Source Code

Attaching Source Codes of TypeScript Files in the Project

The source codes of the TypeScript files are attached to the project in order to allow for easier debugging and understanding of the code. The source codes can be attached to the project in a variety of ways, such as using a version control system, a build system, or manually.

When using a version control system, the source codes of the TypeScript files are committed to the version control system along with the rest of the project files. This allows for easy tracking of changes to the source codes and for reverting to previous versions of the code if necessary.

When using a build system, the source codes of the TypeScript files are compiled to JavaScript and then attached to the project. This allows for the TypeScript files to be executed in a JavaScript environment.

When attaching the source codes of the TypeScript files manually, the source codes are copied to the project directory. This allows for the source codes to be opened in a text editor or IDE for editing and debugging.

By attaching the source codes of the TypeScript files to the project, it is possible to debug the code more easily and to understand the code better. This can help to improve the quality of the code and to resolve problems more quickly.

Login Page (/Login)

Layout.tsx

```
import "../globals.css";
import { WhHeader, Navspan } from "../Components/comps";
import Link from "next/link";

export const metadata = {
  title: "View Image",
  description: "Album to show people",
  icons: {
    icon: "./favicon.ico",
  },
};

export default function RootLayout({
  children,
}: {
```

```
  children: React.ReactNode;
}) {
  return (
    <div>
      <nav className="bg-black p-2 flex justify-evenly sm:flex-wrap">
        <Navspan link="/" value="Home" />
        <Navspan link="Gallery/" value="Gallery" home={true} />
        <Navspan link="Slideshow/" value="Slideshow" />
        <Navspan link="Upload/" value="Upload" />
      </nav>

      <>{children}</>
    </div>
  );
}
```

Page.tsx

```
import { PrismaClient } from "@prisma/client";
import Link from "next/link";
import { LoginPanel } from "./components";
import { Navspan } from "../Components/comps";

const prisma = new PrismaClient();

async function main() {
  const users = await prisma.auth.findMany();
  console.log(users);
}

async function handleSubmit(username: string, password: string) {
  const data = main()
    .then(async () => {
      await prisma.$disconnect();
    })
    .catch(async (e) => {
      console.error(e);
      await prisma.$disconnect();
    });
  console.log(data);
}

export default function Login() {
  return (
    <>
      <nav className="bg-black p-2 flex justify-evenly sm:flex-wrap">
        <Navspan link="Login/" value="Login" home={true} />
      </nav>

      <div className="bg-white bg-opacity-60 border-4 border-black m-10 p-10">
        <LoginPanel />
      </div>
    </>
```

```
  );
}
```

components.tsx

```
"use client";
import axios from "axios";
import Link from "next/link";
import { useRouter } from "next/navigation";
import { useState, useEffect, useRef } from "react";
import { fetchData } from "./functions";

export function LoginPanel() {
  const [err, SetErr] = useState(4); //1:incorrect psw 2:user doesnt exist 3:other 0:n
one
  const [isSubmited, SetSubmit] = useState(false);
  const router = useRouter();
  let username = useRef("");
  let password = useRef("");
  function renderSwitch(err: number) {
    if (true) {
      switch (err) {
        case 0:
          return "Successfully Logged In";
        case 1:
          return "Incorrect Password";
        case 2:
          return "User doesnt Exist";
        default:
          return "";
      }
    }
  }
  return (
    <div>
      <form
        onSubmit={async (e: any) => {
          e.preventDefault();
          const user = await fetchData();
          if (user) {
            let flag = 0;
            // @ts-ignore
            user.forEach((i: any) => {
              if (i.username === username.current) {
                flag = 1;
                SetErr((j) => {
                  return i.password === password.current ? 0 : 1;
                });
                if (i.password === password.current) {
                  SetSubmit(true);
                  router.push("/");
                }
              }
            });
```

```
            if (flag === 0) {
              SetErr(2);
            }
            return user;
          } else {
            return null;
          }
        }}
        className=" flex flex-col items-center"
      >
        <input
          name="username"
          type="text"
          id="userBar"
          placeholder="username"
          className="bg-lbl border-4 border-black m-5 rounded-lg px-2 py-1 placeholde
r:text-dbl"
          onChange={(e: any) => (username.current = e.target.value)}
          required
        />
        <input
          name="password"
          type="password"
          autoComplete="current-password"
          id="passBar"
          placeholder="password"
          className="bg-lbl border-4 black m-5 rounded-lg px-2 py-1 focus:stroke-none
 placeholder:text-dbl"
          onChange={(e: any) => (password.current = e.target.value)}
          required
        />
        <button>
          <input
            name="submit"
            type="submit"
            disabled={isSubmited}
            value="Login"
            className="bg-dbl disabled:bg-black text-white border-4 border-black shado
w-lbl px-7 m-5 py-2"
          />
        </button>
        <span>{renderSwitch(err)}</span>
        <span className="m-10 -mb-3">
          Dont have an account ?{" "}
          <Link href="/Login/Register" className="text-lred ">
            Sign Up
          </Link>
        </span>
      </form>
    </div>
  );
}
```

functions.tsx

```
"use server";
import { PrismaClient } from "@prisma/client";

export async function fetchData() {
  "use server";
  const prisma = new PrismaClient();
  let data;
  await prisma.auth
    .findMany()
    .then(async (i) => {
      data = i;
      await prisma.$disconnect();
    })
    .catch(async (e: any) => {
      console.log(e.response.data);
      await prisma.$disconnect();
    });
  await prisma.$disconnect();
  return data;
}
```

Home Page (/)

Layout.tsx

```
import "./globals.css";
import { WhHeader, Navspan, WhFooter } from "./Components/comps";
import Link from "next/link";

export const metadata = {
  title: "White Album",
  description: "Album to show people",
  icons: {
    icon: "./favicon.ico",
  },
};
export default function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  const darkMode = false;
  return (
    <html lang="en" className={darkMode ? "dark" : ""}>
      <body
        className={
          "relative flex flex-col items-stretch " +
          (darkMode ? "darksoul" : "wafall")
        }
      >
        <WhHeader user={"dommy"} />
        <main>{children}</main>
```

```
        <WhFooter />
      </body>
    </html>
  );
}
```

Page.tsx

```
import Image from "next/image";
import { WhHeader, Navspan } from "./Components/comps";
import fu from "./resources/portrait.gif";
import Link from "next/link";

function Hellobut() {
  return (
    <div>
      <input
        type="button"
        value="Hello"
        className={`text-l text-white bg-bl font-start2p bd-gray py-3 px-4 shadow-md s
hadow-black m-5 border-2 border-black
                            hover:bg-black`}
      />
      <input
        type="button"
        value="World"
        className={`text-l text-white bg-lred font-start2p bd-gray py-3 px-4 shadow-md
shadow-black m-5 border-2 border-black
                            hover:bg-black`}
      />
    </div>
  );
}

export default function Home() {
  return (
    <div className="relative">
      <nav className="bg-black dark:bg-white p-2 flex justify-evenly sm:flex-wrap">
        <Navspan link="/" value="Home" home={true} />
        <Navspan link="Gallery/" value="Gallery" />
        <Navspan link="Slideshow/" value="Slideshow" />
        <Navspan link="Upload/" value="Upload" />
      </nav>
      <div>
        <main className="p-10 pt-6 flex flex-col items-center justify-between ">
          <p className="bg-white border-black border-4 p-10 rounded text-center w-ful
l">
            This process is to create a display gallery for your{" "}
            <span className="text-red text-[1.1em]">stored images</span>
             in backup drives and cloud storages <br /> to access modify
            and view files from the same.
          </p>
```

```
          <div className="flex flex-col items-center justify-center md:flex md:flex-ro
w md:mt-4 w-full lg:items-stretch">
            <div className="bg-white border-4 rounded-lg p-6 flex justify-center items
-center basis-1/3 m-5 md:m-0 md:mr-2">
              <Image src={fu} alt="pixel-portrait" />
            </div>
            <div className="bg-white p-6 bd-black border-4 ml-4 rounded-lg basis-2/3 m
-5 md:m-0 md:ml-2 ">
              <p className="text-gray drop-shadow-2xl">
                A gallery is a mom&apos;s curiosity, which collects memories of
                the past and present , with our loved ones
              </p>
              <article className="text-center text-l text-red p-10">
                <i>
                  The image gallery was a symphony of visual delights,
                  <br />
                  A kaleidoscope of art, a feast for the sights.
                  <br />
                  Each frame captured a moment, a story to tell,
                  <br />
                  Of life&apos;s many wonders, of heaven and hell.
                  <br />
                  The walls were adorned with masterpieces,
                  <br />
                  A canvas of emotions, a treasure of pieces.
                  <br />
                  Each photograph a poem, a verse to read,
                  <br />
                  Of love and hope, of desire and need.
                  <br />
                </i>
              </article>
              <p className="text-dbl p-6 pt-2 text-right">- ChatGPT</p>
              <br />
              <span className="text-bl">
                Maintaining the gallery with upmost cautiousness and care is one
                of the most important work. This app makes sure that the
                requirements for a safe and a easy-to-use utility will be
                provided
              </span>
            </div>
          </div>
        </main>
      </div>
    </div>
  );
}
```

Gallery Page (/Gallery)

Layout.tsx

```
import "../globals.css";
import { WhHeader, Navspan, Butt } from "../Components/comps";
import Link from "next/link";
import { Suspense } from "react";

export const metadata = {
  title: "Server Gallery",
  description: "Album to show people",
  icons: {
    icon: "./favicon.ico",
  },
};

function Loading() {
  return (
    <div className="flex flex-col bg-white items-center border-4 border-black rounded-
lg bg-opacity-70 m-10 p-10">
      <span className="text-center text-lg">Loading ... </span>
    </div>
  );
}

export default function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <div>
      <nav className="bg-black p-2 flex justify-evenly sm:flex-wrap">
        <Navspan link="/" value="Home" />
        <Navspan link="#" value="Gallery" home={true} />
        <Navspan link="Slideshow/" value="Slideshow" />
        <Navspan link="Upload/" value="Upload" />
      </nav>
      <main>
        <Suspense fallback={<Loading />}>{children}</Suspense>
      </main>
    </div>
  );
}
```

Page.tsx

```
import { WhHeader, Navspan } from "../Components/comps";
import Image from "next/image";
import "../globals.css";
import { Imagelist } from "./clicomponents";
import "process";
import { useRouter } from "next/router";
import { v2 as Cloudinary } from "cloudinary";

type cloudimage = {
```

```
  asset_id: string;
  public_id: string;
  folder: string;
  filename: string;
  format: string;
  version: string;
  created_at: string;
  uploaded_at: string;
  bytes: number;
  backup_bytes: number;
  width: number;
  height: number;
  aspect_ratio: number;
  pixels: number;
  url: string;
  secure_url: string;
  status: string;
  access_mode: string;
  access_control: string;
  etag: string;
  created_by: any;
  uploaded_by: any;
  isToggle: boolean;
  isMatched: boolean;
};

// Configuration
Cloudinary.config({
  cloud_name: process.env.cloudapiname,
  api_key: process.env.cloudapikey,
  api_secret: process.env.cloudsecret,
  secure: true,
});

async function fetchThumbUrl(name: string) {
  const url = Cloudinary.url(name, {
    width: 160,
    height: 90,
    Crop: "fill",
  });
  return url;
}

async function fetchImages() {
  let images = { resources: [] };
  await Cloudinary.search
    .expression("resource_type:image")
    .execute()
    .then((result) => {
      images = result;
    });
  return images;
}

export default async function Gallery() {
  const img: any = await fetchImages();
  img.resources = img.resources.map((i: any) => {
    return { ...i, isToggle: false, isMatched: true };
```

```
  });
  return (
    <section className="flex flex-col">
        <Imagelist images={img} />
    </section>
  );
}
```

clicomponents.tsx

```
"use client";
import Image from "next/image";
import { useState, useEffect } from "react";
import React from "react";
import { useRouter } from "next/navigation";
import Link from "next/link";
import Router from "next/router";
import { Suspense } from "react";
import { v2 as Cloudinary } from "cloudinary";
import { State } from "zustand";

type cloudimage = {
  asset_id: string;
  public_id: string;
  folder: string;
  filename: string;
  format: string;
  version: string;
  created_at: string;
  uploaded_at: string;
  bytes: number;
  backup_bytes: number;
  width: number;
  height: number;
  aspect_ratio: number;
  pixels: number;
  url: string;
  secure_url: string;
  status: string;
  access_mode: string;
  access_control: string;
  etag: string;
  created_by: any;
  uploaded_by: any;
  isToggle: boolean;
  isMatched: boolean;
};

function Loading() {
  return <div>Loading ...</div>;
}

export function Imagecomp(props: { src: any; index: number; onCheck: any }) {
  const [checked, setCheck] = useState(props.src.isToggle);
```

```
  if (props.src.isMatched == false) {
    return <></>;
  }
  return (
    <div
      className={`bg-${
        checked ? "white" : "lgray"
      } border-2 border-black drop-shadow-sm rounded-md m-1 p-4 px-5 relative`}
    >
      <Link
        className="flex flex-col items-center"
        href={{ pathname: "/View", query: props.src }}
      >
        <div className="m-1 overflow-hidden border-black border-2 rounded-lg w-[12em]
 h-[6.75em]">
          <Image
            src={props.src.url}
            alt={`Image ${props.src.filename}`}
            width={400}
            height={225}
            quality="60"
            className="bg-white w-[12em] h-[6.75em] object-cover
                          transition-all delay-0
                          hover:scale-150 over:transition-all hover:ease-in-out hove
r:duration-[1000ms]"
            loading="eager"
          />
        </div>
        <div className="truncate text-black ml-2 my-1 overflow-x-hidden w-52">
          <span className="text-xs">
            {" "}
            {props.src.filename + "." + props.src.format}
          </span>
        </div>
      </Link>
      <input
        id={"select" + props.index.toString()}
        type="checkbox"
        className="absolute left-1 top-1 w-lg enabled:bg-dbl z-10"
        name="select"
        defaultChecked={checked}
        onChange={(e) => {
          props.onCheck(props.index, !checked);
          props.src.isToggle = props.src.isToggle ? false : true;
          setCheck(checked ? false : true);
        }}
      />
    </div>
  );
}

export function Imagelist(props: { images: any }) {
  const [sortby, SetSort] = useState("1");
  // 1-date , 2-name , 3-type
  const [imgcomponents, Setimages] = useState(props.images.resources);
  const [searchbut, Setsearch] = useState(0);

  function setToggle(id: number, toggle: boolean) {
```

```
    Setimages((s: Array<cloudimage>) => {
      s[id].isToggle = toggle;
      return s;
    });
  }
  return (
    <div className="flex flex-col items-stretch">
      <div className="border-4 border-black bg-white rounded-xl flex flex-col lg:flex-
row items-center justify-around m-5 px-10 py-2">
        <div className="flex flex-col items-center sm:flex-row">
          <span>Sort By</span>
          <select
            className="m-4 text-sm text-white bg-bl font-start2p bd-gray py-2 px-3 sha
dow-md shadow-black border-2 border-black"
            onChange={(event) => {
              if (event.target.value == "1") {
                Setimages((s: any) => {
                  s.sort((a: cloudimage, b: cloudimage) => {
                    return new Date(a.created_at).getTime() <
                      new Date(b.created_at).getTime()
                      ? 1
                      : -1;
                  });
                  return s;
                });
              } else if (event.target.value == "2") {
                Setimages((s: any) => {
                  s.sort((a: cloudimage, b: cloudimage) =>
                    a.filename > b.filename ? 1 : -1
                  );
                  return s;
                });
              } else if (event.target.value == "3") {
                Setimages((s: any) => {
                  s.sort((a: cloudimage, b: cloudimage) =>
                    a.format > b.format ? 1 : -1
                  );
                  return s;
                });
              }
              SetSort(event.target.value);
            }}
          >
            <option className="m-3 mt-5 bg-white p-3 text-black" value={1}>
              Date
            </option>
            <option className="m-3 mt-5 bg-white p-3 text-black" value={2}>
              Name
            </option>
            <option className="m-3 mt-5 bg-white p-3 text-black" value={3}>
              Type
            </option>
          </select>
        </div>
        <div className="flex flex-col items-center sm:flex-row">
          <input
            type="text"
            className="border-4 bg-lbl mx-4 px-5"
```

```
            onChange={(e) => {
              e.preventDefault();
              Setimages((s: any) => {
                s.map((i: cloudimage) => {
                  if (i.filename.match("^" + e.target.value + ".") != null) {
                    i.isMatched = true;
                  } else {
                    i.isMatched = false;
                  }
                });
                return s;
              });
            }}
          />
          <input
            type="button"
            value="search"
            className="m-4 text-sm text-white bg-bl font-start2p bd-gray py-2 px-3 sha
dow-md shadow-black border-2 border-black hover:bg-black hover:text-white"
            onClick={() => {
              Setsearch((s: any) => (s ? 0 : 1));
            }}
          />
        </div>
      </div>

      <div className="border-4 border-black rounded-xl m-5 my-2 bg-white bg-opacity-60
p-10">
        <div className="flex justify-between items-center w-full">
          <span className="text-black p-3 text-xl">{`Select Files`}</span>
          <div className="p-3  flex justify-center">
            <input
              type="button"
              value="&#128465;"
              className="px-3 py-1 text-xl bg-pur hover:bg-black border-4 border-black
shadow-black shadow-md mx-2 text-white"
            />
            <input
              type="button"
              className="px-3 py-1 font-arcade text-md bg-ylw hover:bg-black border-4
 border-black shadow-black shadow-md mx-2 text-white"
              value="&#129095;"
            />
            <input
              type="button"
              className="px-3 py-1 text-sm bg-green hover:bg-black border-4 border-bla
ck shadow-black shadow-md mx-2 text-white"
              value="Select All"
              onClick={() =>
                Setimages((s: any) => {
                  let count = 0;
                  s.map((i: cloudimage) => {
                    if (i.isMatched) {
                      i.isToggle = true;
                      count++;
                    }
                  });
                  return s;
```

```
                })
              }
            />
            <input
              type="button"
              value="Deselect All"
              className="px-3 py-1 text-sm bg-green hover:bg-black border-4 border-bla
ck shadow-black shadow-md mx-2 text-white"
              onClick={() =>
                Setimages((s: any) => {
                  let count = 0;
                  s.map((i: cloudimage) => {
                    i.isToggle = false;
                  });
                  return s;
                })
              }
            />
          </div>
        </div>
        <div className="grid p-3 xl:grid-cols-5 lg:grid-cols-3 md:grid-cols-2 sm:grid-
cols-2">
          {imgcomponents.map((src: any, index: number) => {
            return (
              <Imagecomp
                src={src}
                key={index}
                index={index}
                onCheck={setToggle}
              />
            );
          })}
        </div>
      </div>
    </div>
  );
}
```

Induvitual View Page (/Gallery)

Layout.tsx

```
import "../globals.css";
import { WhHeader, Navspan } from "../Components/comps";
import Link from "next/link";

export const metadata = {
  title: "View Image",
  description: "Album to show people",
```

```
  icons: {
    icon: "./favicon.ico",
  },
};

export default function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <div>
      <nav className="bg-black p-2 flex justify-evenly sm:flex-wrap">
        <Navspan link="/" value="Home" />
        <Navspan link="Gallery/" value="Gallery" home={true} />
        <Navspan link="Slideshow/" value="Slideshow" />
        <Navspan link="Upload/" value="Upload" />
      </nav>

      <>{children}</>
    </div>
  );
}
```

Page.tsx

```
import { WhHeader, Navspan, Butt } from "../Components/comps";
import Image from "next/image";
import abc from "/images/ghi.png";
import "../globals.css";
import { v2 as Cloudinary } from "cloudinary";
import { useRouter } from "next/navigation";
//import { ImageComponent } from "./clicomponents";
import Link from "next/link";

Cloudinary.config({
  cloud_name: process.env.cloudapiname,
  api_key: process.env.cloudapikey,
  api_secret: process.env.cloudsecret,
  secure: true,
});

function View(props: { params: any; searchParams: any }) {
  const query = props.searchParams;
  return (
    <main className="m-5 flex flex-row justify-around items-center">
      <div className="p-10 border-4 border-black w-[60vw] bg-white bg-opacity-75">
        <Image
          src={query.url}
          width={query.width}
          height={query.height}
          alt="adcf"
        />
      </div>
```

```
      <div className="flex flex-col bg-white p-10 m-4 border-black border-4 text-md">
        <div className="flex justify-around my-2">
          <span className="basis-1/2">File Name : </span>
          <span className="basis-1/2 overflow-x-hidden">{query.filename}</span>
        </div>
        <div className="flex justify-around my-2">
          <span className="basis-1/2">File Type : </span>
          <span className="basis-1/2 overflow-x-hidden">{query.format}</span>
        </div>
        <div className="flex justify-around my-2">
          <span className="basis-1/2">Created At :</span>
          <span className="basis-1/2 overflow-x-hidden">
            {query.created_at}
          </span>
        </div>
        <div className="my-5">
          <a
            href={query.url}
            download
            className="bg-bl border-4 border-black px-3 py-2 shadow-xl my-10 hover:bg-
black hover:text-white"
          >
            Download
          </a>
        </div>
      </div>
    </main>
  );
}
/*
export async function getServerSideProps() {
  return {
    props: {
      path: "/images/abc.png",
    },
  };
}
*/
export default View;
```

Slideshow Page (/Slideshow)

Layout.tsx

```
import "../globals.css";
import { WhHeader, Navspan, Butt } from "../Components/comps";
import Link from "next/link";
import { Suspense } from "react";

export const metadata = {
```

```
  title: "Server Gallery",
  description: "Album to show people",
  icons: {
    icon: "./favicon.ico",
  },
};

function Loading() {
  return (
    <div className="flex flex-col bg-white items-center border-4 border-black rounded-
lg bg-opacity-70 m-10 p-10">
      <span className="text-center text-lg">Loading ... </span>
    </div>
  );
}

export default function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <div>
      <nav className="bg-black p-2 flex justify-evenly sm:flex-wrap">
        <Navspan link="/" value="Home" />
        <Navspan link="Gallery/" value="Gallery" />
        <Navspan link="Slideshow/" value="Slideshow" home={true} />
        <Navspan link="Upload/" value="Upload" />
      </nav>
      <Suspense fallback={<Loading />}>
        <main>{children}</main>
      </Suspense>
    </div>
  );
}
```

Page.tsx

```
import Image from "next/image";
import "../globals.css";
import { Imagelist } from "./clicomponents";
import "process";
import { useRouter } from "next/router";
import { v2 as Cloudinary } from "cloudinary";

type cloudimage = {
  asset_id: string;
  public_id: string;
  folder: string;
  filename: string;
  format: string;
  version: string;
  created_at: string;
  uploaded_at: string;
```

```
  bytes: number;
  backup_bytes: number;
  width: number;
  height: number;
  aspect_ratio: number;
  pixels: number;
  url: string;
  secure_url: string;
  status: string;
  access_mode: string;
  access_control: string;
  etag: string;
  created_by: any;
  uploaded_by: any;
  isToggle: boolean;
  isMatched: boolean;
};

// Configuration
Cloudinary.config({
  cloud_name: process.env.cloudapiname,
  api_key: process.env.cloudapikey,
  api_secret: process.env.cloudsecret,
  secure: true,
});

async function fetchThumbUrl(name: string) {
  const url = Cloudinary.url(name, {
    width: 160,
    height: 90,
    Crop: "fill",
  });
  return url;
}

async function fetchImages() {
  let images = { resources: [] };
  await Cloudinary.search
    .expression("resource_type:image")
    .execute()
    .then((result) => {
      images = result;
    });
  return images;
}

export default async function Gallery() {
  const img: any = await fetchImages();
  img.resources = img.resources.map((i: any) => {
    return { ...i, isToggle: false, isMatched: true };
  });
  return (
    <section className="flex flex-col">
      <Imagelist images={img} />
    </section>
  );
}
```

clicomponents.tsx

```
"use client";
import { useState, useEffect } from "react";
import Image from "next/image";
import { Suspense } from "react";

function Loadingimage() {
  return <div>Loading...</div>;
}

export function Imagelist(props: { images: any }) {
  const [id, setId] = useState(0);
  const [imgcomponents, Setimages] = useState(props.images.resources);
  const [delay, SetDelay] = useState(5);
  const [paused, SetPause] = useState(true);
  useEffect(() => {
    if (!paused) {
      const x = setInterval(() => {
        setId(id == imgcomponents.length - 1 ? 0 : id + 1);
      }, delay * 1000);
      return () => clearInterval(x);
    }
  });
  return (
    <div className="flex flex-col m-10">
      <div className="flex  justify-around items-center mb-10">
        <div className="flex flex-col items-center">
          <div className=" p-5 border-4 border-black bg-white bg-opacity-70 m-3">
            <Suspense fallback={<Loadingimage />}>
              <Image
                src={imgcomponents[id].url}
                alt={imgcomponents[id].filename}
                width={imgcomponents[id].width}
                height={imgcomponents[id].height}
                className="w-[900px] h-[506.25px] object-cover"
              />
            </Suspense>
          </div>
        </div>
      </div>
      <div className=" text-center p-5 border-4 border-black bg-white bg-opacity-70">
        <span className="mb-10 mt-2px text-lg text-center max-w-lg">
          {imgcomponents[id].filename + "." + imgcomponents[id].format}
        </span>
        <div className="flex items-center justify-center">
          <input
            type="button"
            value="<"
            className="text-black hover:text-bl text-4xl font-bold disabled:text-white
px-5 py-2"
            onClick={() => {
              setId((s) => (s === 0 ? imgcomponents.length - 1 : s - 1));
            }}
          />
```

```
          <button className="m-5" disabled={paused}>
            <span
              className={`text-3xl text-${
                paused ? "bl" : "black"
              } hover:text-bl`}
              onClick={() => SetPause(true)}
            >
              ||
            </span>
          </button>
          <button disabled={!paused} className="m-5">
            <span
              className={`text-3xl text-${
                paused ? "black" : "bl"
              } hover:text-bl`}
              onClick={() => SetPause(false)}
            >
              ▶
            </span>
          </button>
          <input
            type="button"
            value=">"
            onClick={() =>
              setId((s: number) => (s === imgcomponents.length - 1 ? 0 : s + 1))
            }
            className="text-black hover:text-bl text-4xl  font-bold disabled:text-whit
e px-5 py-2"
          />
        </div>
      </div>
    </div>
  );
}
```

Upload Page (/Upload)

Layout.tsx

```
import "../globals.css";
import { WhHeader, Navspan, Butt } from "../Components/comps";
import Link from "next/link";

export const metadata = {
  title: "Upload Files",
  description: "Album to show people",
  icons: {
    icon: "./favicon.ico",
```

```
  },
};

export default function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <div>
      <nav className="bg-black p-2 flex justify-evenly sm:flex-wrap">
        <Navspan link="/" value="Home" />
        <Navspan link="Gallery/" value="Gallery" />
        <Navspan link="Slideshow/" value="Slideshow" />
        <Navspan link="Upload/" value="Upload" home={true} />
      </nav>
      <main className="flex flex-col items-center m-5 border-4 border-black rounded-lg
bg-white p-5 min-h-[72vh] h-[80vh] bg-opacity-70">
        <h1 className="text-4xl mb-10 mt-2 text-black font-bold ">
          Upload Files
        </h1>

        {children}
      </main>
    </div>
  );
}
```

Page.tsx

```
import Image from "next/image";
import { useState, useEffect } from "react";
import { v2 as Cloudinary } from "cloudinary";
import { Suspense } from "react";
import { Uploadsection } from "./clicomponents";

export default function Upload() {
  return <Uploadsection />;
}
```

clicomponents.tsx

```
"use client";
import { useState, useEffect } from "react";
import { v2 as Cloudinary } from "cloudinary";
import { Suspense } from "react";
import Image from "next/image";
import { cloudimage } from "../Components/types";
import axios from "axios";

function ImageComponent(props: { images: Array<string> }) {
```

```
  if (props.images.length) {
    return (
      <div className="grid grid-cols-2 md:grid-cols-3 lg:grid-cols-5 xl:grid-cols-6 it
ems-center px-3 bg-gray border-4 border-black max-h-[50vh] overflow-y-scroll my-5">
        {Array.from(props.images).map((file, index) => {
          return (
            <Image
              src={file}
              alt="Image"
              key={index}
              className="m-3 bg-white border-4 border-black"
              width={240}
              height={135}
              quality={30}
              loading="lazy"
            />
          );
        })}
      </div>
    );
  } else {
    return (
      <div className="bg-gray m-10 p-5 rounded-lg border-4 border-black">
        No Files
      </div>
    );
  }
}

function Loading() {
  return <div>Loading...</div>;
}
```

Components (/Components)

comps.tsx

```
import Image from "next/image";
import wa from "../resources/wa.png";
import usans from "../resources/whichbook.gif";
import Link from "next/link";

export function Butt(props: { name: string }) {
  return (
    <input
      type="button"
      value={props.name}
      className={`text-l text-white bg-bl font-start2p bd-gray py-3 px-4 shadow-md sha
dow-black m-5 border-2 border-black
```

```
                            hover:bg-black`}
    />
  );
}

export function Footer() {
  return (
    <footer className="bg-black flex justify-center text-white p-10 py-5 mt-5">
      <span className="">&copy; Copyright titan_rocky@2023</span>
      <Link
        href="https://github.com/titan-rocky/white-album"
        className="text-ylw ml-[20em] text-lg"
        target="_blank"
      >
        Github repo
      </Link>
    </footer>
  );
}

export function WhHeader(props: { user: string }) {
  return (
    <header className="bg-white dark:bg-black stretch flex flex-row p-4 items-center j
ustify-between">
      <h1 className="text-3xl mt-1 p-3 text-dbl dark:text-white">
        White Album
      </h1>
      <div className="flex items-center mx-10">
        <Image src={usans} alt="sans" width={60} height={60} />
        <span className="text-lg">
          Welcome <b className="text-dbl text-lg">{props.user}</b> !
        </span>
        <span className="text-lg text-red mx-10">Logout</span>
      </div>
    </header>
  );
}

export function WhFooter() {
  return (
    <footer className="bg-black flex justify-center text-white p-5 py-2">
      <span className="">&copy; Copyright titan_rocky@2023</span>
      <Link
        href="https://github.com/titan-rocky/white-album"
        className="text-ylw ml-[20em] text-lg"
        target="_blank"
      >
        Github repo
      </Link>
    </footer>
  );
}

export function Navspan(props: {
  link: string;
  value: string;
  home: boolean;
  disabled: boolean;
```

```
}) {
  if (props.home) {
    return (
      <span className="text-bl dark:text-dgray text-l">{props.value}</span>
    );
  } else if (props.disabled) {
    return (
      <span className="text-lred dark:text-lgray text-l">{props.value}</span>
    );
  }
  return (
    <Link
      href={props.link}
      className={`text-${(props.home && "bl") || "white"}
         dark:text-lgray text-l hover:text-bl dark:hover:text-black
        `}
    >
      <span className="">{props.value}</span>
    </Link>
  );
}

Navspan.defaultProps = {
  home: false,
  disabled: false,
};
```

types.tsx

```
export type cloudimage = {
  asset_id: string;
  public_id: string;
  folder: string;
  filename: string;
  format: string;
  version: string;
  created_at: string;
  uploaded_at: string;
  bytes: number;
  backup_bytes: number;
  width: number;
  height: number;
  aspect_ratio: number;
  pixels: number;
  url: string;
  secure_url: string;
  status: string;
  access_mode: string;
  access_control: string;
  etag: string;
  created_by: any;
  uploaded_by: any;
  isToggle: boolean;
```

```
  isMatched: boolean;
};
```

Prisma ORM

schema.prisma

```
// This is your Prisma schema file,
// learn more about it in the docs: https://pris.ly/d/prisma-schema

generator client {
  provider = "prisma-client-js"
}

datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL")
}

model Auth {
        id Int @id @default(autoincrement())
        username String @unique
        password String
    }
```