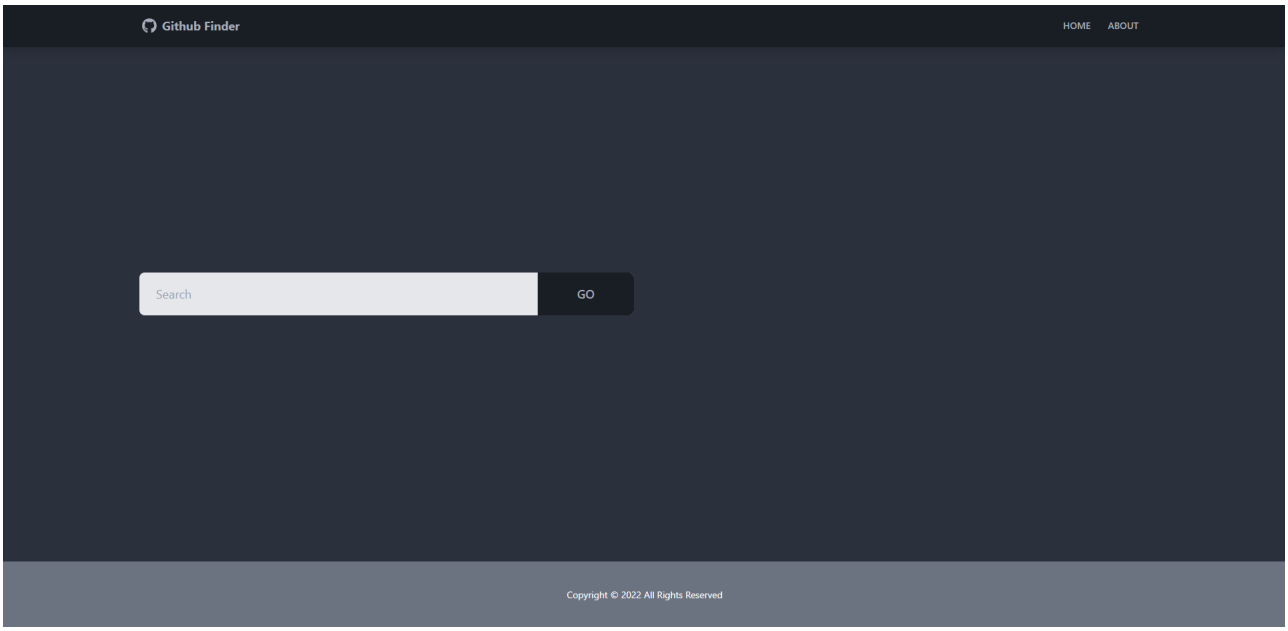
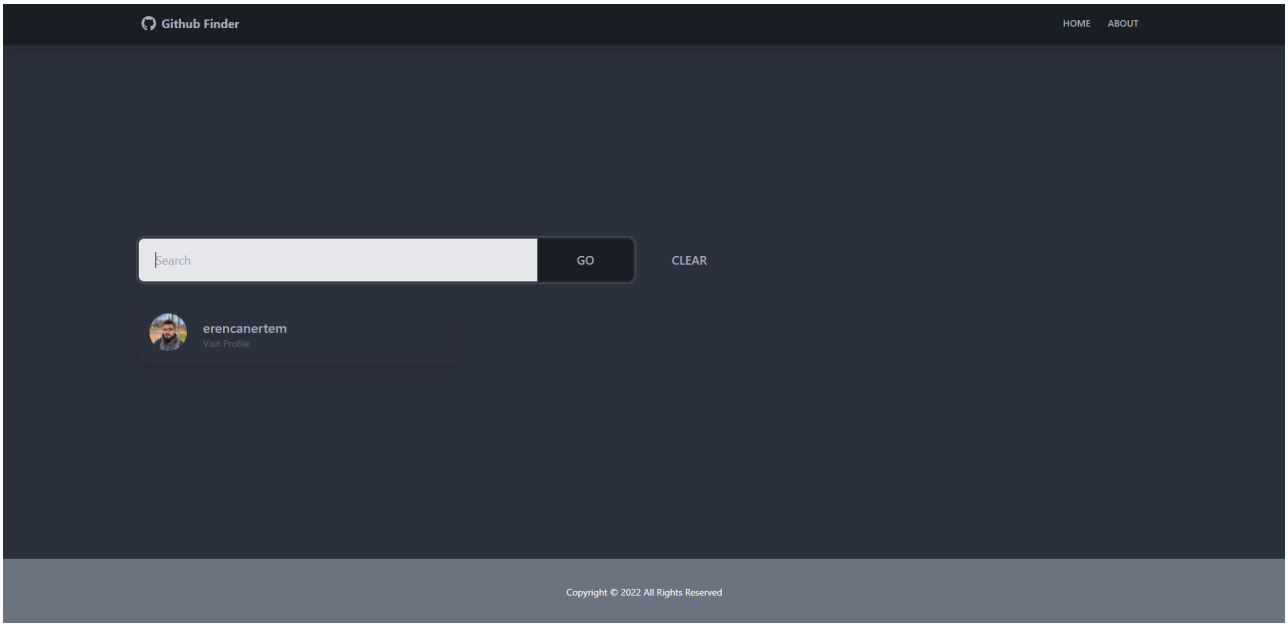
This is Entry Page for our app.



When we pass an input an go this page will show up.




```
GithubContext.js

import { createContext, useReducer } from "react";
```

```
import githubReducer from "./GithubReducer";

const GithubContext = createContext();

const GITHUB_URL = process.env.REACT_APP_GITHUB_URL;
const GITHUB_TOKEN = process.env.REACT_APP_GITHUB_TOKEN;

export const GithubProvider = ({ children }) => {
  const initialState = {
    users: [],
    user: {}, / At here we set our user as an object, We Will set
in at User.jsx file
    repos: [], / At here we set our repos as an array
    loading: false,
  }

  const [state, dispatch] = useReducer(githubReducer, initialState)


  // Get search Result
  const searchUsers = async (text) => {
    setLoading();

    const params = new URLSearchParams({
      q: text
    })

    const response = await fetch(`${GITHUB_URL}/search/users?
${params}`, {
      headers: {
        Authorization: `token ${GITHUB_TOKEN}`,
      },
    });
    const { items } = await response.json();

    dispatch({
      type: "GET_USERS",
      payload: items,

    })
  }
  // Get Single User Result
```

```
  const getUser = async (login) => { / We pass login as param for
the username of searching account
    setLoading();

    const response = await fetch(`${GITHUB_URL}/users/${login}`, {
      headers: {
        Authorization: `token ${GITHUB_TOKEN}`,
      },
    });

    if (response.status === 404) {
      window.location = "/notfound" {/ If there is no result then
notfound page will show up
    } else {
      const data = await response.json();

      dispatch({
        type: "GET_USER", / WE define type and payload for use as
case
        payload: data,
      })
    }


  }

  // Clear Users from  Result

  const clearUsers = () => {
    dispatch({
      type: "CLEAR_USERS",
    })
  }

  // Get Repos

  const getUserRepos = async (username) => { / We pass username as
param and below we catch repos for use at another component(USER)
    setLoading();

    const response = await
fetch(`${GITHUB_URL}/users/${username}/repos`, {
```

```
      headers: {
        Authorization: `token ${GITHUB_TOKEN}`,
      },
    });
    const items = await response.json();

    dispatch({
      type: "GET_REPOS", / We define type and payload for use as
case
      payload: items,

    })
  }

  // Set Loading

  const setLoading = () => dispatch({ type: "SET_LOADING" })

  return <GithubContext.Provider value={{
    users: state.users,
    loading: state.loading,
    repos: state.repos,
    user: state.user,
    searchUsers,
    clearUsers,
    getUser, / We pass it here, cause we'll use it another
component
    getUserRepos {/ We pass it here, cause we'll use it another
component
  }}>
    {children}
  </GithubContext.Provider>
}


export default GithubContext
```

GithubReducer.js

```
const githubReducer = (state, action) => {
  switch (action.type) {
    case "GET_USERS":
      return {
        ...state,
        users: action.payload,
        loading: false
      }
    case "GET_USER":
      return { / We set this for get user page, above i add the
screenshot.
        ...state,
        user: action.payload,
        loading: false
      }
    case "GET_REPOS":
      return {  / We set this for get repos for our User Component
, above i add the screenshot.
        ...state,
        repos: action.payload,
        loading: false
      }
    case "SET_LOADING":
      return {
        ...state,
        loading: true,
      }
    case "CLEAR_USERS":
      return {
        ...state,
        users: []
      }

    default:
      return state
  }
}

export default githubReducer
```

```
import { FaCodepen, FaStore, FaUserFriends, FaUsers } from "react-
icons/fa";
import { useEffect, useContext } from "react";
import { Link, useParams } from "react-router-dom"; / We import our
useParams hook.
import GithubContext from "../context/github/GithubContext";
import Spinner from "../components/layout/Spinner";
import RepoList from "../components/repos/RepoList";

function User() { / We add our user, repos,getUser,getUserRepos and
loading here.
  const { getUser, user, loading, getUserRepos, repos } =
    useContext(GithubContext);

  const params = useParams(); / This is to how it defines

  useEffect(() => { / This is for match our login parameters from
githubContext.
    getUser(params.login);
    getUserRepos(params.login);
  / We leave empty our dependencies. If we put there getUser,it
make a crash for our browser because it makes a loop.Could be error
at console, for manage that error,You can add a comment line like
below.
  // eslint-disable-next-line react-hooks/exhaustive-deps
  }, []);

  const { / At here we are choose parameter we'll use for our page.
(We set user as empty object at our GithubContext, now we filled it
datas from our API)
    name,
    type,
    avatar_url,
    location,
    bio,
```

```
    blog,
    twitter_username,
    login,
    html_url,
    followers,
    following,
    public_repos,
    public_gists,
    hireable,
  } = user;

  if (loading) {
    return <Spinner />;
  }

  return ( / At below we creating UI with tailwindCSS and DaisyUI
and datas from API
    <>
      <div className="w-full mx-auto lg:w-10/12">
        <div className="mb-4">
          <Link to="/" className="btn btn-ghost">
            Back To Search
          </Link>
        </div>
        <div className="grid grid-cols-1 xl:grid-cols-3 lg:grid-
cols-3 md:grid-cols-3 mb-8 md:gap-8">
          <div className="custom-card-image mb-6 md:mb-0">
            <div className="rounded-lg shadow-xl card image-full">
              <figure>
                <img src={avatar_url} alt="" />
              </figure>
              <div className="card-body justify-end">
                <h2 className="card-title mb-0">{name}</h2>
                <p>{login}</p>
              </div>
            </div>
          </div>
          <div className="col-span-2">
            <div className="mb-6">
              <h1 className="text-3xl card-title">
                {name}
```

```
                <div className="ml-2 mr-1 badge badge-success">
{type}</div>
                {hireable && (
                  <div className="mx-1 badge badge-
info">Hireable</div>
                )}
              </h1>
              <p>{bio}</p>
              <div className="mt-4 card-actions">
                <a
                  href={html_url}
                  target="_blank"
                  rel="noreferrer"
                  className="btn btn-outline"
                >
                  Visit Github Profile
                </a>
              </div>
            </div>
            <div className="w-full rounded-lg shadow-md bg-base-100
stats">
              {location && (
                <div className="stat">
                  <div className="stat-title text-
md">Location</div>
                  <div className="text-lg stat-value">{location}
</div>
                </div>
              )}
              {blog && (
                <div className="stat">
                  <div className="stat-title text-md">Website</div>
                  <div className="text-lg stat-value">
                    <a href={`${blog}`} target="_blank"
rel="noreferrer">
                      {blog}
                    </a>
                  </div>
                </div>
              )}
              {twitter_username && (
                <div className="stat">
```

```
                <div className="stat-title text-md">Twitter</div>
                <div className="text-lg stat-value">
                  <a
                    href=
{`https://twitter.com/${twitter_username}`}
                    target="_blank"
                    rel="noreferrer"
                  >
                    {twitter_username}
                  </a>
                </div>
              </div>
            )}
          </div>
        </div>
        <div className="w-full py-5 mb-6 rounded-lg shadow-md bg-
base-100 stats">
          <div className="stat">
            <div className="stat-figure text-secondary">
              <FaUsers className="text-3xl md:text-5xl" />
            </div>
            <div className="stat-title pr-5">Followers</div>
            <div className="stat-value pr-5 text-3xl md:text-4xl">
              {followers}
            </div>
          </div>
          <div className="stat">
            <div className="stat-figure text-secondary">
              <FaUserFriends className="text-3xl md:text-5xl" />
            </div>
            <div className="stat-title pr-5">Following</div>
            <div className="stat-value pr-5 text-3xl md:text-4xl">
              {following}
            </div>
          </div>
          <div className="stat">
            <div className="stat-figure text-secondary">
              <FaCodepen className="text-3xl md:text-5xl" />
            </div>
            <div className="stat-title pr-5">Public Repos</div>
            <div className="stat-value pr-5 text-3xl md:text-4xl">
```

```
              {public_repos}
            </div>
          </div>
          <div className="stat">
            <div className="stat-figure text-secondary">
              <FaStore className="text-3xl md:text-5xl" />
            </div>
            <div className="stat-title pr-5">Public Gists</div>
            <div className="stat-value pr-5 text-3xl md:text-4xl">
              {public_gists}
            </div>
          </div>
        </div>
      </div>
      <RepoList repos={repos} /> / We add our RepoList Component here, We'll see its codes below, we set repos as repos.With that we can use that repos at RepoList Component.
    </>
  );
}

export default User;
```

RepoList.jsx

```
import PropTypes from "prop-types";
import RepoItem from "./RepoItem";

function RepoList({ repos }) { /We pass repos here as state and we put our UI with map method.
  return (
    <div className="rounded-lg shadow-lg card bg-base-100">
      <div className="card-body">
```

```
        <h2 className="text-3xl my-4 font-bold card-title">
          Latest Repositories
        </h2>
        {repos.map((repo) => ( / We set repo as repo below(Param
could be different thing). With that we can use repo at our
RepoItem Component.
          <RepoItem key={repo.id} repo={repo} />
        ))}
      </div>
    </div>
  );
}

export default RepoList;

RepoList.propTypes = {
  repos: PropTypes.array.isRequired,
};
```

RepoItem.jsx

```
import PropTypes from "prop-types";
import { FaEye, FaInfo, FaLink, FaStar, FaUtensils } from "react-
icons/fa";

function RepoItem({ repo }) { / We pass repo here as prop and we
set its type as object and we pass our datas from API at below.with
use tailwindCSS and DaisyUI We put datas to our UI
  const {
    name,
    description,
    html_url,
    forks,
    open_issues,
    watchers_count,
    stargazers_count,
```

```
  } = repo;

  return (
    <div className="mb-2 rounded-mb card bg-gray-800 hover:bg-gray-
900">
      <div className="card-body">
        <h3 className="mb-2 text-xl font-semibold">
          <a href={html_url}>
            <FaLink className="inline mr-1" />
            {name}
          </a>
        </h3>
        <p className="mb-3">{description}</p>
        <div>
          <div className="mr-2 badge badge-info badge-lg">
            <FaEye className="mr-2" /> {watchers_count}
          </div>
          <div className="mr-2 badge badge-success badge-lg">
            <FaStar className="mr-2" /> {stargazers_count}
          </div>
          <div className="mr-2 badge badge-error badge-lg">
            <FaInfo className="mr-2" /> {open_issues}
          </div>
          <div className="mr-2 badge badge-info badge-lg">
            <FaUtensils className="mr-2" /> {forks}
          </div>
        </div>
      </div>
    </div>
  );
}

RepoItem.propTypes = {
  repo: PropTypes.object.isRequired,
};

export default RepoItem;
```

After changes, here is the our UI.