UNIVERSITAS GUNADARMA

PRAKTIKUM PEMROGRAMAN WEB

MANUAL BOOK

"Dream Station"

Nama Anggota :

1. Dimas Arya Sauki A. (50422428)
2. Muhamad Zidan Indratama (50422968)

Kelas : 3IA08

Fakultas : Teknologi Industri

Jurusan : Teknik Informatika

PJ : Muhammad Rizqi Fajri

Ditulis Guna Melengkapi Sebagian Syarat

Praktikum Pemrograaman Web

Universitas Gunadarma

2025

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam kehidupan, setiap individu memiliki mimpi atau cita-cita yang menjadi pendorong semangat untuk mencapai tujuan hidup. Namun, seringkali mimpi-mimpi tersebut terlupakan atau kehilangan daya dorong akibat rutinitas dan tantangan yang dihadapi sehari-hari. Oleh karena itu, diperlukan suatu media yang dapat menjadi wadah untuk mencatat, mengingat, dan membagikan mimpi-mimpi tersebut agar semangat untuk mewujudkannya tetap terjaga.

Website ini dirancang sebagai platform di mana pengguna dapat menuliskan mimpi-mimpi mereka. Dengan fitur yang memungkinkan orang lain untuk melihat mimpi yang telah dibagikan, website ini juga menjadi sumber inspirasi dan motivasi. Melalui konsep berbagi mimpi ini, diharapkan tercipta komunitas yang saling mendukung dan memotivasi untuk terus berusaha mencapai tujuan masing-masing.

Proyek ini dikembangkan menggunakan teknologi modern yang membagi pengembangan menjadi dua sisi: frontend dan backend.

- Frontend: Dibangun menggunakan TypeScript, NextJS, TailwindCSS, ShadcnUI, Lucide Icon, React Hook Form, React Query, Axios, dan Zod.
- Backend: Dibangun menggunakan TypeScript, NestJS, Cloudinary, Prisma, dan MongoDB.

Dengan memanfaatkan teknologi-teknologi tersebut, website ini diharapkan dapat memberikan pengalaman pengguna yang intuitif, responsif, dan aman.

1.2 Tujuan

Dalam Tujuan kami dalam membuat situs web Dream Station ini adalah:

1. Mencatat dan Mengingatkan:

   Menyediakan wadah bagi pengguna untuk mencatat dan mengingat mimpi-mimpi mereka, sehingga tidak terlupakan di tengah kesibukan sehari-hari.
2. Membangun Komunitas Inspiratif:

   Membentuk komunitas yang saling mendukung dan memotivasi melalui berbagi mimpi.

3. Meningkatkan Semangat dan Motivasi:

   Memberikan inspirasi kepada pengguna lain dengan melihat mimpi-mimpi yang telah dibagikan.

4. Mendukung Teknologi Modern:

   Menerapkan teknologi terkini dalam pengembangan untuk memastikan performa, keamanan, dan kenyamanan pengguna.

5. Menyediakan Platform yang User-Friendly:

   Menghadirkan antarmuka yang sederhana, responsif, dan mudah digunakan oleh semua kalangan.

BAB II

PEMBAHASAN

2.1 Tahapan Pembuatan Website Dream Station

Website ini dibangun dengan memanfaatkan teknologi modern yang dibagi menjadi dua bagian utama, yaitu frontend dan backend:

2.1.1 Frontend

Frontend merupakan bagian antarmuka pengguna yang bertanggung jawab atas tampilan dan interaksi pengguna. Teknologi yang digunakan meliputi:

- TypeScript: Bahasa pemrograman yang memperluas JavaScript dengan tipe data statis.
- NextJS: Framework React yang mendukung server-side rendering dan static site generation.
- TailwindCSS: Framework CSS yang mempermudah styling dengan pendekatan utility-first.
- ShadcnUI: Library UI untuk membuat antarmuka yang modern dan konsisten.
- Lucide Icon: Library ikon yang ringan dan mudah digunakan.
- React Hook Form: Library untuk pengelolaan formulir yang efisien dan fleksibel.
- React Query: Library untuk pengelolaan data asynchronous.
- Axios: Library untuk melakukan HTTP request.
- Zod: Library validasi schema untuk memastikan data yang diterima sesuai dengan format yang diharapkan.

2.1.2 Backend

Backend bertanggung jawab atas logika bisnis dan pengelolaan data. Teknologi yang digunakan meliputi:

- TypeScript: Untuk memastikan kode yang lebih aman dan dapat diprediksi.
- NestJS: Framework backend yang modular dan mendukung pengembangan yang terstruktur.
- Cloudinary: Layanan penyimpanan dan pengelolaan media.

- Prisma: ORM untuk pengelolaan database.
- MongoDB: Database NoSQL yang fleksibel dan skalabel.

2.2 Alur Kerja Website

- Pengguna Menulis Mimpi: Pengguna dapat menulis dan menyimpan mimpi mereka melalui formulir yang telah disediakan.
- Penyimpanan Data: Mimpi yang ditulis pengguna akan disimpan dalam database MongoDB melalui backend yang telah dikembangkan.
- Menampilkan Mimpi: Mimpi-mimpi yang telah disimpan akan ditampilkan di halaman utama website untuk dilihat oleh pengguna lain.
- Berbagi Motivasi: Dengan melihat mimpi-mimpi orang lain, pengguna dapat saling memberikan inspirasi dan motivasi.

2.3 Fitur Utama

- Pencatatan Mimpi: Fitur untuk mencatat dan menyimpan mimpi pengguna.
- Berbagi Mimpi: Fitur untuk menampilkan mimpi yang telah dibagikan oleh pengguna lain.
- Pengelolaan Media: Integrasi dengan Cloudinary untuk mengelola gambar atau media lain yang terkait dengan mimpi.
- Validasi Data: Validasi input pengguna menggunakan Zod untuk memastikan data yang disimpan valid.

2.4 Keunggulan Teknologi

- Responsif dan Cepat: Menggunakan NextJS dan TailwindCSS untuk memastikan performa yang optimal di berbagai perangkat.
- Keamanan Data: Dengan Prisma dan validasi berbasis schema, data pengguna terlindungi.
- Pengalaman Pengguna yang Intuitif: Antarmuka dirancang agar mudah digunakan bahkan oleh pengguna awam.
- Pengelolaan Media yang Efisien: Cloudinary memastikan media dikelola dengan baik dan dapat diakses dengan cepat.

Dengan pembahasan ini, diharapkan pengguna memahami bagaimana platform ini bekerja serta keunggulan yang ditawarkan.

BAB III

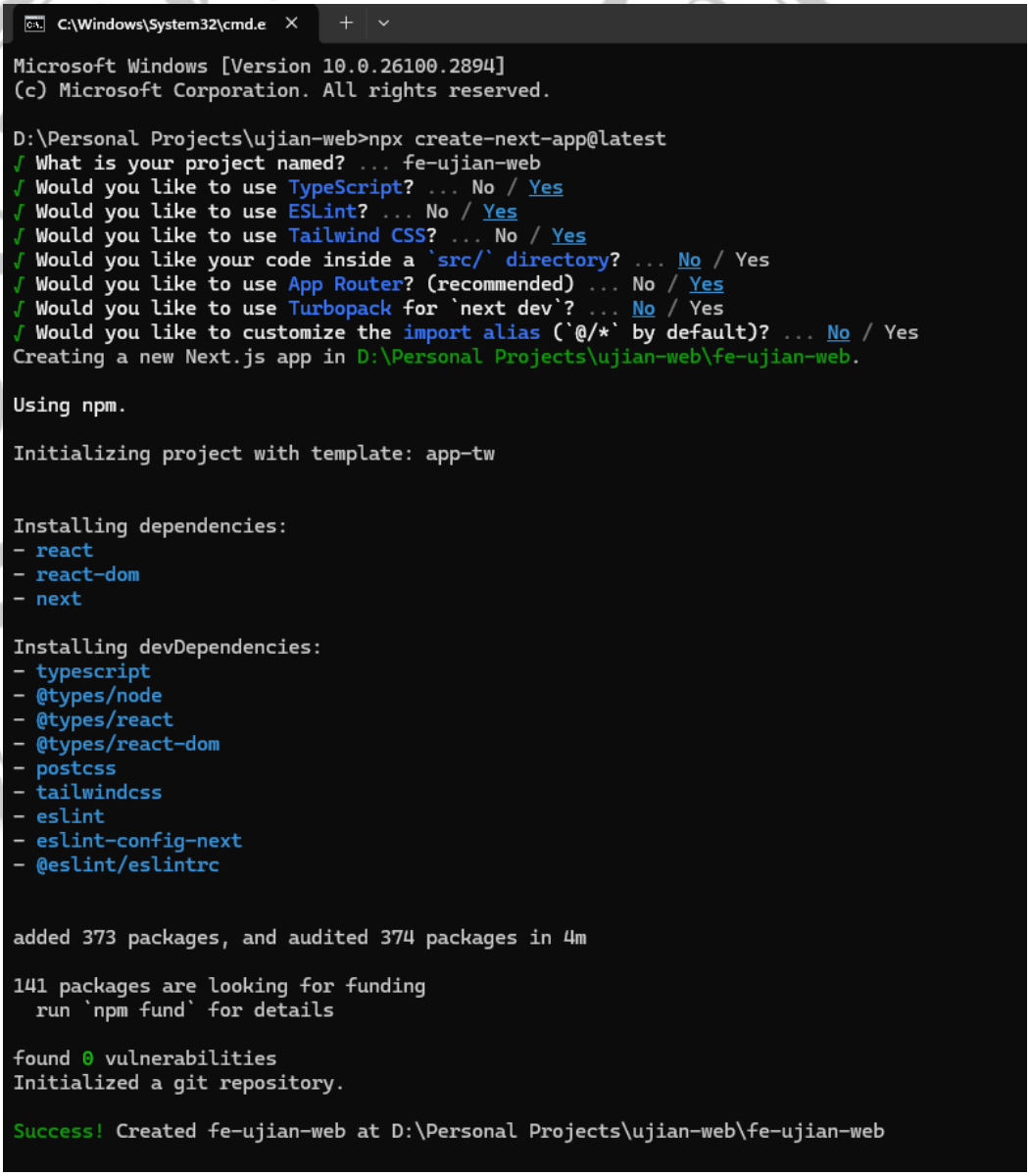
ANALISA DAN PERANCANGAN

3.1 Frontend

3.1.1 Instalasi Project

Instalasi project adalah langkah awal untuk mempersiapkan lingkungan pengembangan yang sesuai dengan kebutuhan. Pada bagian ini, akan dijelaskan cara mengatur dependensi dan konfigurasi awal project.

- Instalasi NextJS



- Instalasi React Query

```
D:\Personal Projects\ujian-web\fe-ujian-web>npm install @tanstack/react-query @tanstack/eslint-plugin-query @tanstack/react-query-devtools

added 5 packages, and audited 379 packages in 10s

146 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

- Instalasi Zod dan React Hook Form

```
D:\Personal Projects\ujian-web\fe-ujian-web>npm install zod react-hook-form @hookform/resolvers axios

added 12 packages, and audited 391 packages in 5s

149 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

3.1.2 Providers

Providers adalah komponen yang berfungsi menyediakan konteks atau layanan tertentu ke dalam aplikasi. Penjelasan ini mencakup pengaturan dan penggunaannya di dalam project.

- Tanstack Query Provider

```tsx
"use client";

import { QueryClient, QueryClientProvider } from "@tanstack/react-query";
import { ReactQueryDevtools } from "@tanstack/react-query-devtools";
import React from "react";

const TanStackProvider = ({ children }: { children: React.ReactNode }) => {
  const queryClient = new QueryClient();
  return (
    <QueryClientProvider client={queryClient}>
      {children}
      <ReactQueryDevtools initialIsOpen={false} />
    </QueryClientProvider>
  );
};

export default TanStackProvider;
```

- Theme Provider

```tsx
"use client";

import * as React from "react";
import { ThemeProvider as NextThemesProvider } from "next-themes";

export function ThemeProvider({
  children,
  ...props
}: React.ComponentProps<typeof NextThemesProvider>) {
  return <NextThemesProvider {...props}>{children}</NextThemesProvider>;
}
```

3.1.3 Types

Types merujuk pada tipe data yang digunakan dalam pengembangan. Bagian ini menjelaskan definisi tipe-tipe data untuk memastikan konsistensi dan keandalan dalam kode.

- TBenefit

```
import { IconType } from "react-icons/lib";

export type TBenefit = {
  id: string;
  icon: IconType;
  title: string;
  description: string;
};
```

- TDream

```
export type TDream = {
  id: string;
  username: string;
  title: string;
  slug: string;
  description: string;
  image: string;
  createdAt: string;
  updatedAt: string;
};
```

- TNavigation

```
export type TNavigation = {
  id: string;
  label: string;
  href: string;
};
```

- TSocial

```
import { IconType } from "react-icons/lib";

export type TSocial = {
  id: string;
  icon: IconType;
  label: string;
  href: string;
};
```

- TTeam

```
export type TTeam = {
  id: string;
  name: string;
  bio: string;
  photo: string;
  link: string;
};
```

- TTech

```
import { IconType } from "react-icons/lib";

export type TTech = {
  name: string;
  icon: IconType;
};
```

- TTestimonial

```
export type TTestimonial = {
  id: string;
  image: string;
  name: string;
  job: string;
  description: string;
};
```

3.1.4 Mock Data

Mock data digunakan untuk pengujian dan pengembangan fitur. Pada bagian ini, akan dijelaskan bagaimana data dummy dibuat dan dimanfaatkan dalam pengembangan.

- benefitData

```
import { TBenefit } from "@/types/benefit.type";
import { FaLightbulb, FaUsers, FaRocket, FaChartLine } from "react-icons/fa";

export const benefitData: TBenefit[] = [
  {
    id: "1",
    icon: FaLightbulb,
    title: "Unlock New Perspectives",
    description:
      "Sharing your dreams can spark fresh ideas and innovative solutions that you may
have never considered before.",
  },
  {
    id: "2",
    icon: FaUsers,
    title: "Connect with Like-Minded People",
    description:
      "Build meaningful relationships with individuals who share your aspirations and are
eager to support your journey.",
  },
  {
    id: "3",
    icon: FaRocket,
    title: "Take the First Step Forward",
    description:
      "Expressing your dreams is the first step toward turning them into actionable plans
and achieving your goals.",
  },
  {
    id: "4",
    icon: FaChartLine,
    title: "Track Your Progress",
    description:
      "Documenting your dreams helps you measure your growth and stay motivated as you
move closer to your aspirations.",
  },
];
```

- navigationData

```
import { TNavigation } from "@/types/navigation.type";

export const navigationData: TNavigation[] = [
  {
    id: "1",
    label: "Home",
    href: "/",
  },
  {
    id: "2",
    label: "About Us",
    href: "/about-us",
  },
  {
    id: "3",
    label: "Dreams",
    href: "/dreams",
  },
];
```

- socialData

```
import { TSocial } from "@/types/social.type";
import { SiNestjs, SiNextdotjs } from "react-icons/si";

export const socialData: TSocial[] = [
  {
    id: "1",
    icon: SiNextdotjs,
    label: "Repo GitHub Frontend",
    href: "https://github.com/zidanindratama/ujian-pemrograman-web/tree/main/fe-ujian-
pemrograman-web",
  },
  {
    id: "2",
    icon: SiNestjs,
    label: "Repo GitHub Backend",
    href: "https://github.com/zidanindratama/ujian-pemrograman-web/tree/main/be-ujian-
pemrograman-web",
  },
];
```

- teamData

```
import { TTeam } from "@/types/team.type";

export const teamData: TTeam[] = [
  {
    id: "1",
    name: "Muhamad Zidan Indratama",
    bio: "Fullstack Web Developer from Bekasi with a strong track record in web design and
development.",
    photo: "/teams/zidan.jpg",
    link: "https://www.linkedin.com/in/zidan-indratama/",
  },
  {
    id: "2",
    name: "Dimas Arya Sauki A.",
    bio: "Passionate mobile app developer. Computer Science student seeking opportunities
to build innovative apps.",
    photo: "/teams/dimas.jpg",
    link: "https://www.instagram.com/arya_dimas99/",
  },
];
```

- techData

```ts
import { TTech } from "@/types/tech.type";
import { BiLogoTypescript } from "react-icons/bi";
import { RiNextjsFill } from "react-icons/ri";
import {
  SiNestjs,
  SiCloudinary,
  SiPrisma,
  SiMongodb,
  SiTailwindcss,
  SiShadcnui,
  SiLucide,
  SiReacthookform,
  SiReactquery,
  SiAxios,
  SiZod,
} from "react-icons/si";

export const backendTechData: TTech[] = [
  {
    name: "Typescript",
    icon: BiLogoTypescript,
  },
  {
    name: "NestJS",
    icon: SiNestjs,
  },
  {
    name: "Cloudinary",
    icon: SiCloudinary,
  },
  {
    name: "Prisma ORM",
    icon: SiPrisma,
  },
  {
    name: "MongoDB",
    icon: SiMongodb,
  },
];

export const frontendTechData: TTech[] = [
  {
    name: "Typescript",
    icon: BiLogoTypescript,
  },
  {
    name: "NextJS",
    icon: RiNextjsFill,
  },
  {
    name: "Tailwind CSS",
    icon: SiTailwindcss,
  },
  {
    name: "ShadCN UI",
    icon: SiShadcnui,
  },
  {
    name: "Lucide Icon",
    icon: SiLucide,
  },
  {
    name: "React Hook Form",
    icon: SiReacthookform,
  },
  {
    name: "React Query",
    icon: SiReactquery,
  },
  {
    name: "Axios",
    icon: SiAxios,
  },
  {
    name: "Zod",
    icon: SiZod,
  },
];
```

- testimonialData

```
import { TTestimonial } from "@/types/testimonial.type";

export const testimonialData: TTestimonial[] = [
  {
    id: "1",
    image: "https://randomuser.me/api/portraits/men/1.jpg",
    name: "Ravi Pratama",
    job: "Software Engineer",
    description:
      "This platform helped me grow my career by connecting me with amazing mentors. The
resources are insightful, and the community is incredibly supportive.",
  },
  {
    id: "2",
    image: "https://randomuser.me/api/portraits/men/2.jpg",
    name: "Dwi Santoso",
    job: "Product Designer",
    description:
      "I love how easy it is to connect with people who share the same passions. The
feedback I received was invaluable for refining my design skills.",
  },
  {
    id: "3",
    image: "https://randomuser.me/api/portraits/men/3.jpg",
    name: "Ari Prakoso",
    job: "Digital Marketer",
    description:
      "The advice I got from sharing my dreams here motivated me to push harder in my
marketing strategies. I feel more confident about my career path now.",
  },
  {
    id: "4",
    image: "https://randomuser.me/api/portraits/men/4.jpg",
    name: "Budi Hartono",
    job: "Entrepreneur",
    description:
      "I've found so many like-minded individuals who support my startup journey. This
platform is a great place to receive guidance and share experiences.",
  },
  {
    id: "5",
    image: "https://randomuser.me/api/portraits/men/5.jpg",
    name: "Andi Setiawan",
    job: "UX/UI Designer",
    description:
      "As a designer, it's hard to find good advice. This community gave me the insights I
needed to make my work more user-friendly and visually appealing.",
  },
  {
    id: "6",
    image: "https://randomuser.me/api/portraits/men/6.jpg",
    name: "Indra Kurniawan",
    job: "Software Developer",
    description:
      "Being part of this community made me realize the potential of my dream. The support
I received was essential for overcoming obstacles in my development projects.",
  },
  {
    id: "7",
    image: "https://randomuser.me/api/portraits/men/7.jpg",
    name: "Hendra Wijaya",
    job: "Photographer",
    description:
      "This platform gave me the courage to share my dream of becoming a professional
photographer. I've learned so much and made great connections.",
  },
  {
    id: "8",
    image: "https://randomuser.me/api/portraits/men/8.jpg",
    name: "Toni Susanto",
    job: "Web Developer",
    description:
      "The experience here helped me develop new skills and connect with people who
understand the struggles of being a developer. I feel more motivated to continue
improving.",
  },
  {
    id: "9",
    image: "https://randomuser.me/api/portraits/men/9.jpg",
    name: "Eko Pratama",
    job: "Business Analyst",
    description:
      "By sharing my dreams on this platform, I received the support I needed to make
better decisions in my career. I've gained so much clarity.",
  },
  {
    id: "10",
    image: "https://randomuser.me/api/portraits/men/10.jpg",
```

```
    name: "Mulyadi Arifin",
    job: "Marketing Specialist",
    description:
      "I found the motivation to take my marketing skills to the next level. The advice
and resources here have been incredibly valuable in shaping my career.",
  },
];
```

3.1.5 Helpers

Helpers adalah fungsi utilitas yang membantu menjalankan logika umum di berbagai bagian aplikasi. Bagian ini memberikan panduan tentang fungsionalitas yang telah disediakan.

- Axios Instance

```
import axios from "axios";

const DEV_API_URL = "http://localhost:3001";
const PROD_API_URL = "https://be-ujian-pemrograman-web.vercel.app";

const axiosInstance = axios.create({
  baseURL: process.env.NODE_ENV === "development" ? DEV_API_URL : PROD_API_URL,
  withCredentials: true,
  headers: {
    "Content-Type": "application/x-www-form-urlencoded",
  },
});

export default axiosInstance;
```

- Helper

```
export function formatBytes(
  bytes: number,
  opts: {
    decimals?: number;
    sizeType?: "accurate" | "normal";
  } = {}
) {
  const { decimals = 0, sizeType = "normal" } = opts;

  const sizes = ["Bytes", "KB", "MB", "GB", "TB"];
  const accurateSizes = ["Bytes", "KiB", "MiB", "GiB", "TiB"];
  if (bytes === 0) return "0 Byte";
  const i = Math.floor(Math.log(bytes) / Math.log(1024));
  return `${(bytes / Math.pow(1024, i)).toFixed(decimals)} ${
    sizeType === "accurate" ? accurateSizes[i] ?? "Bytes" : sizes[i] ?? "Bytes"
  }`;
}

export const formatDate = (dateString: string): string => {
  const date = new Date(dateString);
  return new Intl.DateTimeFormat("en-US", {
    month: "long",
    day: "numeric",
    year: "numeric",
  }).format(date);
};

export const generateSlug = (title: string): string => {
  return title
    .toLowerCase()
    .replace(/\s+/g, "-")
    .replace(/[^\w-]+/g, "")
    .replace(/^-+|-+$/g, "");
};
```

3.1.6 Hooks

Hooks digunakan untuk mengelola state dalam komponen React. Bagian ini menjelaskan hooks yang telah dibuat dan cara penggunaannya.

- Use Add Data

```tsx
import { useMutation, useQueryClient } from "@tanstack/react-query";
import { LoaderCircle } from "lucide-react";
import { toast } from "sonner";
import { useRouter } from "next/navigation";
import axiosInstance from "@/helpers/axios-instance";

type fetchProps = {
  queryKey: string;
  dataProtected: string;
  backUrl?: string;
  multipart?: boolean;
};

export const useAddData = ({
  queryKey,
  dataProtected,
  backUrl,
  multipart = false,
}: fetchProps) => {
  const queryClient = useQueryClient();
  const router = useRouter();

  const mutation: any = useMutation<any>({
    mutationFn: (addData: any) => {
      if (multipart === false) {
        return axiosInstance.post(`/api/${dataProtected}`, addData);
      }
      return axiosInstance.post(`/api/${dataProtected}`, addData, {
        headers: {
          "Content-Type": "multipart/form-data",
        },
      });
    },
    onMutate: (variables) => {
      toast(
        <>
          <p className="flex gap-x-2 items-center">
            <LoaderCircle className="animate-spin h-4 w-4" />
            Adding the data...
          </p>
        </>
      );
    },
    onError: (error: any, variables, context) => {
      toast.error(`${error.response.data.message}`);
    },
    onSuccess: (data, variables, context) => {
      toast.success("Successfully added the data!");

      queryClient.invalidateQueries({ queryKey: [queryKey] });

      if (backUrl) {
        router.push(backUrl);
      }
    },
  });

  return mutation;
};
```

- Use Callback Ref

```
import * as React from "react";

/**
 * @see https://github.com/radix-ui/primitives/blob/main/packages/react/use-callback-ref/src/useCallbackRef.tsx
 */

/**
 * A custom hook that converts a callback to a ref to avoid triggering re-renders when passed as a
 * prop or avoid re-executing effects when passed as a dependency
 */
function useCallbackRef<T extends (...args: never[]) => unknown>(
  callback: T | undefined
): T {
  const callbackRef = React.useRef(callback);

  React.useEffect(() => {
    callbackRef.current = callback;
  });

  // https://github.com/facebook/react/issues/19240
  return React.useMemo(
    () => ((...args) => callbackRef.current?.(...args)) as T,
    []
  );
}

export { useCallbackRef };
```

- Use Controllable State

```
import * as React from "react";
import { useCallbackRef } from "./use-callback-ref";

/**
 * @see https://github.com/radix-ui/primitives/blob/main/packages/react/use-controllable-state/src/useControllableState.tsx
 */

type UseControllableStateParams<T> = {
  prop?: T | undefined;
  defaultProp?: T | undefined;
  onChange?: (state: T) => void;
};

type SetStateFn<T> = (prevState?: T) => T;

function useControllableState<T>({
  prop,
  defaultProp,
  onChange = () => {},
}: UseControllableStateParams<T>) {
  const [uncontrolledProp, setUncontrolledProp] = useUncontrolledState({
    defaultProp,
    onChange,
  });
  const isControlled = prop !== undefined;
  const value = isControlled ? prop : uncontrolledProp;
  const handleChange = useCallbackRef(onChange);

  const setValue: React.Dispatch<React.SetStateAction<T | undefined>> =
    React.useCallback(
      (nextValue) => {
        if (isControlled) {
          const setter = nextValue as SetStateFn<T>;
          const value =
            typeof nextValue === "function" ? setter(prop) : nextValue;
          if (value !== prop) handleChange(value as T);
        } else {
          setUncontrolledProp(nextValue);
        }
      },
      [isControlled, prop, setUncontrolledProp, handleChange]
    );

  return [value, setValue] as const;
}

function useUncontrolledState<T>({
  defaultProp,
  onChange,
}: Omit<UseControllableStateParams<T>, "prop">) {
```

```
  const uncontrolledState = React.useState<T | undefined>(defaultProp);
  const [value] = uncontrolledState;
  const prevValueRef = React.useRef(value);
  const handleChange = useCallbackRef(onChange);

  React.useEffect(() => {
    if (prevValueRef.current !== value) {
      handleChange(value as T);
      prevValueRef.current = value;
    }
  }, [value, prevValueRef, handleChange]);

  return uncontrolledState;
}

export { useControllableState };
```

- Use Debounce

```
"use cllient";

import { useState, useEffect } from "react";

export function useDebounce<T>(value: T, delay: number): T {
  const [debouncedValue, setDebouncedValue] = useState<T>(value);

  useEffect(() => {
    const handler = setTimeout(() => {
      setDebouncedValue(value);
    }, delay);

    return () => {
      clearTimeout(handler);
    };
  }, [value, delay]);

  return debouncedValue;
}
```

- Use Delete Data

```
import { useMutation, useQueryClient } from "@tanstack/react-query";
import { LoaderCircle } from "lucide-react";
import { toast } from "sonner";
import { useRouter } from "next/navigation";
import axiosInstance from "@/helpers/axios-instance";

type fetchProps = {
  queryKey: string;
  dataProtected: string;
  backUrl?: string;
};

export const useDeleteData = ({
  queryKey,
  dataProtected,
  backUrl,
}: fetchProps) => {
  const queryClient = useQueryClient();
  const router = useRouter();

  const mutation: any = useMutation<any>({
    mutationFn: (deleteData: any) => {
      return axiosInstance.delete(`/api/${dataProtected}`, deleteData);
    },
    onMutate: (variables) => {
      toast(
        <>
          <p className="flex gap-x-2 items-center">
            <LoaderCircle className="animate-spin h-4 w-4" />
            Deleting the data...
          </p>
        </>
      );
    },
    onError: (error: any, variables, context) => {
      toast.error(`${error.response.data.message}`);
    },
    onSuccess: (data, variables, context) => {
      toast.success("Successfully delete the data!");
      queryClient.invalidateQueries({ queryKey: [queryKey] });
      if (backUrl) {
```

```
        router.push(backUrl);
      }
    },
  });

  return mutation;
};
```

- Use Fetch Data

```
import axiosInstance from "@/helpers/axios-instance";
import { keepPreviousData, useQuery } from "@tanstack/react-query";

type fetchProps = {
  queryKey: [string, string] | [string];
  dataProtected: string;
  keep?: boolean;
};

export const useFetchData = ({ queryKey, dataProtected, keep }: fetchProps) => {
  const {
    data,
    isLoading,
    isError,
    isSuccess,
    refetch,
    isRefetching,
    isStale,
  } = useQuery({
    queryKey: queryKey,
    queryFn: async () => await axiosInstance.get(`/api/${dataProtected}`),
    placeholderData: keepPreviousData,
  });

  return {
    data,
    isLoading,
    isError,
    isStale,
    isSuccess,
    refetch,
    isRefetching,
  };
};
```

- Use Update Data

```
import axiosInstance from "@/helpers/axios-instance";
import { useMutation, useQueryClient } from "@tanstack/react-query";
import { LoaderCircle } from "lucide-react";
import { useRouter } from "next/navigation";
import { toast } from "sonner";

type fetchProps = {
  queryKey: string;
  dataProtected: string;
  multipart?: boolean;
  backUrl?: string;
};

export const useUpdateData = ({
  queryKey,
  dataProtected,
  multipart = false,
  backUrl,
}: fetchProps) => {
  const queryClient = useQueryClient();
  const router = useRouter();

  const mutation: any = useMutation<any>({
    mutationFn: (updateData: any) => {
      if (multipart === false) {
        return axiosInstance.patch(`/api/${dataProtected}`, updateData);
      }
      return axiosInstance.patch(`/api/${dataProtected}`, updateData, {
        headers: {
          "Content-Type": "multipart/form-data",
        },
      });
    },
    onMutate: (variables) => {
      toast(
```

```
        <>
          <p className="flex gap-x-2 items-center">
            <LoaderCircle className="animate-spin h-4 w-4" />
            Changing the data...
          </p>
        </>
      );
    },
    onError: (error: any, variables, context) => {
      console.log(error);
      toast.error(`${error.response.data.message}`);
    },
    onSuccess: (data, variables, context) => {
      toast.success("Successfully change the data!");

      queryClient.invalidateQueries({ queryKey: [queryKey] });

      if (backUrl) {
        router.push(backUrl);
      }
    },
  });

  return mutation;
};
```

3.1.7 Schema Zod

Schema Zod digunakan untuk validasi data. Pada bagian ini, akan dijelaskan definisi dan implementasi schema untuk memverifikasi input pengguna.

- dreamSchema

```
import { z } from "zod";

export const dreamSchema = z.object({
  username: z
    .string({
      required_error: "Username cannot be empty",
      invalid_type_error: "Username cannot be empty",
    })
    .min(1, { message: "Username cannot be empty" })
    .max(50, { message: "Username must be at most 50 characters long" }),
  title: z
    .string({
      required_error: "Title cannot be empty",
      invalid_type_error: "Title cannot be empty",
    })
    .min(1, { message: "Title cannot be empty" })
    .min(5, { message: "Title must be at least 5 characters long" })
    .max(50, { message: "Title must be at most 50 characters long" }),
  description: z
    .string({
      required_error: "Description cannot be empty",
      invalid_type_error: "Description must be a valid string",
    })
    .min(1, { message: "Description cannot be empty" })
    .min(5, { message: "Description must be at least 5 characters long" })
    .max(150, { message: "Description must be at most 150 characters long" }),
  image: z.any(),
});

export type TDreamForm = z.infer<typeof dreamSchema>;
```

3.1.8 Component

Component adalah elemen utama dalam pengembangan frontend. Bagian ini menjelaskan struktur, gaya, dan interaksi dari berbagai komponen yang telah dibuat.

- Benefit

```
"use client";

import React from "react";
import { HoverEffectBenefit } from "../ui/card-hover-effect-benefit";
import { benefitData } from "@/data/benefit.data";

const Benefit = () => {
  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col gap-2">
          <h1 className="text-3xl font-extrabold">Why Share Your Dream?</h1>
          <p className="text-muted-foreground max-w-[800px]">
            Sharing your dreams opens the door to endless possibilities.
            Discover the benefits of expressing your aspirations and turning
            them into reality.
          </p>
        </div>
        <HoverEffectBenefit benefits={benefitData} />
      </div>
    </div>
  );
};

export default Benefit;
```

- CTA

```
import Image from "next/image";
import React from "react";
import { WobbleCard } from "../ui/wobble-card";
import InteractiveHoverButton from "../ui/interactive-hover-button";
import Link from "next/link";

const CTA = () => {
  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="grid grid-cols-1 lg:grid-cols-3 gap-4 max-w-7xl mx-auto w-full">
          <WobbleCard containerClassName="col-span-1 lg:col-span-3 bg-black min-h-[500px]
lg:min-h-[600px] xl:min-h-[300px]">
            <div className="max-w-sm">
              <h2 className="max-w-sm md:max-w-lg  text-left text-balance text-base
md:text-xl lg:text-3xl font-semibold tracking-[-0.015em] text-white">
                Bring Your Vision to Life and Take the First Step Toward
                Achieving Your Dreams!
              </h2>
              <p className="my-4 max-w-[26rem] text-left  text-base/6 text-neutral-200">
                Join a community dedicated to turning dreams into reality. Start
                your journey today and see the difference it makes.
              </p>
              <Link href={"/dreams/create"}>
                <InteractiveHoverButton text="Write it!" />
              </Link>
            </div>
            <Image
              src="/cta.jpg"
              width={500}
              height={500}
              alt="linear demo image"
              className="absolute -right-20 md:-right-[30%] lg:-right-[10%] -bottom-32
md:-bottom-10 object-contain rounded-2xl"
            />
          </WobbleCard>
        </div>
      </div>
    </div>
  );
};

export default CTA;
```

- Footer

```
import { navigationData } from "@/data/navigation.data";
import { socialData } from "@/data/social.data";
import { teamData } from "@/data/team.data";
import Link from "next/link";
import React from "react";

const Footer = () => {
  return (
    <div className="relative py-12 md:py-20 bg-black dark:bg-inherit border-t text-white
mt-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col md:flex-row justify-between gap-10">
          <div className="flex flex-col md:col-span-2 gap-3">
            <Link href={"/"} className="font-bold text-xl">
              Dream Station
            </Link>
            <p className="md:max-w-sm text-justify">
              Dream Station is a platform where aspirations come alive. Share
              your dreams, connect with like-minded individuals, and take steps
              to turn your goals into reality.
            </p>
            <div className="flex flex-row gap-4">
              {socialData.map((social) => {
                return (
                  <Link href={social.href} target="_blank" key={social.id}>
                    <social.icon className="w-6 h-6 hover:text-muted-foreground" />
                  </Link>
                );
              })}
            </div>
          </div>
          <div className="flex flex-col gap-3">
            <h1 className="md:text-lg">Navigations</h1>
            <div className="flex flex-col gap-2">
              {navigationData.map((link) => {
                return (
                  <Link
                    href={link.href}
                    key={link.id}
                    className="text-sm md:text-base hover:text-muted-foreground"
                  >
                    {link.label}
                  </Link>
                );
              })}
            </div>
          </div>
          <div className="flex flex-col gap-3">
            <h1 className="md:text-lg">Teams</h1>
            <div className="flex flex-col gap-2">
              {teamData.map((team) => {
                return (
                  <Link
                    target="_blank"
                    href={team.link}
                    key={team.id}
                    className="text-sm md:text-base hover:text-muted-foreground"
                  >
                    {team.name}
                  </Link>
                );
              })}
            </div>
          </div>
        </div>
      </div>
    </div>
  );
};

export default Footer;
```

- Hero Main

```
import React from "react";
import ShinyButton from "../ui/shiny-button";
import { Button } from "../ui/button";
import { ExternalLink } from "lucide-react";
import Link from "next/link";

const Hero = () => {
  return (
    <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
      <div className="flex sm:min-h-[71vh] min-h-[68vh] flex-col items-center justify-
center text-center px-2 py-8">
        <Link
          href={"https://github.com/zidanindratama/fe-ujian-pemrograman-web"}
          className="mb-5 sm:text-lg flex items-center gap-2 underline underline-offset-4"
        >
          Follow along on GitHub
          <ExternalLink className="w-5 h-5" />
        </Link>
        <h1 className="text-3xl font-bold mb-4 sm:text-7xl">
          Share Your Dream, Shape Your Future
        </h1>
        <p className="mb-8 sm:text-xl max-w-[800px] text-muted-foreground">
          Everyone has a dream worth pursuing—whether it&apos;s becoming a
          skilled programmer, starting a business, or exploring the world.
          Let&apos;s take the first step together. Share your aspirations, and
          let&apos;s bring them to life.
        </p>
        <div className="flex flex-row items-center gap-5">
          <Link href={"/dreams/create"}>
            <ShinyButton>Write it!</ShinyButton>
          </Link>
          <Button variant="link" effect="underline" asChild>
            <Link href={"/about-us"}>Learn More</Link>
          </Button>
        </div>
      </div>
    </div>
  );
};

export default Hero;
```

- Navbar

```
"use client";

import {
  Sheet,
  SheetContent,
  SheetHeader,
  SheetTitle,
  SheetTrigger,
} from "@/components/ui/sheet";
import { Button } from "@/components/ui/button";
import Link from "next/link";
import { navigationData } from "@/data/navigation.data";
import { ArrowRightIcon, Menu } from "lucide-react";
import React from "react";
import { ModeToggle } from "../mode-toggle";

const Navbar = () => {
  return (
    <div className="sticky top-0 p-6 drop-shadow bg-white dark:bg-black z-50">
      <div className="flex flex-row justify-between items-center max-w-7xl mx-auto w-full
text-[#050505] dark:text-white">
        <Link href={"/"} className="font-bold text-xl">
          Dream Station
        </Link>
        <div className="hidden md:flex flex-row gap-12 justify-between items-center">
          {navigationData.map((link) => {
            return (
              <Link
                href={link.href}
                key={link.id}
                className="font-semibold hover:text-muted-foreground"
              >
                {link.label}
              </Link>
            );
          })}
          <ModeToggle />
          <Button
```

```
            effect="expandIcon"
            icon={ArrowRightIcon}
            iconPlacement="right"
            asChild
          >
            <Link href={"/dreams/create"} className="font-bold">
              Join Us
            </Link>
          </Button>
        </div>
        <Sheet>
          <SheetTrigger className="md:hidden">
            <Menu />
          </SheetTrigger>
          <SheetContent className="">
            <SheetHeader>
              <SheetTitle className="text-left">
                <ModeToggle />
              </SheetTitle>
              <div className="flex flex-col gap-4 text-left py-10">
                {navigationData.map((link) => {
                  return (
                    <Link
                      href={link.href}
                      key={link.id}
                      className="font-semibold hover:text-muted-foreground"
                    >
                      {link.label}
                    </Link>
                  );
                })}
                <Button
                  effect="expandIcon"
                  icon={ArrowRightIcon}
                  iconPlacement="right"
                  asChild
                >
                  <Link href={"/dreams"} className="font-bold">
                    Join Us
                  </Link>
                </Button>
              </div>
            </SheetHeader>
          </SheetContent>
        </Sheet>
      </div>
    </div>
  );
};

export default Navbar;
```

- Testimonials

```
import { testimonialData } from "@/data/testimonial.data";
import { cn } from "@/lib/utils";
import { TTestimonial } from "@/types/testimonial.type";
import React from "react";
import { Marquee } from "../ui/marquee";

const ReviewCard = ({ name, image, job, description }: TTestimonial) => {
  return (
    <figure
      className={cn(
        "relative w-64 cursor-pointer overflow-hidden rounded-xl border p-4",
        "border-gray-950/[.1] bg-gray-950/[.01] hover:bg-gray-950/[.05]",
        "dark:border-gray-50/[.1] dark:bg-gray-50/[.10] dark:hover:bg-gray-50/[.15]"
      )}
    >
      <div className="flex flex-row items-center gap-2">
        <img
          className="rounded-full"
          width="32"
          height="32"
          alt=""
          src={image}
        />
        <div className="flex flex-col">
          <figcaption className="text-sm font-medium dark:text-white">
            {name}
          </figcaption>
          <p className="text-xs font-medium dark:text-white/40">{job}</p>
        </div>
      </div>
      <blockquote className="mt-2 text-sm">{description}</blockquote>
    </figure>
```

```
  );
};

const Testimonials = () => {
  const firstRow = testimonialData.slice(0, testimonialData.length / 2);
  const secondRow = testimonialData.slice(testimonialData.length / 2);

  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col text-center mx-auto gap-2">
          <h1 className="text-3xl font-extrabold">What People Are Saying</h1>
          <p className="text-muted-foreground max-w-[800px]">
            Hear from those who have shared their dreams and transformed them
            into reality. Get inspired by their journeys and discover how this
            platform helped shape their success.
          </p>
        </div>
      </div>
      <div className="relative flex h-[500px] w-full flex-col items-center justify-center
overflow-hidden">
        <Marquee pauseOnHover className="[--duration:20s]">
          {firstRow.map((review) => (
            <ReviewCard key={review.id} {...review} />
          ))}
        </Marquee>
        <Marquee reverse pauseOnHover className="[--duration:20s]">
          {secondRow.map((review) => (
            <ReviewCard key={review.id} {...review} />
          ))}
        </Marquee>
        <div className="pointer-events-none absolute inset-y-0 left-0 w-1/3 bg-gradient-
to-r from-white dark:from-background"></div>
        <div className="pointer-events-none absolute inset-y-0 right-0 w-1/3 bg-gradient-
to-l from-white dark:from-background"></div>
      </div>
    </div>
  );
};

export default Testimonials;
```

- Hero About Us

```
import React from "react";
import Ripple from "../ui/ripple";

const HeroAboutUs = () => {
  return (
    <div className="relative flex sm:min-h-[71vh] min-h-[68vh] w-full flex-col items-
center justify-center overflow-hidden">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col items-center justify-center text-center px-2 py-8">
          <h1 className="text-3xl font-bold mb-4 sm:text-7xl">
            Discover the Heart of Our Mission and the People Behind It
          </h1>
          <p className="mb-8 sm:text-xl max-w-[800px] text-muted-foreground">
            At Dream Station, we believe in the power of dreams. Our mission is
            to create a space where aspirations turn into action, and
            possibilities become realities. Join us in shaping a brighter
            future, one dream at a time.
          </p>
        </div>
      </div>
      <Ripple />
    </div>
  );
};

export default HeroAboutUs;
```

- History

```
import React from "react";
import { HeroVideoDialog } from "../ui/hero-video-dialog";

const History = () => {
  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col gap-2">
          <h1 className="text-3xl font-extrabold">
            Our Journey Through Dreams
          </h1>
          <p className="text-muted-foreground max-w-[800px]">
            Dream Station started as a simple idea: to build a space where
            dreams are shared and realized. Today, we empower individuals to
            turn aspirations into achievements.
          </p>
        </div>
        <div className="relative mt-10">
          <HeroVideoDialog
            className="dark:hidden block"
            animationStyle="from-center"
            videoSrc="https://www.youtube.com/embed/Td7JBUcyql8?si=3fLLwOmGH4dfq7N5"
            thumbnailSrc="histories/history-3.jpg"
            thumbnailAlt="Hero Video"
          />
          <HeroVideoDialog
            className="hidden dark:block"
            animationStyle="from-center"
            videoSrc="https://www.youtube.com/embed/Td7JBUcyql8?si=3fLLwOmGH4dfq7N5"
            thumbnailSrc="histories/history-3.jpg"
            thumbnailAlt="Hero Video"
          />
        </div>
      </div>
    </div>
  );
};

export default History;
```

- Team

```
import React from "react";
import { DirectionAwareHoverTeam } from "../ui/direction-aware-hover-team";
import { teamData } from "@/data/team.data";
import Link from "next/link";

const Team = () => {
  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col text-center mx-auto gap-2">
          <h1 className="text-3xl font-extrabold">
            Meet the Visionaries Behind Dream Station
          </h1>
          <p className="text-muted-foreground max-w-[800px]">
            Our team is a group of passionate individuals driven by a shared
            commitment to help people achieve their dreams. Together, we bring
            expertise, creativity, and dedication to turn aspirations into
            impactful realities.
          </p>
        </div>
        <div className="flex flex-col md:flex-row justify-around gap-10 mt-10">
          {teamData.map((team) => {
            return (
              <Link target="_blank" href={team.link} key={team.id}>
                <DirectionAwareHoverTeam imageUrl={team.photo}>
                  <p className="font-bold text-xl">{team.name}</p>
                  <p className="font-normal text-sm">{team.bio}</p>
                </DirectionAwareHoverTeam>
              </Link>
            );
          })}
        </div>
      </div>
    </div>
  );
};

export default Team;
```

- Techs

```
import { backendTechData, frontendTechData } from "@/data/tech.data";
import React from "react";

const Tech = () => {
  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col gap-2">
          <h1 className="text-3xl font-extrabold">Technologies Used</h1>
          <p className="text-muted-foreground max-w-[800px]">
            Our project utilizes a variety of advanced technologies to deliver
            seamless and efficient user experiences.
          </p>
        </div>
        <div className="grid grid-cols-1 gap-5 md:gap-10 mt-10">
          <div className="">
            <h1 className="md:text-lg font-semibold p-2 bg-black text-white dark:bg-white
dark:text-black w-fit">
              Frontend
            </h1>
            <div className="grid grid-cols-2 md:grid-cols-4 gap-10 mt-5">
              {frontendTechData.map((fe) => {
                return (
                  <div key={fe.name} className="flex flex-col items-center">
                    <fe.icon className="w-10 md:w-20 h-10 md:h-20" />
                    <h1 className="font-semibold mt-2">{fe.name}</h1>
                  </div>
                );
              })}
            </div>
          </div>
          <div className="">
            <h1 className="md:text-lg font-semibold p-2 bg-black text-white dark:bg-white
dark:text-black w-fit">
              Backend
            </h1>
            <div className="grid grid-cols-2 md:grid-cols-4 gap-10 mt-5">
              {backendTechData.map((be) => {
                return (
                  <div key={be.name} className="flex flex-col items-center">
                    <be.icon className="w-10 md:w-20 h-10 md:h-20" />
                    <h1 className="font-semibold mt-2">{be.name}</h1>
                  </div>
                );
              })}
            </div>
          </div>
        </div>
      </div>
    </div>
  );
};

export default Tech;
```

- Hero Dreams

```
import React from "react";
import Ripple from "../ui/ripple";
import Link from "next/link";
import ShinyButton from "../ui/shiny-button";

const HeroDreams = () => {
  return (
    <div className="relative flex sm:min-h-[71vh] min-h-[68vh] w-full flex-col items-
center justify-center overflow-hidden">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col items-center justify-center text-center px-2 py-8">
          <h1 className="text-3xl font-bold mb-4 sm:text-7xl">
            Dreams Written Down
          </h1>
          <p className="mb-8 sm:text-xl max-w-[800px] text-muted-foreground">
            Explore the dreams shared by people from all walks of life. Each
            story represents a journey toward achieving something meaningful,
            and we&apos;re here to help turn those dreams into reality.
          </p>
          <Link href={"/dreams/create"}>
            <ShinyButton>Write it!</ShinyButton>
```

```
          </Link>
        </div>
      </div>
      <Ripple />
    </div>
  );
};

export default HeroDreams;
```

- Dreams

```
"use client";

import {
  Select,
  SelectContent,
  SelectItem,
  SelectTrigger,
  SelectValue,
} from "@/components/ui/select";
import React, { useEffect, useState } from "react";
import { HoverEffectDream } from "../ui/card-hover-effect-dream";
import { useFetchData } from "@/hooks/use-fetch-data";
import { useDebounce } from "@/hooks/use-debounce";
import { TDream } from "@/types/dream.type";
import { Input } from "../ui/input";
import { Button } from "../ui/button";

const Dream = () => {
  const [pageIndex, setPageIndex] = useState(0);

  const [totalPages, setTotalPages] = useState(0);
  const [search, setSearch] = useState("");
  const [selectedSortOrderBy, setSelectedSortOrderBy] = useState<string |
null>(
    null
  );

  const debouncedSearch = useDebounce<string>(search, 1000);
  const pageSize = 6;

  const { data: dreamsData, refetch } = useFetchData({
    queryKey: ["dreamsData", pageIndex.toString()],
    dataProtected: `dreams?pgNum=${
      pageIndex + 1
    }&pgSize=${pageSize}&sortOrder=${
      selectedSortOrderBy || ""
    }&title=${debouncedSearch}`,
  });

  useEffect(() => {
    if (dreamsData?.data.meta?.count) {
      setTotalPages(Math.ceil(dreamsData?.data.meta.count / pageSize));
    }
  }, [dreamsData, pageSize]);

  useEffect(() => {
    refetch();
  }, [debouncedSearch, selectedSortOrderBy, pageIndex, refetch]);

  const resetFilter = () => {
    setSearch("");
    setSelectedSortOrderBy(null);
    setPageIndex(0);
  };

  const handlePageChange = (newPageIndex: number) => {
    setPageIndex(newPageIndex);
  };

  const dreamData = dreamsData?.data.dreams as TDream[];
  console.log(dreamData);

  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
```

```
        <div className="flex flex-col gap-2">
          <h1 className="text-3xl font-extrabold">A Glimpse into Dreams</h1>
          <p className="text-muted-foreground max-w-[800px]">
            Dive into an inspiring collection of dreams shared by individuals
            striving to make a difference. These stories remind us of the
            limitless possibilities when ambition meets action.
          </p>
        </div>
        <div className="flex flex-col md:flex-row gap-5 mt-10">
          <Input
            type="text"
            className=""
            placeholder="Search dreams"
            onChange={(e) => setSearch(e.target.value)}
          />
          <div className="flex flex-row gap-5">
            <Select
              onValueChange={(value) => {
                setSelectedSortOrderBy(value);
                setPageIndex(0);
              }}
            >
              <SelectTrigger className="md:w-[180px]">
                <SelectValue
                  placeholder={
                    selectedSortOrderBy !== null
                      ? selectedSortOrderBy
                      : "Sort By"
                  }
                />
              </SelectTrigger>
              <SelectContent>
                <SelectItem value="newest">Newest</SelectItem>
                <SelectItem value="oldest">Oldest</SelectItem>
              </SelectContent>
            </Select>
            <Button variant={"destructive"} onClick={resetFilter}>
              Reset
            </Button>
          </div>
        </div>
        <div className="h-[1px] w-full bg-[#D0D1D3] my-5"></div>
        <HoverEffectDream dreams={dreamData} />
        <div className="flex justify-center gap-3 mt-8">
          <Button
            variant={"outline"}
            onClick={() => handlePageChange(pageIndex - 1)}
            disabled={pageIndex === 0}
          >
            Previous
          </Button>
          <Button
            variant={"outline"}
            onClick={() => handlePageChange(pageIndex + 1)}
            disabled={pageIndex + 1 === totalPages}
          >
            Next
          </Button>
        </div>
      </div>
    </div>
  );
};

export default Dream;
```

- Dream Create Form

```
/* eslint-disable  @typescript-eslint/no-explicit-any */
"use client";

import { dreamSchema, TDreamForm } from "@/schema/dream.schema";
import { zodResolver } from "@hookform/resolvers/zod";
import { useForm } from "react-hook-form";
import { Button } from "@/components/ui/button";
import {
  Form,
```

```
  FormControl,
  FormDescription,
  FormField,
  FormItem,
  FormLabel,
  FormMessage,
} from "@/components/ui/form";
import { Input } from "@/components/ui/input";
import { Textarea } from "@/components/ui/textarea";
import { FileUploader } from "@/components/ui/file-uploader";
import { useAddData } from "@/hooks/use-add-data";
import { toast } from "sonner";

const DreamCreateForm = () => {
  const form = useForm<TDreamForm>({
    resolver: zodResolver(dreamSchema),
    defaultValues: {
      username: "",
      title: "",
      description: "",
      image: [],
    },
  });

  const mutationCreateDream = useAddData({
    queryKey: "dreamData",
    dataProtected: `dreams`,
    backUrl: `/dreams`,
    multipart: true,
  });

  const onSubmit = (values: TDreamForm) => {
    const form = new FormData();

    if (values.image.length > 0) {
      const appendIfNotNull = (key: string, value: any) => {
        if (value != null) {
          form.append(key, value);
        }
      };

      appendIfNotNull("title", values.title);
      appendIfNotNull("username", values.username);
      appendIfNotNull("description", values.description);

      if (values.image[0] !== undefined) {
        form.append("image", values.image[0]);
      }

      mutationCreateDream.mutate(form);
    } else {
      toast.error("Image cover should not be empty!");
    }
  };

  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col gap-2 mx-auto text-center">
          <h1 className="text-3xl font-extrabold">Create Your Dream</h1>
          <p className="text-muted-foreground max-w-[800px]">
            Share your aspirations and bring your dreams to life. Fill out the
            form below to start turning your vision into reality.
          </p>
        </div>
        <Form {...form}>
          <form
            onSubmit={form.handleSubmit(onSubmit)}
            className="space-y-5 mt-10 max-w-md mx-auto w-full"
          >
            <FormField
              control={form.control}
              name="username"
              render={({ field }) => (
                <FormItem>
                  <FormLabel>Username</FormLabel>
                  <FormControl>
                    <Input placeholder="eg: zidanindratama" {...field} />
                  </FormControl>
                  <FormDescription>
                    This is your public display username.
                  </FormDescription>
                  <FormMessage />
                </FormItem>
              )}
            />
            <FormField
              control={form.control}
              name="title"
              render={({ field }) => (
                <FormItem>
                  <FormLabel>Title</FormLabel>
```

```
                  <FormControl>
                    <Input
                      placeholder="eg: An expert Frontend Engineer"
                      {...field}
                    />
                  </FormControl>
                  <FormMessage />
                </FormItem>
              )}
            />
            <FormField
              control={form.control}
              name="description"
              render={({ field }) => (
                <FormItem>
                  <FormLabel>Description</FormLabel>
                  <FormControl>
                    <Textarea
                      placeholder="Tell us a little bit about your dream!"
                      className="resize-none"
                      {...field}
                    />
                  </FormControl>
                  <FormMessage />
                </FormItem>
              )}
            />
            <FormField
              control={form.control}
              name="image"
              render={({ field }) => (
                <div className="space-y-6">
                  <FormItem className="w-full">
                    <FormLabel>Image</FormLabel>
                    <FormControl>
                      <FileUploader
                        value={field.value}
                        onValueChange={field.onChange}
                        maxFileCount={4}
                        maxSize={4 * 1024 * 1024}
                      />
                    </FormControl>
                    <FormMessage />
                  </FormItem>
                </div>
              )}
            />
            <Button type="submit" className="w-full md:w-fit">
              Submit
            </Button>
          </form>
        </Form>
      </div>
    </div>
  );
};

export default DreamCreateForm;
```

- Dream Update Form

```
/* eslint-disable  @typescript-eslint/no-explicit-any */
"use client";

import { dreamSchema, TDreamForm } from "@/schema/dream.schema";
import { zodResolver } from "@hookform/resolvers/zod";
import { useForm } from "react-hook-form";
import { Button } from "@/components/ui/button";
import {
  Form,
  FormControl,
  FormDescription,
  FormField,
  FormItem,
  FormLabel,
  FormMessage,
} from "@/components/ui/form";
import { Input } from "@/components/ui/input";
import { Textarea } from "@/components/ui/textarea";
import { useFetchData } from "@/hooks/use-fetch-data";
import { useUpdateData } from "@/hooks/use-update-data";
import { useEffect } from "react";
import { useDeleteData } from "@/hooks/use-delete-data";

type Props = {
  slug: string;
```

```
};

const DreamUpdateForm = ({ slug }: Props) => {
  const { data: dreamData } = useFetchData({
    queryKey: ["dreamData"],
    dataProtected: `dreams/${slug}`,
  });

  const dream = dreamData?.data as TDreamForm;

  const form = useForm<TDreamForm>({
    resolver: zodResolver(dreamSchema),
    mode: "onChange",
    values: {
      username: dream?.username || "",
      title: dream?.title || "",
      description: dream?.description || "",
      image: dream?.image || "",
    },
  });

  useEffect(() => {
    if (dream) {
      form.reset(dream);
    }
  }, [dream, form]);

  const imageRef = form.register("image");

  const mutationUpdateDream = useUpdateData({
    queryKey: "dreamData",
    dataProtected: `dreams/${slug}`,
    backUrl: `/dreams`,
  });

  const onSubmit = async (values: TDreamForm) => {
    const form = new FormData();

    const isFileList =
      values.image instanceof FileList && values.image.length > 0;

    const appendIfNotNull = (key: string, value: any) => {
      if (value != null) {
        form.append(key, value);
      }
    };

    appendIfNotNull("title", values.title);
    appendIfNotNull("description", values.description);
    appendIfNotNull("username", values.username);

    if (isFileList) {
      form.append("image", values.image[0]);
    } else if (typeof values.image === "string") {
      appendIfNotNull("image", values.image);
    }

    mutationUpdateDream.mutate(form);
  };

  const mutationDreamDelete = useDeleteData({
    queryKey: "dreamsData",
    dataProtected: `dreams/${slug}`,
    backUrl: `/dreams`,
  });

  const handleDelete = (e: any) => {
    e.preventDefault();
    mutationDreamDelete.mutate();
  };

  return (
    <div className="relative py-12 md:py-20">
      <div className="flex flex-col max-w-7xl justify-center mx-auto px-6">
        <div className="flex flex-col gap-2 mx-auto text-center">
          <h1 className="text-3xl font-extrabold">Update Your Dream</h1>
          <p className="text-muted-foreground max-w-[800px]">
            Reimagine your aspirations and keep your vision alive. Update the
            details below to ensure your dream reflects your latest goals and
            ambitions.
          </p>
        </div>
        <div className="flex flex-row md:hidden w-full justify-end mt-10 mb-5">
          <Button
            type="button"
            onClick={handleDelete}
            variant="destructive"
            effect="gooeyRight"
          >
            Delete
          </Button>
        </div>
```

```
        <Form {...form}>
          <form
            onSubmit={form.handleSubmit(onSubmit)}
            className="space-y-5 max-w-md mx-auto w-full"
          >
            <FormField
              control={form.control}
              name="username"
              render={({ field }) => (
                <FormItem>
                  <FormLabel>Username</FormLabel>
                  <FormControl>
                    <Input placeholder="eg: zidanindratama" {...field} />
                  </FormControl>
                  <FormDescription>
                    This is your public display username.
                  </FormDescription>
                  <FormMessage />
                </FormItem>
              )}
            />
            <FormField
              control={form.control}
              name="title"
              render={({ field }) => (
                <FormItem>
                  <FormLabel>Title</FormLabel>
                  <FormControl>
                    <Input
                      placeholder="eg: An expert Frontend Engineer"
                      {...field}
                    />
                  </FormControl>
                  <FormMessage />
                </FormItem>
              )}
            />
            <FormField
              control={form.control}
              name="description"
              render={({ field }) => (
                <FormItem>
                  <FormLabel>Description</FormLabel>
                  <FormControl>
                    <Textarea
                      placeholder="Tell us a little bit about your dream!"
                      className="resize-none"
                      {...field}
                    />
                  </FormControl>
                  <FormMessage />
                </FormItem>
              )}
            />
            <FormField
              control={form.control}
              name="image"
              render={() => (
                <div className="space-y-6">
                  <FormItem className="w-full">
                    <FormLabel>Image</FormLabel>
                    <FormControl>
                      <Input id="picture" type="file" {...imageRef} />
                    </FormControl>
                    <FormMessage />
                  </FormItem>
                </div>
              )}
            />
            <div className="flex gap-3 flex-col md:flex-row justify-between">
              <Button
                type="button"
                onClick={handleDelete}
                variant="destructive"
                effect="gooeyRight"
                className="hidden md:block"
              >
                Delete
              </Button>
              <Button type="submit" className="w-full md:w-fit">
                Submit
              </Button>
            </div>
          </form>
        </Form>
      </div>
    </div>
  );
};

export default DreamUpdateForm;
```

3.2 Backend

3.2.1 Instalasi Project

Instalasi project adalah tahap awal dalam pengembangan backend. Pada bagian ini, dijelaskan langkah-langkah mengatur lingkungan pengembangan, menginstal dependensi yang diperlukan, dan menyiapkan konfigurasi awal.

- Instalasi NestJS

```
D:\Personal Projects\ujian-web>nest new be-ujian-web
⚡  We will scaffold your app in a few seconds..

? Which package manager would you ❤️  to use? npm
CREATE be-ujian-web/.eslintrc.js (688 bytes)
CREATE be-ujian-web/.prettierrc (54 bytes)
CREATE be-ujian-web/nest-cli.json (179 bytes)
CREATE be-ujian-web/package.json (2020 bytes)
CREATE be-ujian-web/README.md (4454 bytes)
CREATE be-ujian-web/tsconfig.build.json (101 bytes)
CREATE be-ujian-web/tsconfig.json (567 bytes)
CREATE be-ujian-web/src/app.controller.ts (286 bytes)
CREATE be-ujian-web/src/app.module.ts (259 bytes)
CREATE be-ujian-web/src/app.service.ts (150 bytes)
CREATE be-ujian-web/src/main.ts (216 bytes)
CREATE be-ujian-web/src/app.controller.spec.ts (639 bytes)
CREATE be-ujian-web/test/jest-e2e.json (192 bytes)
CREATE be-ujian-web/test/app.e2e-spec.ts (654 bytes)

√ Installation in progress... ☕

🚀  Successfully created project be-ujian-web
👉  Get started with the following commands:

$ cd be-ujian-web
$ npm run start


                        Thanks for installing Nest 🙏
                Please consider donating to our open collective
                       to help us maintain this package.


                🍷  Donate: https://opencollective.com/nest
```

- Instalasi Prisma

```
D:\Personal Projects\ujian-web\be-ujian-web>npm install prisma @prisma/client

added 7 packages, and audited 707 packages in 2m

116 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

- Instalasi Cloudinary

```
D:\Personal Projects\ujian-web\be-ujian-web>npm install cloudinary streamifier --save-dev @types/multer
npm WARN deprecated q@1.5.1: You or someone you depend on is using Q, the JavaScript Promise library that gave JavaScript developers strong feelings about promises. They c
an almost certainly migrate to the native JavaScript promise now. Thank you literally everyone for joining me in this bet against the odds. Be excellent to each other.
npm WARN deprecated
npm WARN deprecated (For a CapTP with native promises, see @endo/eventual-send and @endo/captp)

added 4 packages, and audited 716 packages in 3s

116 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

- Instalasi Class Validator dan Class Transformer

```
D:\Personal Projects\ujian-web\be-ujian-web>npm install class-validator class-transformer

added 5 packages, and audited 712 packages in 4s

116 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

3.2.2 Prisma Schema

Prisma schema adalah inti dari pengelolaan database dalam proyek ini. Bagian ini menjelaskan bagaimana skema database dirancang.

- Schema Database Prisma

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

// Looking for ways to speed up your queries, or scale easily with your serverless or edge functions?
// Try Prisma Accelerate: https://pris.ly/cli/accelerate-init

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

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

model Dream {
  id          String   @id @default(auto()) @map("_id") @db.ObjectId
  username    String
  title       String   @unique
  slug        String?  @unique
  description String
  image       String?
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt

  @@map("dreams")
}
```

3.2.3 Modules

Modules adalah komponen utama dalam struktur NestJS yang membantu pengelolaan kode secara modular. Pada bagian ini, akan dijelaskan cara membagi fungsionalitas ke dalam modul-modul terpisah dan memastikan integrasi antar modul berjalan dengan baik.

- Cloudinary
  - Cloudinary Response

```
import { UploadApiErrorResponse, UploadApiResponse } from 'cloudinary';

export type CloudinaryResponse = UploadApiResponse | UploadApiErrorResponse;
```

  - Cloudinary Provider

```
// cloudinary.provider.ts

import { v2 as cloudinary } from 'cloudinary';

export const CloudinaryProvider = {
  provide: 'CLOUDINARY',
  useFactory: () => {
    return cloudinary.config({
      cloud_name: `${process.env.CLOUDINARY_NAME}`,
      api_key: `${process.env.CLOUDINARY_API_KEY}`,
      api_secret: `${process.env.CLOUDINARY_API_SECRET}`,
    });
  },
};
```

  - Cloudinary Module

```
import { Module } from '@nestjs/common';
import { CloudinaryService } from './cloudinary.service';
import { CloudinaryProvider } from './cloudinary/cloudinary.provider';

@Module({
  providers: [CloudinaryProvider, CloudinaryService],
  exports: [CloudinaryProvider, CloudinaryService],
})
export class CloudinaryModule {}
```

- Cloudinary Service

```
import { HttpException, Injectable } from '@nestjs/common';
import { v2 as cloudinary } from 'cloudinary';
import { CloudinaryResponse } from './cloudinary/cloudinary-response';
const streamifier = require('streamifier');

@Injectable()
export class CloudinaryService {
  uploadFile(
    file: Express.Multer.File,
    onlyImage: boolean,
  ): Promise<CloudinaryResponse> {
    if (!file) throw new HttpException('File not found!', 404);

    const fileSizeLimit = 10 * 1024 * 1024;
    if (file.size > fileSizeLimit)
      throw new HttpException('File size exceeds 10 MB limit.', 400);

    const allowedExtensions = ['jpeg', 'jpg', 'png', 'JPEG', 'JPG', 'PNG'];
    const [fileType, fileExtension] = file.mimetype.split('/');

    if (onlyImage === true)
      if (!allowedExtensions.includes(fileExtension))
        throw new HttpException(
          'File type not supported. Only JPEG, JPG, and PNG files are allowed.',
          400,
        );

    return new Promise<CloudinaryResponse>((resolve, reject) => {
      const uploadStream = cloudinary.uploader.upload_stream(
        { folder: 'DREAM-STATION' },
        (error, result) => {
          if (error) return reject(error);
          resolve(result);
        },
      );

      streamifier.createReadStream(file.buffer).pipe(uploadStream);
    });
  }
}
```

- Dream
  - Dream Controller

```
import {
  Body,
  Controller,
  Delete,
  Get,
  HttpException,
  HttpStatus,
  Param,
  Patch,
  Post,
  Query,
  UploadedFile,
  UseInterceptors,
} from '@nestjs/common';
import { FileInterceptor } from '@nestjs/platform-express';
import { DreamsService } from './dreams.service';
import { QueryDreamsDto } from './dtos/query-dreams.dto';
import { CreateDreamDto } from './dtos/create-dream.dto';
import { PrismaService } from '../prisma/prisma.service';
import { CloudinaryService } from '../cloudinary/cloudinary.service';
import { UpdateDreamDto } from './dtos/update-dream-dto';

const slugify = (title: string) => {
  return title
    .toLowerCase()
    .replace(/\s+/g, '-')
    .replace(/[^\w\-]+/g, '')
    .trim();
};

@Controller('/api/dreams')
export class DreamsController {
  constructor(
    private dreamsService: DreamsService,
    private prismaService: PrismaService,
    private readonly cloudinaryService: CloudinaryService,
```

```
  ) {}

  @Get()
  getAllDreams(@Query() query: QueryDreamsDto) {
    return this.dreamsService.getAllDreams(query);
  }

  @Get('/:slug')
  async getDreamBySlug(@Param('slug') slug: string) {
    const category = await this.dreamsService.getDreamBySlug(slug);
    if (!category) throw new HttpException('Dream not found!', 404);
    return category;
  }

  @Post()
  @UseInterceptors(FileInterceptor('image'))
  async createDream(
    @Body() body: CreateDreamDto,
    @UploadedFile() image: Express.Multer.File,
  ) {
    try {
      const slug = slugify(body.title);
      const dream = await this.dreamsService.getDreamBySlug(slug);
      if (dream)
        throw new HttpException('Dream already exist!', HttpStatus.BAD_REQUEST);

      if (image) {
        const uploadedImage = await this.cloudinaryService.uploadFile(
          image,
          true,
        );
        body.image = uploadedImage.secure_url;
      }

      return this.dreamsService.createDream(body);
    } catch (error) {
      throw new HttpException(
        error.message || 'Failed to create dream',
        HttpStatus.BAD_REQUEST,
      );
    }
  }

  @Patch('/:slug')
  @UseInterceptors(FileInterceptor('image'))
  async updateDreamBySlug(
    @Param('slug') slug: string,
    @Body() body: UpdateDreamDto,
    @UploadedFile() image: Express.Multer.File,
  ) {
    try {
      const dream = await this.dreamsService.getDreamBySlug(slug);
      if (!dream) throw new HttpException('Dream not found!', 404);

      if (image) {
        const uploadedImage = await this.cloudinaryService.uploadFile(
          image,
          true,
        );
        body.image = uploadedImage.secure_url;
      }

      const slugUpdate = slugify(body.title);
      return this.dreamsService.updateDreamBySlug(slug, slugUpdate, body);
    } catch (error) {
      throw new HttpException(
        error.message || 'Failed to update dream',
        HttpStatus.BAD_REQUEST,
      );
    }
  }

  @Delete('/:slug')
  async deleteBlogBySlug(@Param('slug') slug: string) {
    const dream = await this.prismaService.dream.findUnique({
      where: {
        slug: slug,
      },
    });
    if (!dream) throw new HttpException('Drea not found!', 404);
    return this.dreamsService.deleteDreamBySlug(slug);
  }
}
```

- Dream Module

```
import { Module } from '@nestjs/common';
import { DreamsService } from './dreams.service';
import { DreamsController } from './dreams.controller';
import { PrismaModule } from '../prisma/prisma.module';
import { CloudinaryModule } from '../cloudinary/cloudinary.module';

@Module({
  imports: [PrismaModule, CloudinaryModule],
  providers: [DreamsService],
  controllers: [DreamsController],
  exports: [DreamsService],
})
export class DreamsModule {}
```

- Dream Service

```
import { HttpException, Injectable } from '@nestjs/common';
import { PrismaService } from '../prisma/prisma.service';
import { QueryDreamsDto } from './dtos/query-dreams.dto';
import { Prisma } from '@prisma/client';

const slugify = (title: string) => {
  return title
    .toLowerCase()
    .replace(/\s+/g, '-')
    .replace(/[^\w\-]+/g, '')
    .trim();
};

@Injectable()
export class DreamsService {
  constructor(private prismaService: PrismaService) {}

  async getAllDreams(query: QueryDreamsDto) {
    const pgNum = +(query.pgNum ?? 1);
    const pgSize = +(query.pgSize ?? 10);
    const skip = (pgNum - 1) * pgSize;
    const take = pgSize;

    const where: Prisma.DreamWhereInput = {
      ...(query.title && {
        title: { contains: query.title, mode: 'insensitive' },
      }),
    };

    const orderBy: Prisma.DreamOrderByWithAggregationInput =
      query.sortOrder === 'newest'
        ? { createdAt: 'desc' }
        : query.sortOrder === 'oldest'
          ? { createdAt: 'asc' }
          : { createdAt: 'desc' };

    const dreams = await this.prismaService.dream.findMany({
      skip,
      take,
      where,
      orderBy,
    });

    const dreamsCount = await this.prismaService.dream.count({ where });

    return {
      dreams,
      meta: {
        count: dreamsCount,
      },
    };
  }

  getDreamBySlug(slug: string) {
    return this.prismaService.dream.findUnique({
      where: {
        slug,
      },
    });
  }

  async createDream(createDreamData: Prisma.DreamCreateInput) {
    const slug = slugify(createDreamData.title);
    const category = await this.getDreamBySlug(slug);
    if (category) throw new HttpException('Dream already exist!', 401);
```

```
    return this.prismaService.dream.create({
      data: {
        ...createDreamData,
        slug,
      },
    });
  }

  updateDreamBySlug(
    slug: string,
    slugUpdate: string,
    updateDreamByIdData: Prisma.DreamUpdateInput,
  ) {
    return this.prismaService.dream.update({
      where: {
        slug,
      },
      data: {
        ...updateDreamByIdData,
        slug: slugUpdate,
      },
    });
  }

  async deleteDreamBySlug(slug: string) {
    await this.prismaService.dream.delete({
      where: {
        slug,
      },
    });

    return {
      message: 'Successfully delete the dream!',
      statusCode: 200,
    };
  }
}
```

- Create Dream DTO

```
import {
  IsOptional,
  IsString,
  IsUrl,
  IsNotEmpty,
  MinLength,
  MaxLength,
} from 'class-validator';

export class CreateDreamDto {
  @IsString({ message: 'Username must be a string' })
  @IsNotEmpty({ message: 'Username cannot be empty' })
  @MinLength(1, { message: 'Username must be at least 1 character long' })
  @MaxLength(50, { message: 'Username cannot exceed 50 characters' })
  username: string;

  @IsString({ message: 'Title must be a string' })
  @IsNotEmpty({ message: 'Title cannot be empty' })
  @MinLength(1, { message: 'Title must be at least 1 character long' })
  @MaxLength(50, { message: 'Title cannot exceed 50 characters' })
  title: string;

  @IsString({ message: 'Description must be a string' })
  @IsNotEmpty({ message: 'Description cannot be empty' })
  @MinLength(1, { message: 'Description must be at least 1 character long' })
  @MaxLength(150, { message: 'Description cannot exceed 150 characters' })
  description: string;

  @IsOptional()
  @IsUrl({}, { message: 'Image must be a valid URL' })
  image?: string;
}
```

- Query Dream DTO

```
import { IsOptional, IsString } from 'class-validator';

export class QueryDreamsDto {
  @IsOptional()
  @IsString()
  title?: string;

  @IsString()
  @IsOptional()
  sortOrder?: 'newest' | 'oldest';

  @IsString()
  @IsOptional()
  pgNum?: string;

  @IsString()
  @IsOptional()
  pgSize?: string;
}
```

- Update Dream DTO

```
import {
  IsOptional,
  IsString,
  MinLength,
  MaxLength,
  IsUrl,
} from 'class-validator';

export class UpdateDreamDto {
  @IsOptional()
  @IsString({ message: 'Username must be a string' })
  @MinLength(1, { message: 'Username must be at least 1 character long' })
  @MaxLength(50, { message: 'Username cannot exceed 50 characters' })
  username?: string;

  @IsOptional()
  @IsString({ message: 'Title must be a string' })
  @MinLength(1, { message: 'Title must be at least 1 character long' })
  @MaxLength(50, { message: 'Title cannot exceed 50 characters' })
  title?: string;

  @IsOptional()
  @IsString({ message: 'Description must be a string' })
  @MinLength(1, { message: 'Description must be at least 1 character long' })
  @MaxLength(150, { message: 'Description cannot exceed 150 characters' })
  description?: string;

  @IsOptional()
  @IsUrl({}, { message: 'Image must be a valid URL' })
  image?: string;
}
```

- Prisma
  - Prisma Module

```
import { Module } from '@nestjs/common';
import { PrismaService } from './prisma.service';

@Module({
  providers: [PrismaService],
  exports: [PrismaService],
})
export class PrismaModule {}
```

- Prisma Service

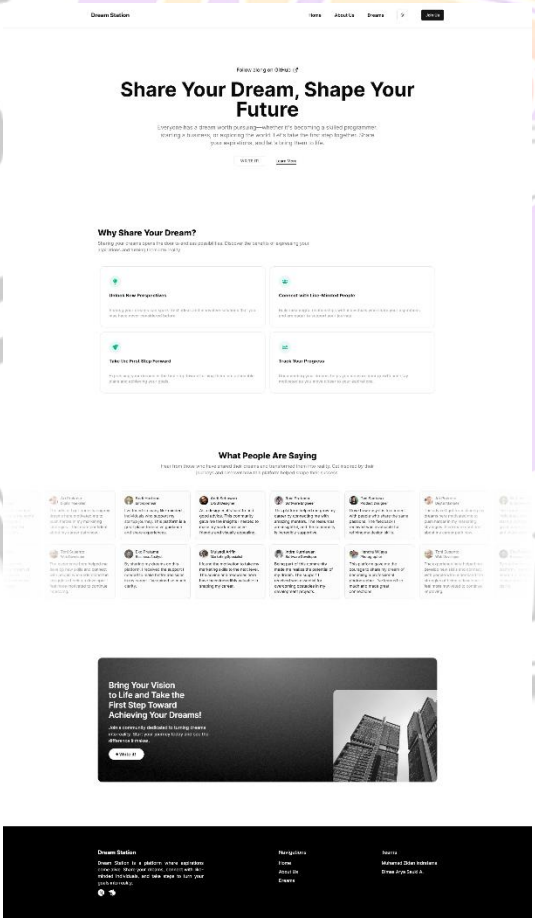
```
import { Injectable, OnModuleInit } from '@nestjs/common';
import { PrismaClient } from '@prisma/client';

@Injectable()
export class PrismaService extends PrismaClient implements OnModuleInit {
  onModuleInit() {
    this.$connect()
      .then(() => console.log('Connected to DB with Prisma!'))
      .catch((err) => {
        console.log(err);
      });
  }
}
```

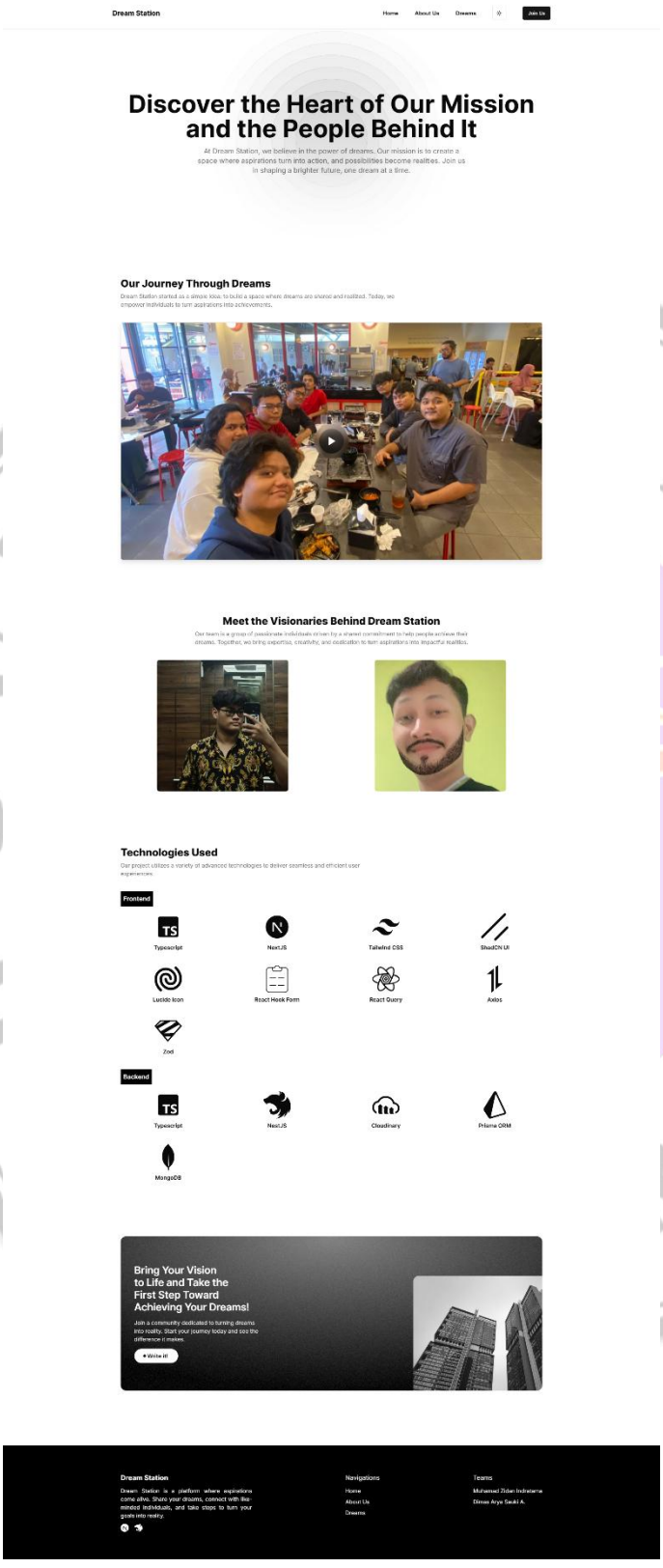
3.3 Output

Output merupakan hasil akhir dari sistem yang telah dirancang dan dikembangkan. Bagian ini akan menjelaskan bagaimana data yang telah diinput oleh pengguna ditampilkan kembali melalui antarmuka pengguna, serta bagaimana informasi tersebut dapat dimanfaatkan untuk mencapai tujuan website. Penjelasan ini mencakup visualisasi mimpi yang dibagikan, format tampilan, dan bagaimana data tersebut memberikan motivasi kepada pengguna lain.

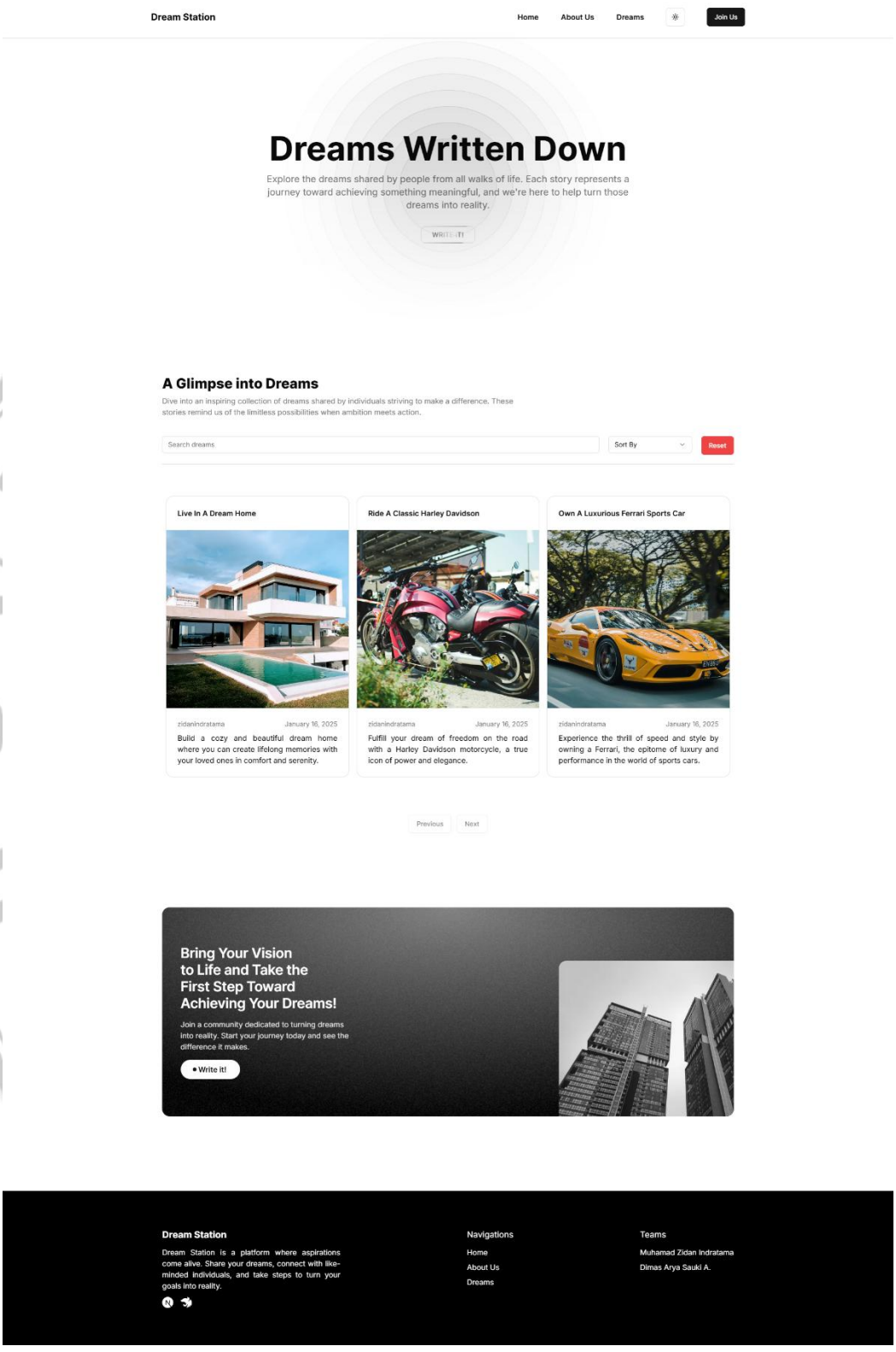
- Landing



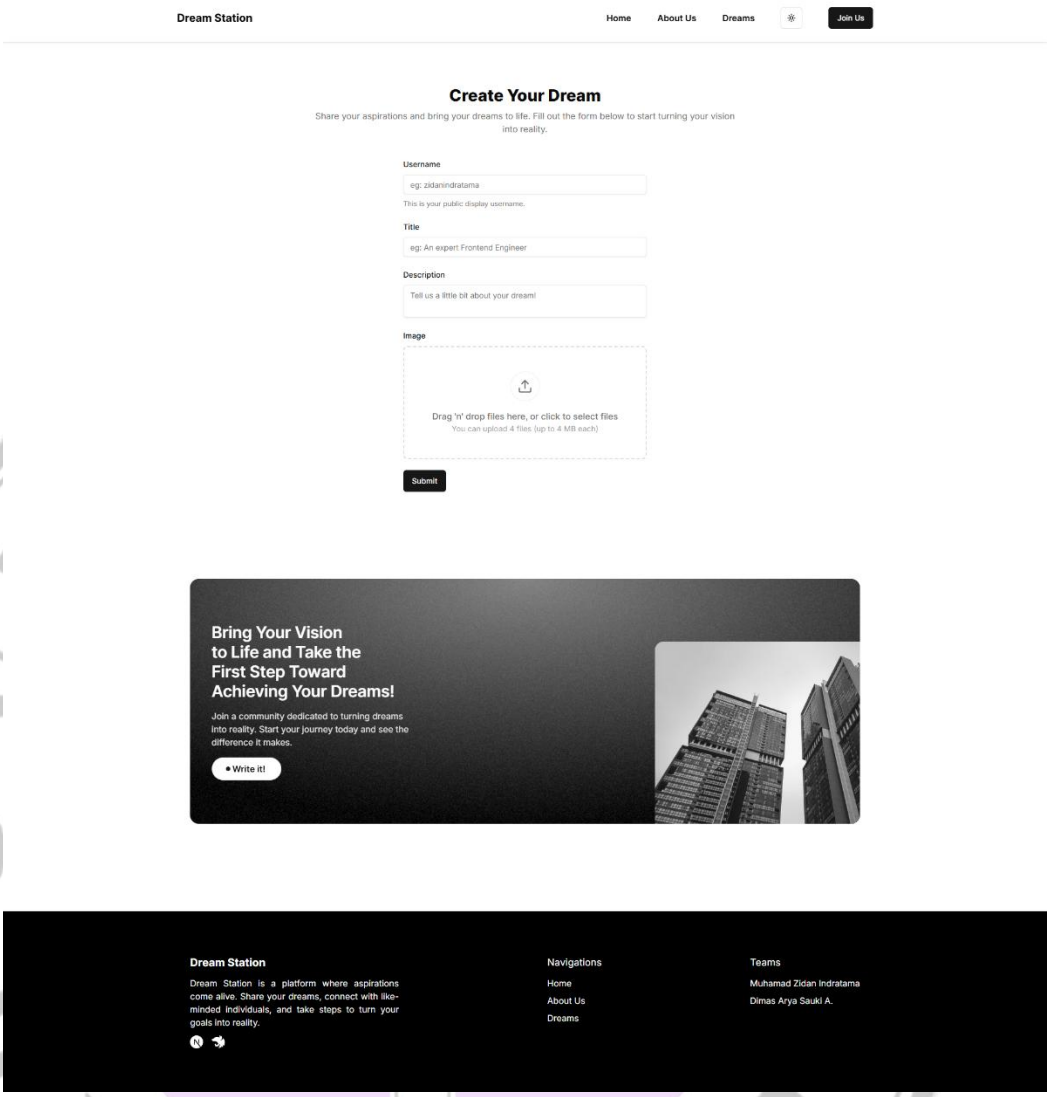
- About Us


Dream Station
Home
About Us
Dreams
Join Us
Discover the Heart of Our Mission and the People Behind It
At Dream Station, we believe in the power of dreams. Our mission is to create a space where aspirations turn into action, and possibilities become realities. Join us in shaping a brighter future, one dream at a time.
Our Journey Through Dreams
Dream Station started as a simple idea: to build a space where dreams are shared and realized. Today, we empower individuals to turn aspirations into achievements.
Meet the Visionaries Behind Dream Station
Our team is a group of passionate individuals driven by a shared commitment to help people achieve their dreams. Together, we bring expertise, creativity, and dedication to turn aspirations into impactful realities.
Technologies Used
Our project utilizes a variety of advanced technologies to deliver seamless and efficient user experiences.
Frontend
Typescript
NextJS
Tailwind CSS
ShadCN UI
Lucide Icon
React Hook Form
React Query
Axios
Zod
Backend
Typescript
NextJS
Cloudinary
Prisma ORM
MongoDB
Bring Your Vision to Life and Take the First Step Toward Achieving Your Dreams!
Join a community dedicated to turning dreams into reality. Start your journey today and see the difference it makes.
Write it!
Dream Station
Dream Station is a platform where aspirations come alive. Share your dreams, connect with like-minded individuals, and take steps to turn your goals into reality.
Navigations
Home
About Us
Dreams
Teams
Muhamad Zidan Indratama
Dimas Arya Seuki A.

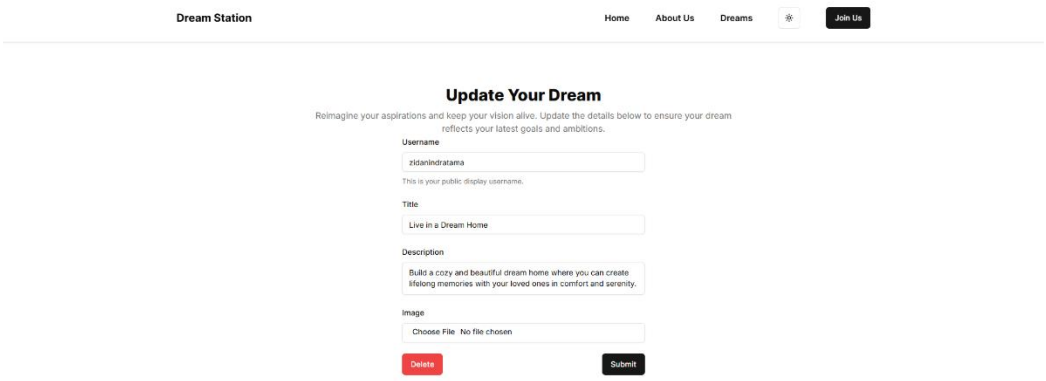
- Dreams

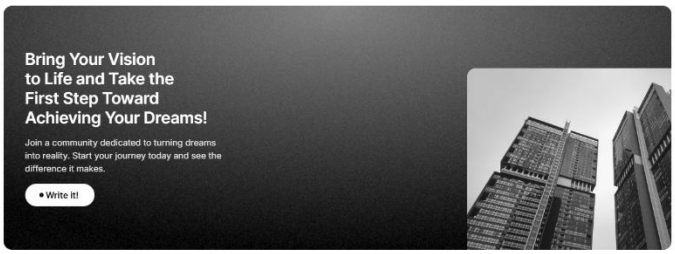

Dream Station
Home
About Us
Dreams
Join Us
Dreams Written Down
Explore the dreams shared by people from all walks of life. Each story represents a journey toward achieving something meaningful, and we're here to help turn those dreams into reality.
A Glimpse into Dreams
Dive into an inspiring collection of dreams shared by individuals striving to make a difference. These stories remind us of the limitless possibilities when ambition meets action.
Search dreams
Sort By
Reset
Live In A Dream Home
Ride A Classic Harley Davidson
Own A Luxurious Ferrari Sports Car
zidanindratama
January 16, 2025
Build a cozy and beautiful dream home where you can create lifelong memories with your loved ones in comfort and serenity.
Fulfill your dream of freedom on the road with a Harley Davidson motorcycle, a true icon of power and elegance.
Experience the thrill of speed and style by owning a Ferrari, the epitome of luxury and performance in the world of sports cars.
Previous
Next
Bring Your Vision to Life and Take the First Step Toward Achieving Your Dreams!
Join a community dedicated to turning dreams into reality. Start your journey today and see the difference it makes.
Write it!
Dream Station
Dream Station is a platform where aspirations come alive. Share your dreams, connect with like-minded individuals, and take steps to turn your goals into reality.
Navigations
Home
About Us
Dreams
Teams
Muhamad Zidan Indratama
Dimas Arya Sauki A.

- Dream Create



- Dream Update





3.4 Link

- Front-end
  https://fe-ujian-pemrograman-web.vercel.app

- Back-end
  https://be-ujian-pemrograman-web.vercel.app

BAB IV

PENUTUP

4.1 Kesimpulan

Dalam proyek ini, telah dirancang dan dikembangkan sebuah website yang memungkinkan pengguna untuk mencatat, membagikan, dan melihat mimpi-mimpi dari pengguna lain. Dengan memanfaatkan teknologi modern baik di sisi frontend maupun backend, website ini tidak hanya memberikan pengalaman pengguna yang optimal tetapi juga mampu menjadi media motivasi yang inspiratif.

Melalui fitur pencatatan mimpi, berbagi motivasi, dan komunitas yang saling mendukung, platform ini bertujuan untuk mengingatkan pengguna akan mimpi mereka serta membangun semangat bersama untuk meraih tujuan hidup. Pemanfaatan teknologi seperti NextJS, TailwindCSS, NestJS, dan Prisma menjadikan sistem ini responsif, aman, dan terstruktur dengan baik.

Dengan platform ini, diharapkan pengguna tidak hanya mampu mengingat mimpi mereka, tetapi juga terinspirasi untuk terus berusaha dan mendukung orang lain dalam perjalanan mencapai mimpi tersebut.

4.2 Saran

Untuk pengembangan lebih lanjut, berikut beberapa saran yang dapat dipertimbangkan:

1. Pengembangan Fitur Tambahan

   Untuk meningkatkan nilai dan manfaat platform ini, pengembang dapat mempertimbangkan penambahan fitur seperti notifikasi pengingat, sistem komentar atau dukungan antar pengguna, serta gamifikasi untuk meningkatkan keterlibatan pengguna.

2. Optimisasi Performansi

   Sebaiknya dilakukan pengujian berkala untuk memastikan performansi website tetap optimal, terutama jika jumlah pengguna dan data meningkat.

3. Peningkatan Keamanan

   Pengamanan data pengguna dapat terus ditingkatkan dengan menambahkan fitur seperti autentikasi dua faktor atau enkripsi data.

4. Kolaborasi dengan Komunitas

   Mengintegrasikan platform dengan komunitas atau organisasi terkait dapat membantu memperluas jangkauan pengguna dan menciptakan dampak yang lebih besar.

Dengan melanjutkan pengembangan dan perbaikan, platform ini memiliki potensi besar untuk menjadi alat yang bermanfaat bagi banyak orang dalam mewujudkan mimpi mereka.