pokemon in the show view

- Copy/paste your code from your Index.jsx into your show.jsx`
- Change all your html code inside your Show.jsx file so that
- Your h1 tag says "Gotta Catch 'Em All"
- Add an h2 tag that will display the name of the pokemon
- Add an image tag that will display an image of the pokemon
- Add an anchor tag with the text of back, that will take you back to your Index.jsx view
- Update the route in the server.js to render the show view with the pokemon data

Oh no! The image is broken because in our database the image links don't have the .jpg ending, let's fix that **programatically**! *Without* going back to the database and editing it there, add on .jpg to the end of the pokemon's image data

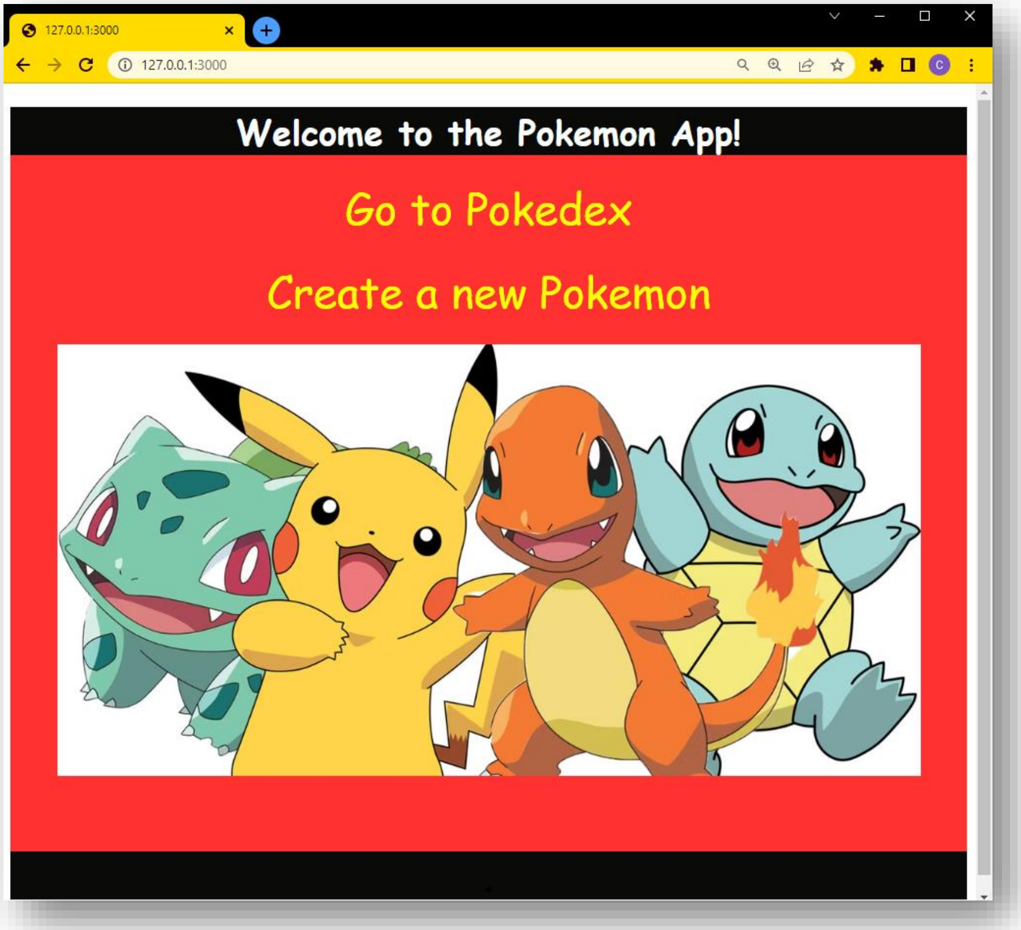
---

Commit:
"Created show views of each pokemon"

Let’s start again

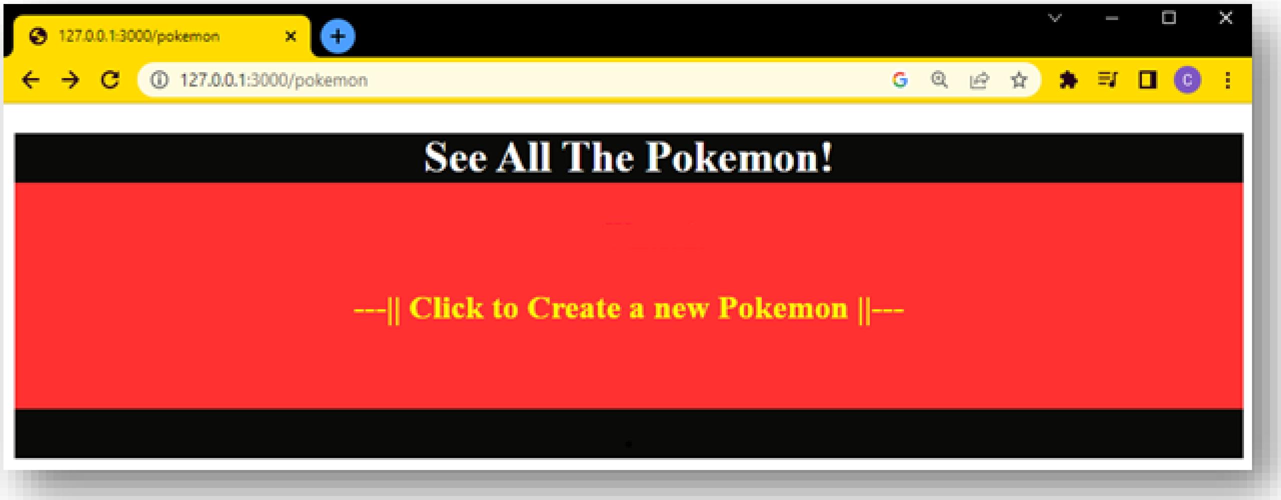
In Landing page, we have two options:

- Go to Pokedex
- Create a new Pokemon



Welcome to the Pokemon App!

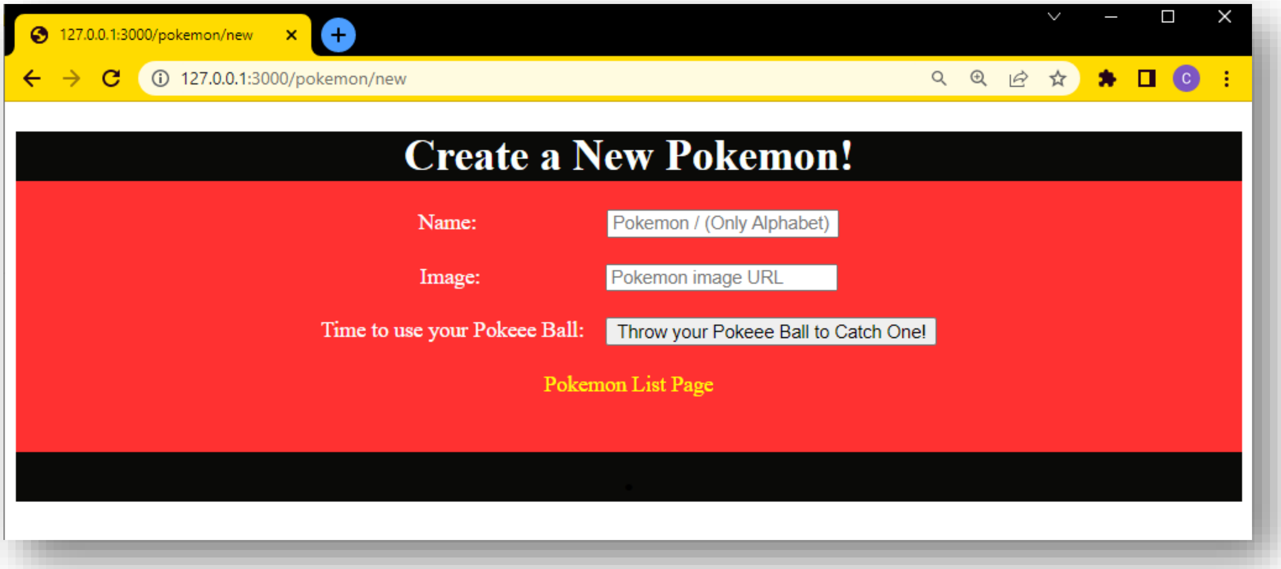
If we click on “Go to Pokedex” it shows an empty list originally.



See All The Pokemon!

---|| Click to Create a new Pokemon ||---

If we click on “Click to Create a new Pokemon” it will take user to “Create a New Pokemon.”



Create a New Pokemon!

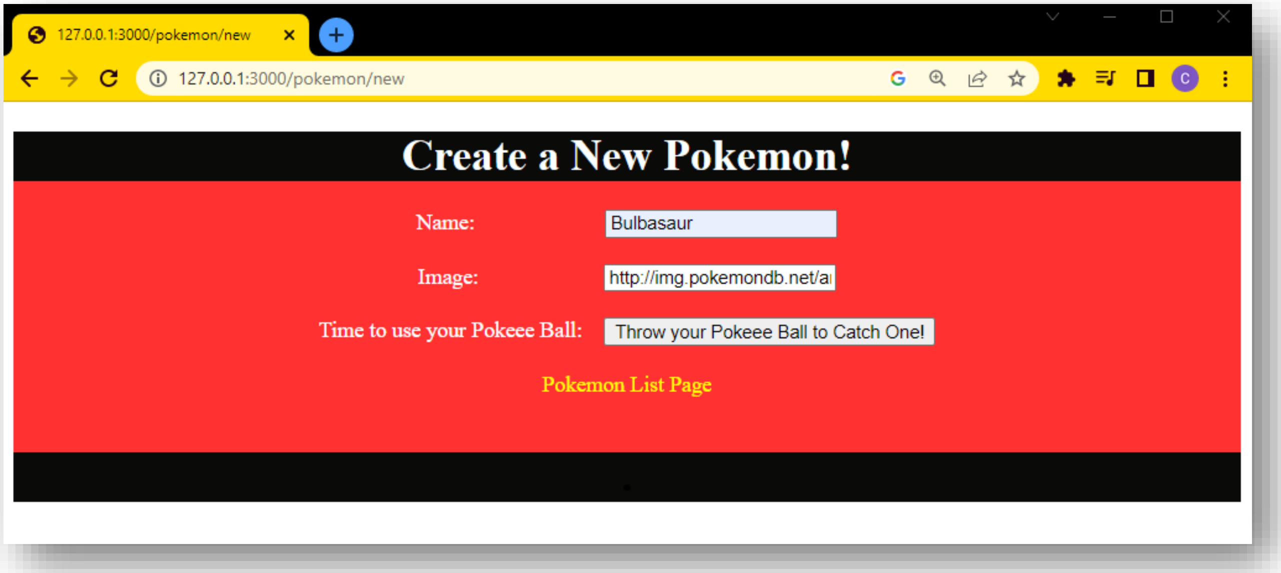
Name: Pokemon / (Only Alphabet)

Image: Pokemon image URL

Time to use your Pokeee Ball: Throw your Pokeee Ball to Catch One!

Pokemon List Page

I have set each text box to give a short instruction on what to type in. If user knows the actual name and URL:



Create a New Pokemon!

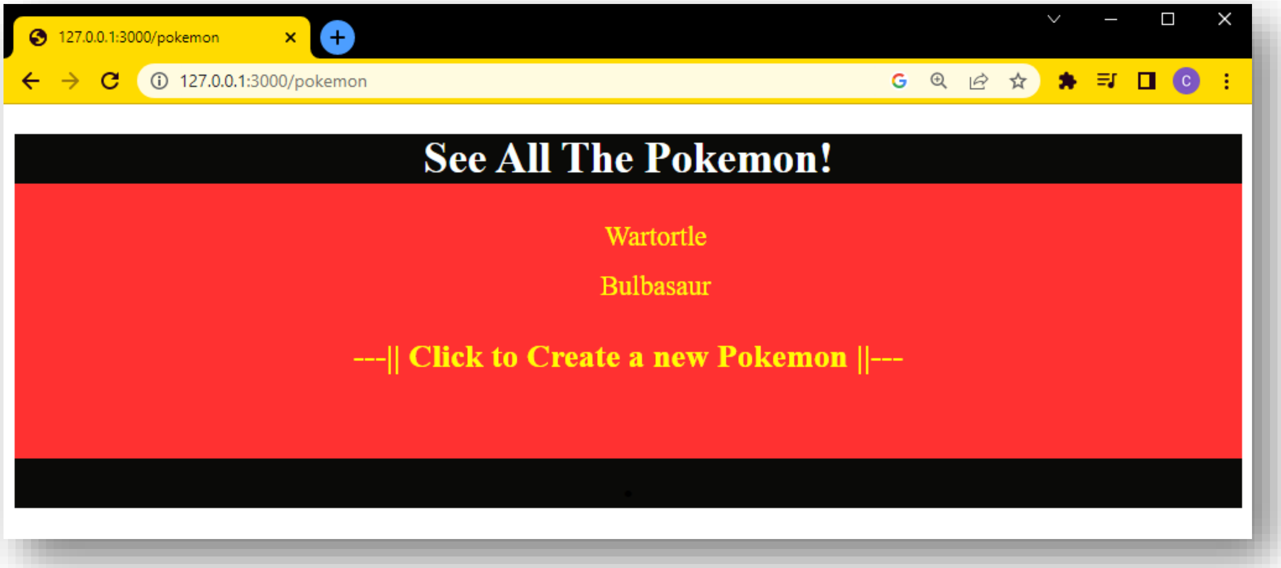
Name: Bulbasaur

Image: http://img.pokemondb.net/a

Time to use your Pokeee Ball: Throw your Pokeee Ball to Catch One!

Pokemon List Page

After Clicking on the button ("Throw your Pokee ball to Catch One"), it will create a record for object in MongoDB with the name and associated image. This will also make a name entry in Pokedex since it refreshes itself with data on the server next time it loads.



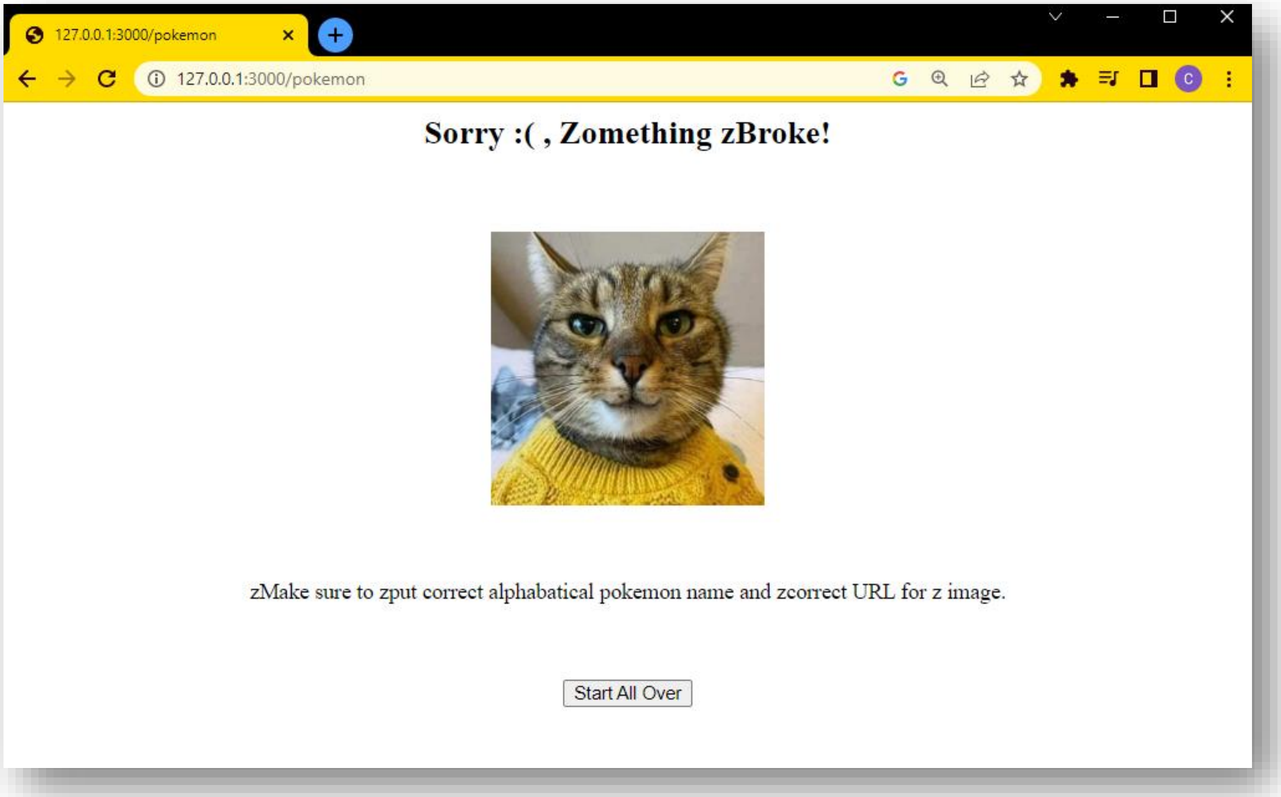
See All The Pokemon!

Wartortle

Bulbasaur

---|| Click to Create a new Pokemon ||---

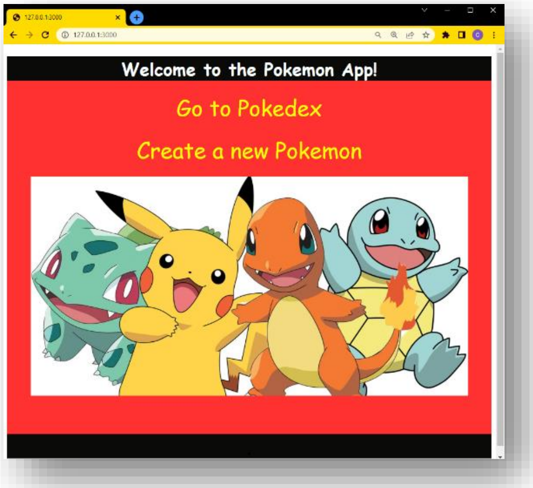
If there is an error, it opens the error message page and give the option to start all over again.



Sorry :(, Zomething zBroke!

zMake sure to zput correct alphabatical pokemon name and zcorrect URL for z image.

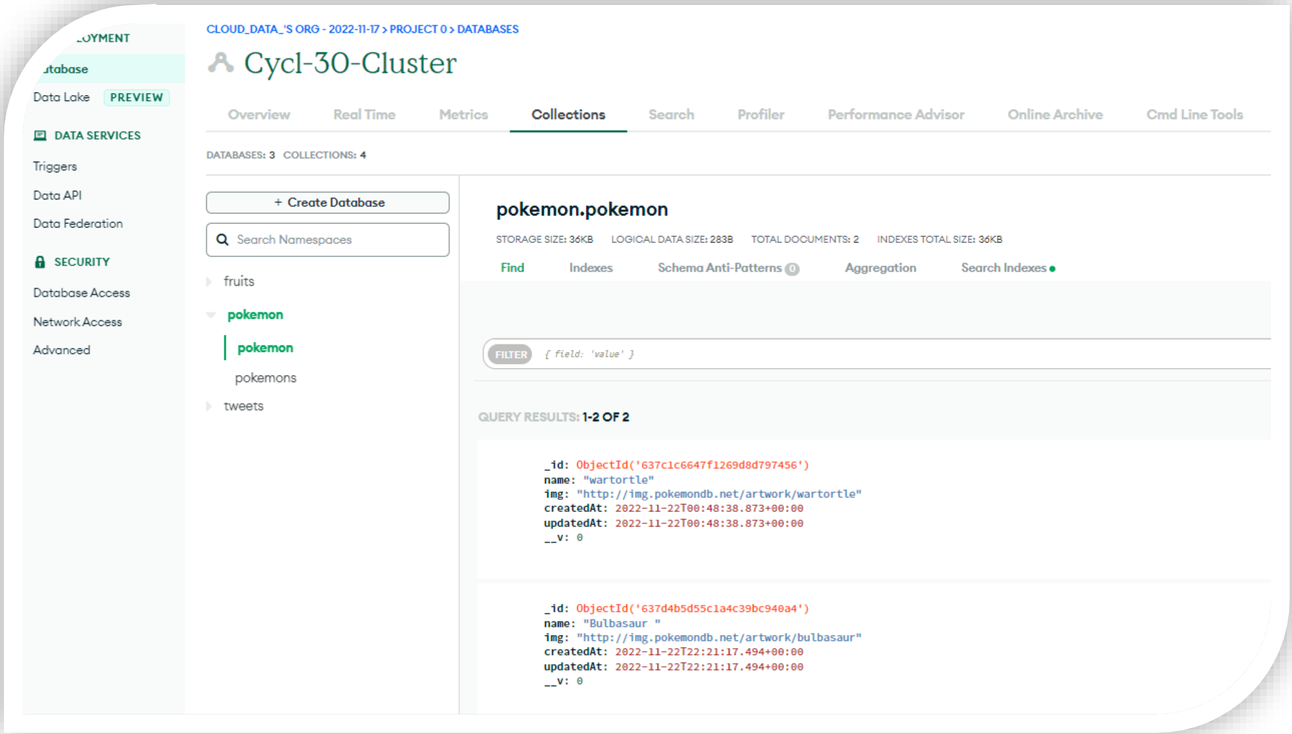
“Start over” button takes it to the landing page:



Welcome to the Pokemon App!

And user can pick any of two options available.

When the object created database will have entries as follow (for above two objects):



Cycl-30-Cluster

DATABASES: 3 COLLECTIONS: 4

pokemon.pokemon

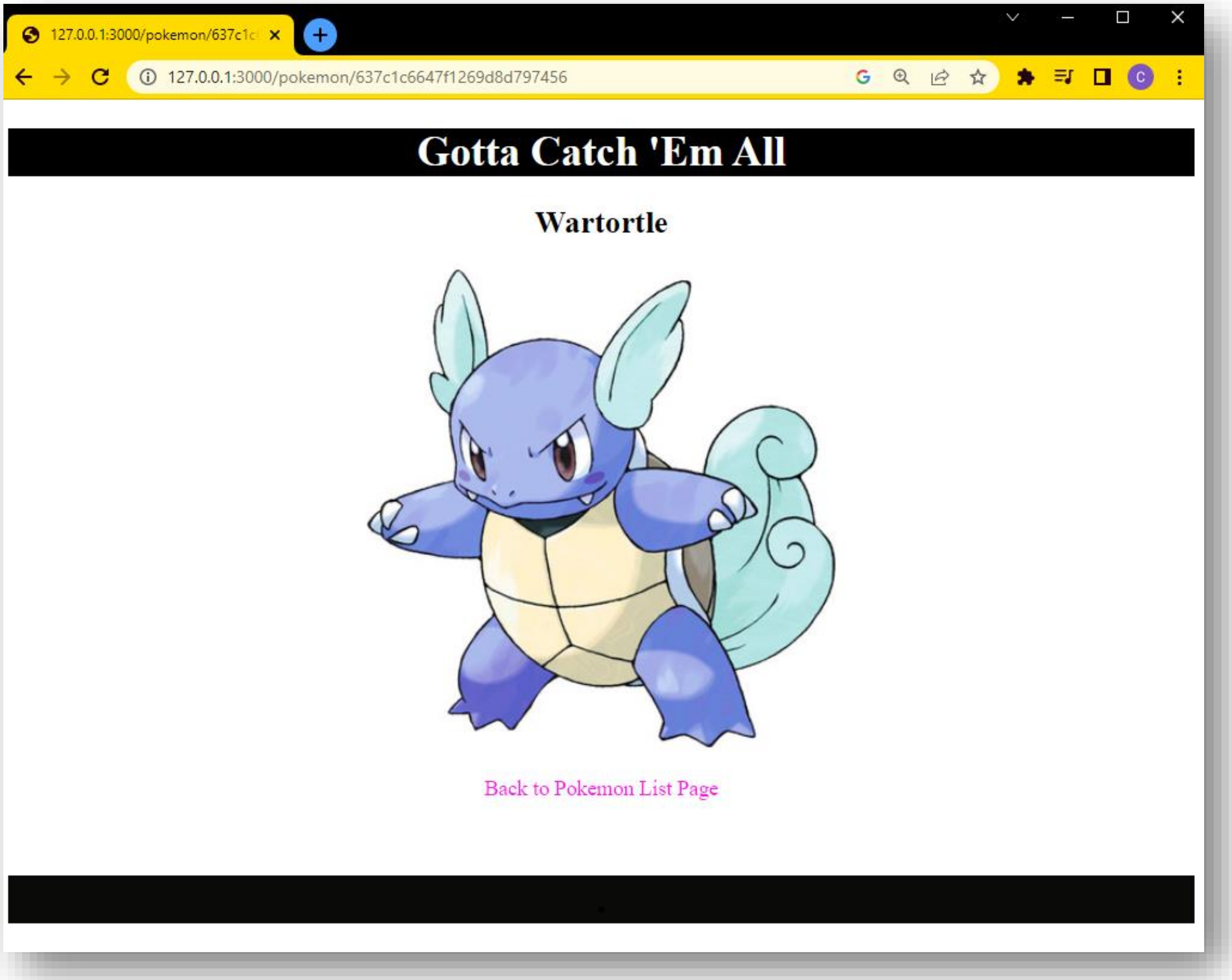
STORAGE SIZE: 36KB LOGICAL DATA SIZE: 283B TOTAL DOCUMENTS: 2 INDEXES TOTAL SIZE: 36KB

QUERY RESULTS: 1-2 OF 2

```
_id: ObjectId('637c1c6647f1269d8d797456')
name: "wartortle"
img: "http://img.pokemondb.net/artwork/wartortle"
createdAt: 2022-11-22T00:48:38.873+00:00
updatedAt: 2022-11-22T00:48:38.873+00:00
__v: 0
```

```
_id: ObjectId('637d4b5d55c1a4c39bc940a4')
name: "Bulbasaur "
img: "http://img.pokemondb.net/artwork/bulbasaur"
createdAt: 2022-11-22T22:21:17.494+00:00
updatedAt: 2022-11-22T22:21:17.494+00:00
__v: 0
```

In Pokedex we can click on each name (link) to get the name and image of Pokemon object loaded (from MongoDB to

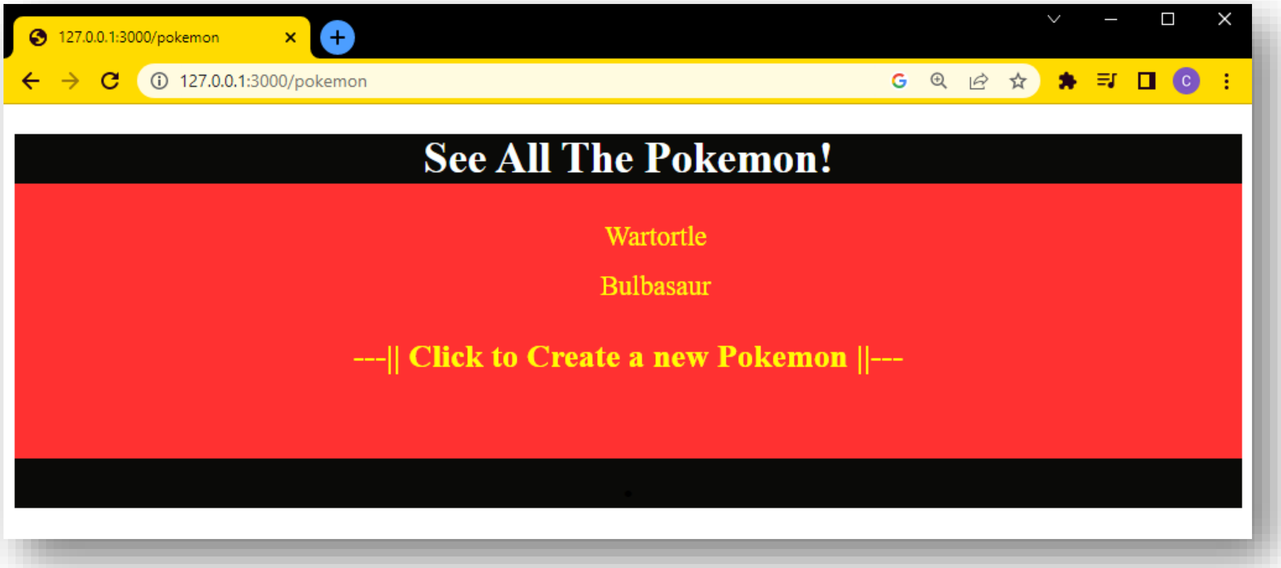


127.0.0.1:3000/pokemon/637c1c6647f1269d8d797456

Gotta Catch 'Em All

Wartortle

Back to Pokemon List Page



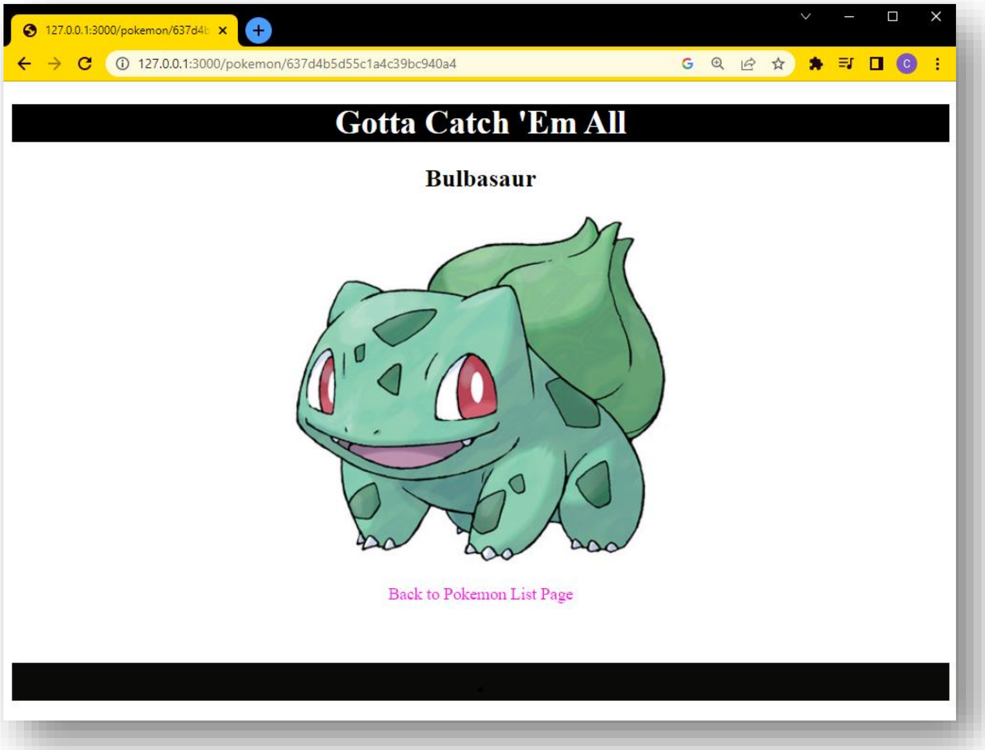
127.0.0.1:3000/pokemon

See All The Pokemon!

Wartortle

Bulbasaur

---|| Click to Create a new Pokemon ||---



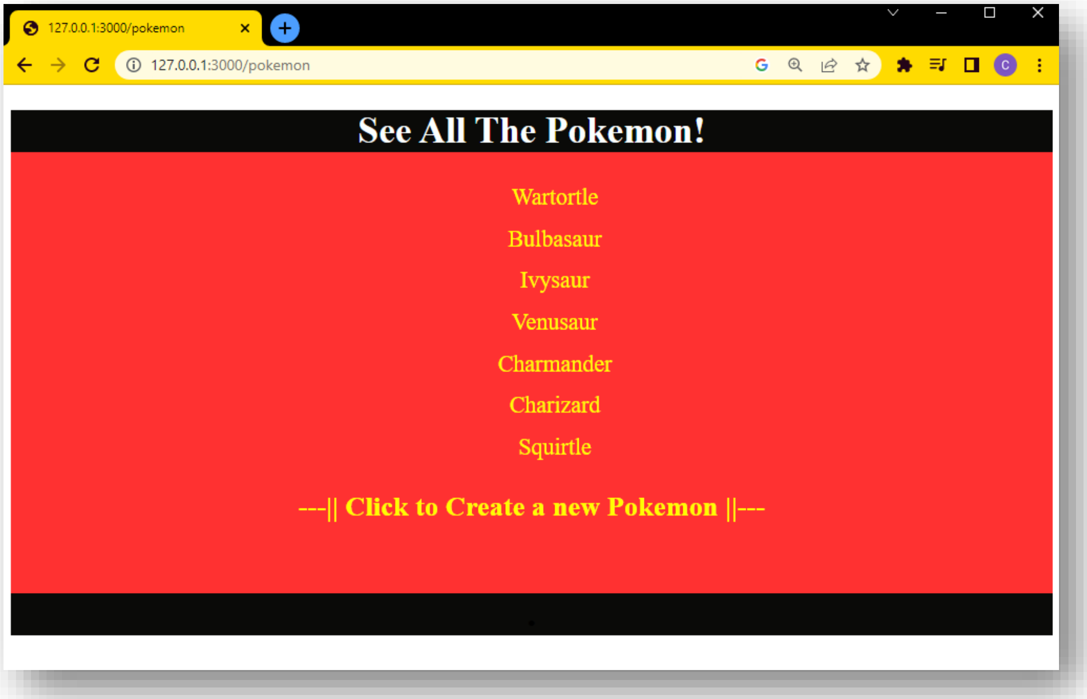
127.0.0.1:3000/pokemon/637d4b
127.0.0.1:3000/pokemon/637d4b5d55c1a4c39bc940a4

Gotta Catch 'Em All

Bulbasaur

Back to Pokemon List Page

Following is the list I created, using the name and image URL links in the description of Poke Express (seral page up – you can see them).



See All The Pokemon!

Wartortle

Bulbasaur

Ivysaur

Venusaur

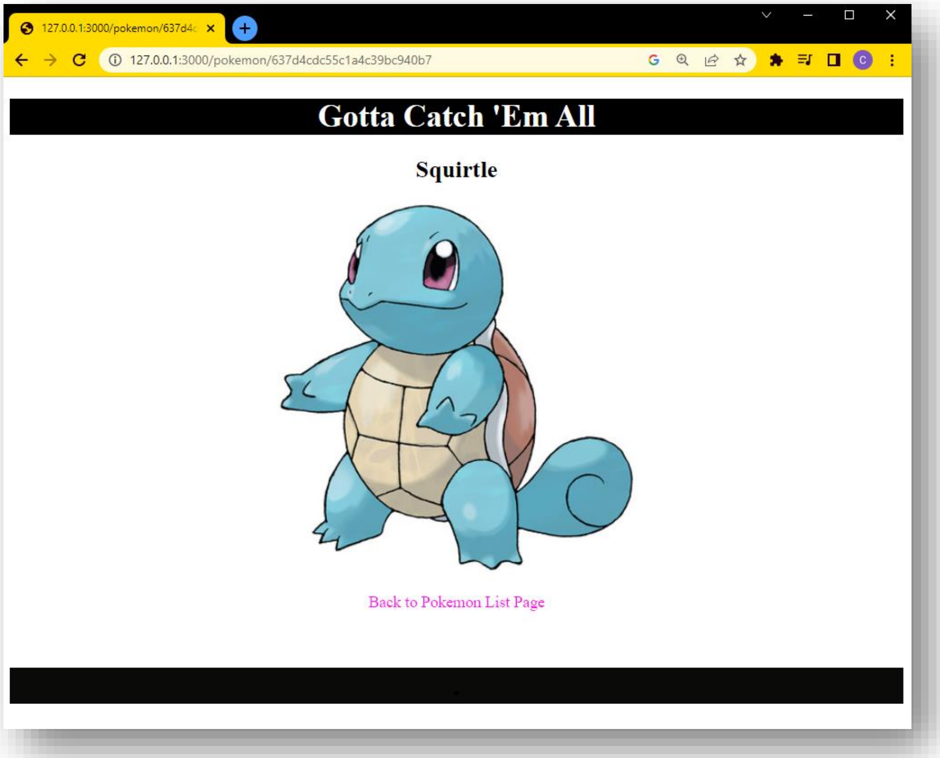
Charmander

Charizard

Squirtle

---|| Click to Create a new Pokemon ||---

Another example of image and name, after clicking on the its name in Pokedex:



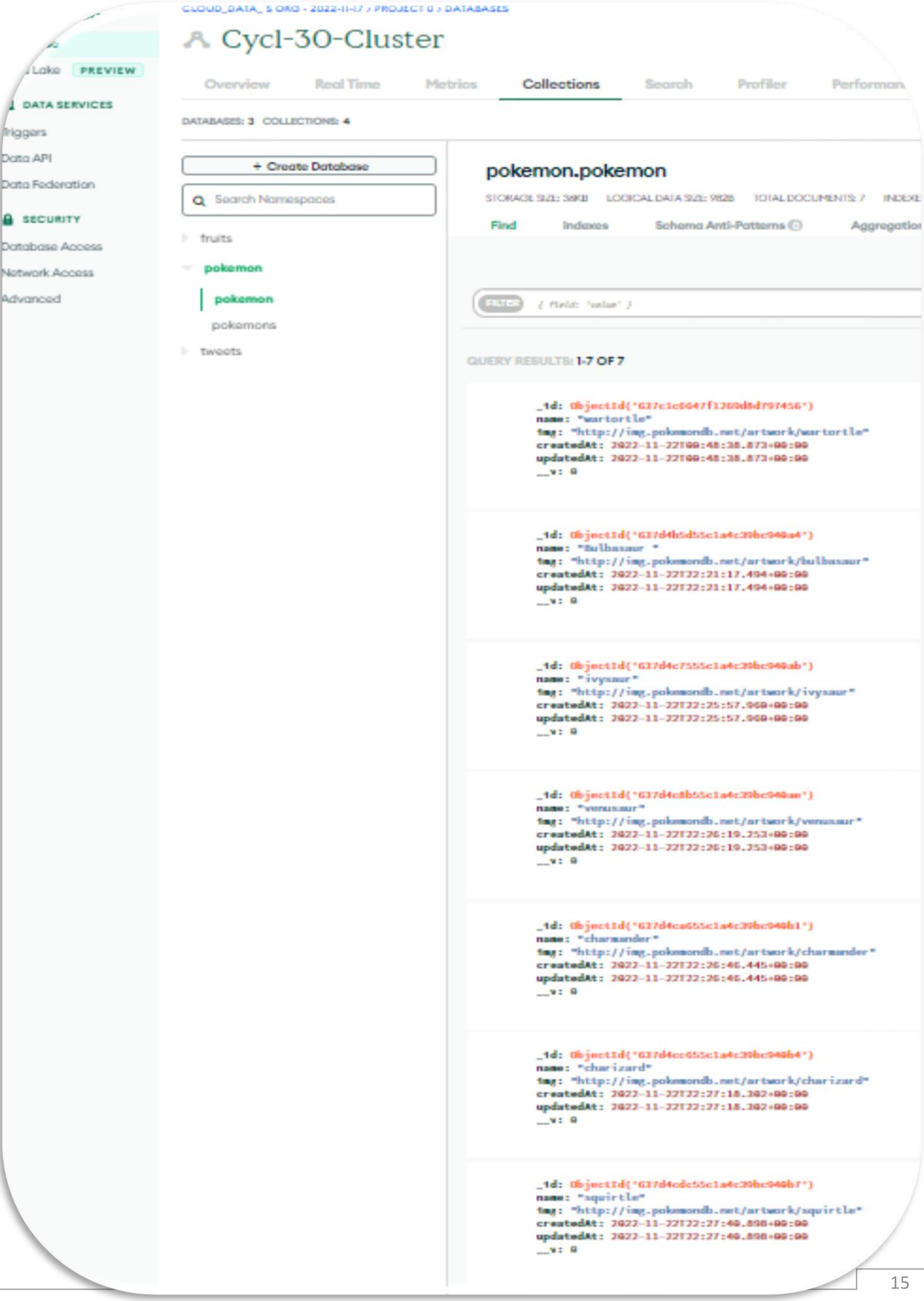
127.0.0.1:3000/pokemon/637d4cdc55c1a4c39bc940b7

Gotta Catch 'Em All

Squirtle

Back to Pokemon List Page

The database entries were as following:



```
_id: ObjectId("637c1c6647f1269d8d797456")
name: "wartortle"
img: "http://img.pokemondb.net/artwork/wartortle"
createdAt: 2022-11-22T09:48:38.873+00:00
updatedAt: 2022-11-22T09:48:38.873+00:00
__v: 0

_id: ObjectId("637d4b5d55c1a4c39bc940a4")
name: "Bulbasaur "
img: "http://img.pokemondb.net/artwork/bulbasaur"
createdAt: 2022-11-22T22:21:17.494+00:00
updatedAt: 2022-11-22T22:21:17.494+00:00
__v: 0

_id: ObjectId("637d4c7555c1a4c39bc940ab")
name: "ivysaur"
img: "http://img.pokemondb.net/artwork/ivysaur"
createdAt: 2022-11-22T22:25:57.960+00:00
updatedAt: 2022-11-22T22:25:57.960+00:00
__v: 0

_id: ObjectId("637d4c8b55c1a4c39bc940ae")
name: "venusaur"
img: "http://img.pokemondb.net/artwork/venusaur"
createdAt: 2022-11-22T22:26:19.253+00:00
updatedAt: 2022-11-22T22:26:19.253+00:00
__v: 0

_id: ObjectId("637d4ca655c1a4c39bc940b1")
name: "charmander"
img: "http://img.pokemondb.net/artwork/charmander"
createdAt: 2022-11-22T22:26:46.445+00:00
updatedAt: 2022-11-22T22:26:46.445+00:00
__v: 0

_id: ObjectId("637d4cc655c1a4c39bc940b4")
name: "charizard"
img: "http://img.pokemondb.net/artwork/charizard"
createdAt: 2022-11-22T22:27:18.302+00:00
updatedAt: 2022-11-22T22:27:18.302+00:00
__v: 0

_id: ObjectId("637d4cdc55c1a4c39bc940b7")
name: "squirtle"
img: "http://img.pokemondb.net/artwork/squirtle"
createdAt: 2022-11-22T22:27:40.898+00:00
updatedAt: 2022-11-22T22:27:40.898+00:00
__v: 0
```

The End