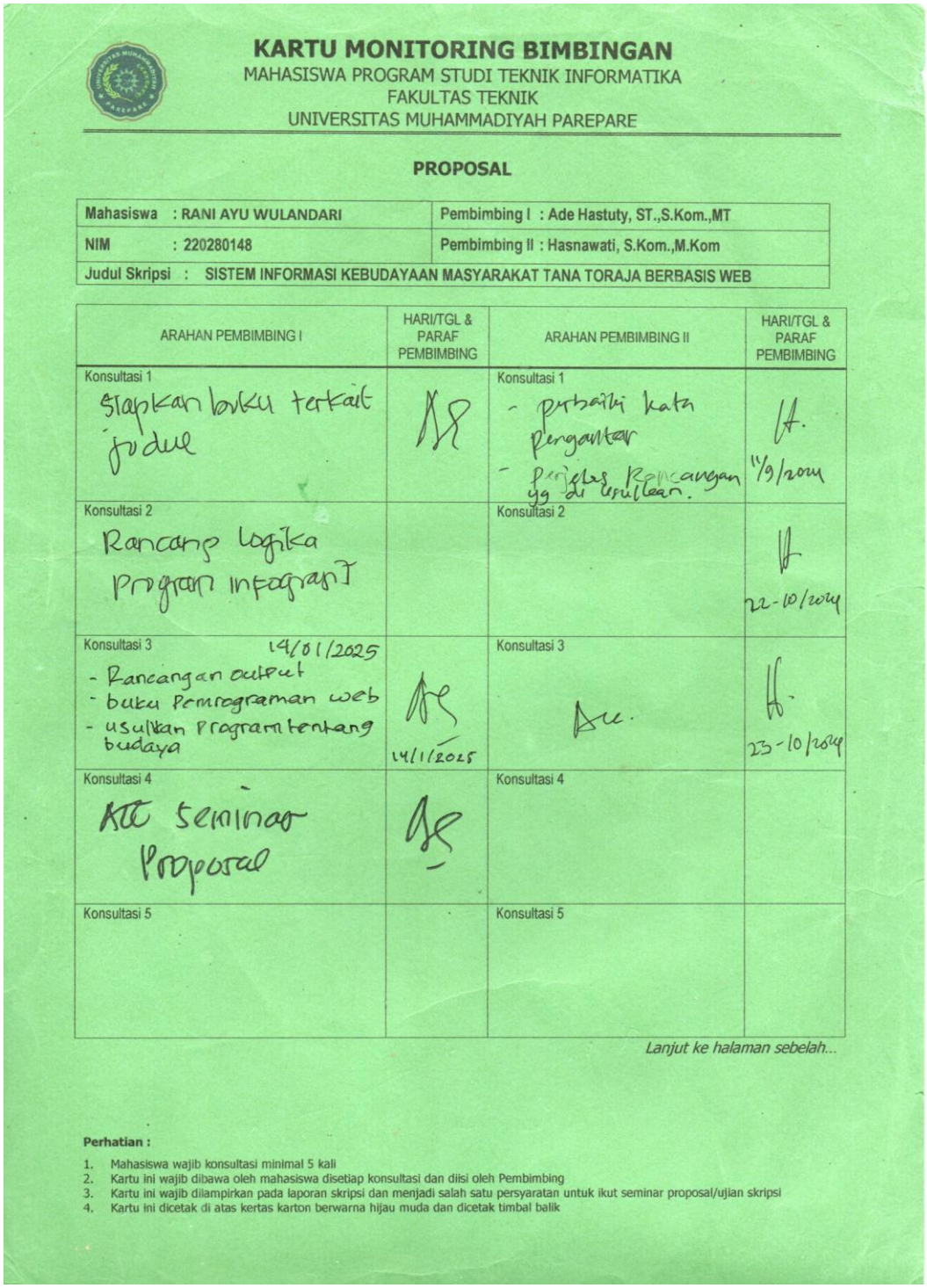
LAMPIRAN

1. Kartu Monitoring Bimbingan Proposal



KARTU MONITORING BIMBINGAN
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : RANI AYU WULANDARI	Pembimbing I : Ade Hastuty, ST.,S.Kom.,MT
NIM : 220280148	Pembimbing II : Hasnawati, S.Kom.,M.Kom
Judul Skripsi : SISTEM INFORMASI KEBUDAYAAN MASYARAKAT TANA TORAJA BERBASIS WEB	

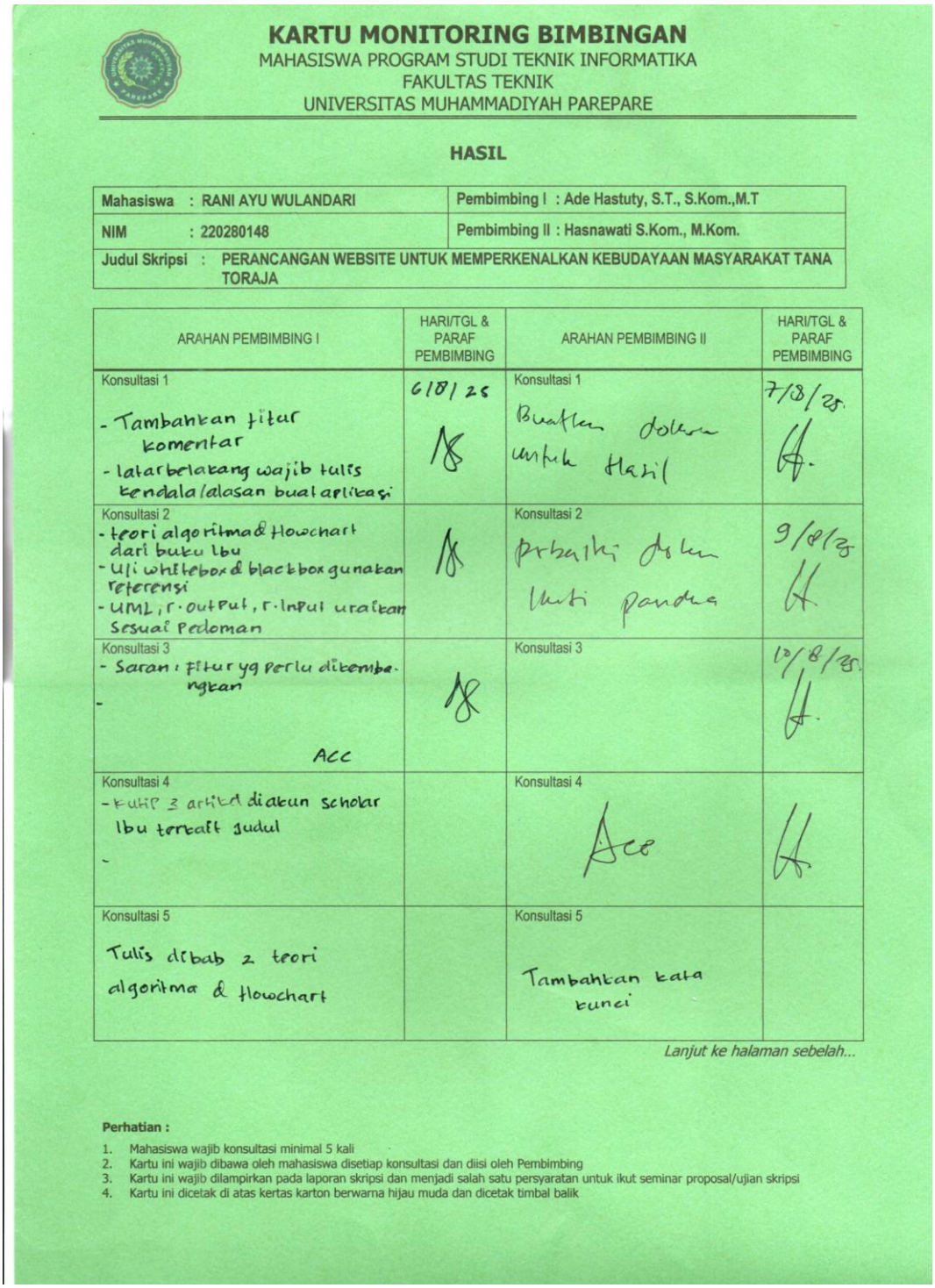
ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 siapkan buku terkait judul		Konsultasi 1 - perbaiki kata pengantar - perjelas Rancangan yg di usulkan.	11/9/2024
Konsultasi 2 Rancang logika program infograf		Konsultasi 2	22-10-2024
Konsultasi 3 14/01/2025 - Rancangan output - buku pemrograman web - usulkan program tentang budaya	14/1/2025	Konsultasi 3	23-10-2024
Konsultasi 4 ACC seminar Proposal		Konsultasi 4	
Konsultasi 5		Konsultasi 5	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

2. Kartu Monitoring Bimbingan Skripsi



KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

HASIL

Mahasiswa : RANI AYU WULANDARI	**Pembimbing I : Ade Hastuty, S.T., S.Kom.,M.T**
NIM : 220280148	**Pembimbing II : Hasnawati S.Kom., M.Kom.**
Judul Skripsi : PERANCANGAN WEBSITE UNTUK MEMPERKENALKAN KEBUDAYAAN MASYARAKAT TANA TORAJA	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 - Tambahkan fitur komentar - latarbelakang wajib tulis kendala/alasan buat aplikasi	6/8/25	Konsultasi 1 Buatkan dalam untuk Hasil	7/8/25
Konsultasi 2 - teori algoritma & flowchart dari buku Ibu - Uji whitebox & blackbox gunakan referensi - UML, r.output, r.input uraikan sesuai Pedoman		Konsultasi 2 perbaiki dalam Ibu pandu	9/8/25
Konsultasi 3 - Saran : fitur yg perlu dikembangkan - ACC		Konsultasi 3	12/8/25
Konsultasi 4 - kutip 3 artikel diakun scholar Ibu terkait judul -		Konsultasi 4 Acc	
Konsultasi 5 Tulis dibab 2 teori algoritma & flowchart		Konsultasi 5 Tambahkan kata kunci	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

1. Home User

Home.js

```
import React, { useEffect, useState } from "react";
import HeroWithVideo from "../MainComponent/homeComponent/hero";
import HomeSlider from "../MainComponent/homeComponent/homeSlider";
import MapSection from "../MainComponent/homeComponent/map";
import HomeTarian from "../MainComponent/homeComponent/homeTarian";
import HomeMakanan from "../MainComponent/homeComponent/homeMakanan";
import Footer from "../MainComponent/footer";
import HomeWisata from "../MainComponent/homeComponent/homeWisata";
import API from "../config/api";
import filosofi from "../assets/filosofi.jpg";
import { url } from "../config/route";
import LoaderPage from "./loader";
// pastikan file loader sudah ada

const HomePage = () => {
  const [dataSejarah, setDataSejarah] = useState([]);
  const [dataDance, setDataDance] = useState([]);
  const [dataFood, setDataFood] = useState([]);
  const [dataTour, setDataTour] = useState([]);
  const [error, setError] = useState("");
  const [isLoading, setIsLoading] = useState(true);

  const fetchAllData = async () => {
    setIsLoading(true);
    try {
      const resHistory = await API.get("/history");
      const resPhilosophy = await API.get("/philosophy");
      const resDance = await API.get("/dance/part");
      const resFood = await API.get("/food/part");
      const resTour = await API.get("/tourist/part");

      // Default image URL
      const defaultImage =
        "https://512pixels.net/wp-content/uploads/2025/06/11-0-Night-thumbnail.jpg";

      // Transform history data
      let transformedHistory = resHistory.data.map((item) => ({
        id: item.id,
        title: item.title,
        desc: item.description,
        img: item.images?.[0]?.image || defaultImage,
      }));

      // Add philosophy data
      if (resPhilosophy.data.length > 0) {
        transformedHistory.push({
          id: resPhilosophy.data[0].id,
          title: resPhilosophy.data[0].title,
          desc: resPhilosophy.data[0].description,
          img: filosofi || defaultImage,
        });
      }
```

```
      // Transform dance data
      const transformedDance =
        resDance.data?.map((item) =>
({
          ...item,
          images: [
            {
              image:
item.images?.[0]?.image ||
defaultImage,
            },
          ],
        })) || [];

      // Transform food data
      const transformedFood =
        resFood.data?.map((item) => ({
          ...item,
          images: [
            {
              image:
item.images?.[0]?.image ||
defaultImage,
            },
          ],
        })) || [];

      // Transform tour data
      const transformedTour =
        resTour.data?.map((item) => ({
          ...item,
          images: [
            {
              image:
item.images?.[0]?.image ||
defaultImage,
            },
          ],
        })) || [];

      console.log(transformedDance);
      setDataSejarah(transformedHisto
ry);
      setDataDance(transformedDance
);
      setDataFood(transformedFood);
      setDataTour(transformedTour);
      setError(""); // clear error
    } catch (err) {
      console.error("Error loading
data:", err);
      setError("Gagal memuat data.");
    } finally {
      setIsLoading(false);
    }
  };

  useEffect(() => {
    fetchAllData();
  }, []);

  if (isLoading) {
    return (
      <div className="flex justify-
center items-center h-screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory">
      {/* Hero Section */}
      <div className="snap-start h-
screen">
        <HeroWithVideo />
      </div>
      <div className="snap-start h-
screen">
        <MapSection />
      </div>
      <div className="snap-start h-
screen">
        <HomeSlider
slides={dataSejarah} />
      </div>
      <div className="snap-start h-
screen">
                <HomeTarian
          data={dataDance} />
```

```
        </div>
        <div className="snap-
    start h-screen">
          <HomeMakanan
    data={dataFood} />
        </div>
        <div className="snap-
    start h-screen">
          <HomeWisata
    data={dataTour} />
```

2. Sejarah & Asal Usul

Sejarah.js

```
import React, { useEffect,
useState } from "react";

import Footer from
"../MainComponent/footer";
import HeaderList from
"../MainComponent/listCompone
nt/headerList";
import ListSejarah from
"../MainComponent/sejarahComp
onent/listSejarah";
import ListFilosofi from
"../MainComponent/sejarahComp
onent/listFilosofi";
import SejarahSection from
"../MainComponent/sejarahComp
onent/sejarahSection";
import AsalSection from
"../MainComponent/sejarahComp
onent/asalSection";
import API from "../config/api";
import { url } from
"../config/route";
import LoaderPage from
"./loader";

const SejarahPage = () => {
  const [dataSejarah,
setDataSejarah] = useState({
    id: null,
    title: "",
    desc: "",
    img: [],
  });
  const [dataFilosofi,
setDataFilosofi] = useState([]);
  const [dataAsal, setDataAsal] =
useState({
    id: null,
    title: "",
    desc: "",
    img: [],
  });
  const [error, setError] =
useState("");
  const [isLoading, setIsLoading]
= useState(true);

  const getData = async () => {
    try {
      // ambil data sejarah & asal
      const resSejarah = await
API.get("/history");
      let transformedData =
resSejarah.data.map((item) => ({
        id: item.id,
        title: item.title,
        desc: item.description,
        img: item.images &&
item.images.length > 0 ?
item.images : [],
      }));
      const sejarah =
transformedData.find((a) =>
        a.title.toLowerCase().includ
es("sejarah")
```

```
        </div>
        <div className="snap-
    start h-screen">
          <Footer />
        </div>
      </div>
    );
  };

  export default HomePage;
```

```
      ) || { id: null, title: "", desc:
"", img: [] };
      const asal =
transformedData.find(
        (a) =>
!a.title.toLowerCase().includes("
sejarah")
      ) || { id: null, title: "", desc:
"", img: [] };

      setDataSejarah(sejarah);
      setDataAsal(asal);

      // ambil data filosofi
      const resFilosofi = await
API.get("/philosophy");
      setDataFilosofi(resFilosofi.da
ta || []);

      setError("");
    } catch (err) {
      console.error("Error loading
data:", err);
      setError("Gagal memuat
data");
    } finally {
      setIsLoading(false);
    }
  };

  useEffect(() => {
    getData();
  }, []);

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory">
      {/* Hero Section */}
      <HeaderList
        title="Sejarah dan Asal
Usul"
        image="https://i.pinimg.com
/736x/3c/22/3f/3c223fb465fcd2f
831cb1b27ebc8cdf7.jpg"
      />

      <ListSejarah />
      {isLoading ? (
        <>
          <LoaderPage />
        </>
      ) : (
        <>
          <SejarahSection
data={dataSejarah} />
          <AsalSection
data={dataAsal} />
          <ListFilosofi
data={dataFilosofi} />
        </>
      )}

      <Footer />
    </div>
  );
};

export default SejarahPage;
```

3. Tarian

Tarian.js

```
import React, { useEffect,
useState } from "react";
import Footer from
"../MainComponent/footer";
import HeaderList from
"../MainComponent/listCompone
nt/headerList";
import ListCard from
"../MainComponent/listCompone
nt/listCard";
import API from "../config/api";
import Swal from "sweetalert2";
import LoaderPage from
"./loader";
```

```
const desc =
  "Tarian adat Toraja tidak hanya sekadar pertunjukan seni, tetapi juga medium penting untuk mengekspresikan nilai-nilai budaya, kepercayaan spiritual, dan struktur sosial masyarakat Toraja. Sebagian besar tarian ini terkait erat dengan siklus kehidupan, terutama upacara kematian Rambu Solo' dan upacara syukuran Rambu Tuka'. Gerakan-gerakan dalam tarian seringkali simbolis, mencerminkan perjalanan arwah, kegembiraan, atau kesedihan.";

const TarianPage = () => {
  const [data, setData] = useState([]); // default: []
  const [categories, setCategories] = useState([]); // default: []
  const [error, setError] = useState("");
  const [isLoading, setIsLoading] = useState(true);
  const defaultTariData = [
    {
      id: 1,
      title: "Tari Pa'gellu",
      description:
        "Tarian sukacita yang biasa dipertunjukkan pada upacara adat.",
      image:
        "https://i.pinimg.com/1200x/11/ce/30/11ce30ac14196a6aa5c4312016583cca.jpg",
    },
  ];
  const defaultTariCategories = [
    { value: 1, label: "Upacara" },
    { value: 2, label: "Penyambutan" },
  ];

  const fetchAllData = async () => {
    setIsLoading(true);
    try {
      const [resDance, resCategories] = await Promise.all([
        API.get("/dance"),
        API.get("/dance/categories"),
      ]);
      const transformedCategories = resCategories.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setData(resDance.data || []);
      setCategories(transformedCategories || []);
    } catch (err) {
      console.error(err);
      const result = await Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat data tarian atau kategori.",
        showCancelButton: true,
        confirmButtonText: "Coba Lagi",
        cancelButtonText: "Tampilkan Default",
      });
      if (result.isConfirmed) {
        fetchAllData();
        return;
      } else {
        setData(defaultTariData);
        setCategories(defaultTariCategories);
      }
    } finally {
      setIsLoading(false);
    }
```

```
  };

  useEffect(() => {
    fetchAllData();
  }, []);

  if (isLoading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory">
      {/* Hero Section */}
      <HeaderList
        title="Tarian Khas Toraja"
        image="https://i.pinimg.com
/1200x/9d/5c/26/9d5c26608b136
4bbb47ae4761f6f3a55.jpg"
      />

      <ListCard
        topic="dance"
        data={data}
        categories={categories}
        title="Tarian Tradisional
Khas Toraja"
        description={desc}
        image="https://i.pinimg.com
/1200x/11/ce/30/11ce30ac14196a
6aa5c4312016583cca.jpg"
      />

      <Footer />
    </div>
  );
};

export default TarianPage;
```

4. Makanan

Makanan.js

```
import React, { useEffect,
useState } from "react";
import Footer from
"../MainComponent/footer";
import HeaderList from
"../MainComponent/listCompone
nt/headerList";
import ListCard from
"../MainComponent/listCompone
nt/listCard";
import API from "../config/api";
import Swal from "sweetalert2";
import LoaderPage from
"./loader"; // pastikan ada loader

const desc =
  "Kuliner Toraja memiliki cita
rasa yang kuat dan unik,
didominasi oleh penggunaan
bumbu rempah alami serta cara
memasak tradisional. Banyak
hidangan dimasak dalam bambu
atau daun pisang, memberikan
aroma khas dan mempertahankan
kekayaan rasa. Makanan Toraja
tidak hanya sekadar santapan,
tetapi juga bagian integral dari
ritual adat dan kehidupan sehari-
hari.";

const MakananPage = () => {
  const [data, setData] =
useState([]);
  const [categories, setCategories]
= useState([]);
  const [error, setError] =
useState("");
```

```
  const [isLoading, setIsLoading]
= useState(true);
  const defaultFoodData = [
    {
      id: 1,
      title: "Pa'piong",
      description: "Makanan khas
Toraja yang dimasak dalam
bambu.",
      image:
        "https://i.pinimg.com/1200x
/fc/0b/dd/fc0bdd8b1b86fff3cc80
80ec650985f1.jpg",
    },
  ];
  const defaultFoodCategories = [
    { value: 1, label: "Daging" },
    { value: 2, label: "Sayuran" },
  ];

  const fetchAllData = async ()
=> {
    setIsLoading(true);
    try {
      const [resFoods,
resCategories] = await
Promise.all([
        API.get("/food"),
        API.get("/food/categories"),
      ]);
      const transformedCategories
= resCategories.data.map((item)
=> ({
        value: item.id,
        label: item.name_category,
      }));
      setData(resFoods.data || []);
      setCategories(transformedCat
egories || []);
    } catch (err) {
      console.error(err);
      const result = await
Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat data
makanan atau kategori.",
        showCancelButton: true,
        confirmButtonText: "Coba
Lagi",
        cancelButtonText:
"Tampilkan Default",
      });
      if (result.isConfirmed) {
        fetchAllData();
        return;
      } else {
        setData(defaultFoodData);
        setCategories(defaultFoodC
ategories);
      }
    } finally {
      setIsLoading(false);
    }
  };

  useEffect(() => {
    fetchAllData();
  }, []);

  if (isLoading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory">
      {/* Hero Section */}
      <HeaderList
        title="Makanan Khas
Toraja"
        image="https://i.pinimg.com
/1200x/1d/3f/60/1d3f60d5b1ae11
bfc8aa92f769bfe95a.jpg"
```

```
      />
      <ListCard
        topic="food"
        data={data}
        categories={categories}
        title="Makanan Tradisional
Khas Toraja"
        description={desc}
        image="https://i.pinimg.com
/1200x/fc/0b/dd/fc0bdd8b1b86fff
3cc8080ec650985f1.jpg"
      />
      <Footer />
    </div>
  );
};

export default MakananPage;
```

5. Wisata

Wisata.js

```
import React, { useEffect,
useState } from "react";
import Footer from
"../MainComponent/footer";
import HeaderList from
"../MainComponent/listCompone
nt/headerList";
import ListCard from
"../MainComponent/listCompone
nt/listCard";
import Swal from "sweetalert2";
import API from "../config/api";
import LoaderPage from
"./loader";

const desc =
  "Toraja menawarkan
pengalaman wisata yang kaya,
memadukan keindahan alam
pegunungan yang memukau
dengan kekayaan budaya yang
otentik. Anda bisa menjelajahi
situs megalitikum kuno,
menyaksikan upacara adat yang
penuh makna, atau menikmati
panorama lanskap persawahan
terasering yang hijau. Setiap
sudut Toraja menjanjikan cerita
dan keunikan yang tak
terlupakan.";

// ✅ Default data untuk backup
const defaultData = [
  {
    id: 1,
    title: "Kete Kesu'",
    description: "Desa tradisional
dengan rumah Tongkonan yang
indah dan kuburan tebing.",
    image:
"https://i.pinimg.com/1200x/74/7
b/d9/747bd9458f1151a8c076128
ad30ebbe0.jpg",
  },
  {
    id: 2,
    title: "Londa",
    description: "Kuburan di
tebing batu dengan peti mati
kayu dan patung tau-tau.",
    image:
"https://i.pinimg.com/1200x/cf/2
c/c0/cf2cc00bef38e3e3eaca1682
75fd8887.jpg",
  },
];

const defaultCategories = [
  { value: 1, label: "Budaya" },
  { value: 2, label: "Alam" },
];

const WisataPage = () => {
  const [data, setData] =
useState([]);
```

```
  const [categories, setCategories]
= useState([]);
  const [isLoading, setIsLoading]
= useState(true);

  const fetchAllData = async ()
=> {
    setIsLoading(true);
    try {
      const [resWisata,
resCategories] = await
Promise.all([
        API.get("/tourist"),
        API.get("/tourist/categories"
),
      ]);

      const transformedCategories
= resCategories.data.map((item)
=> ({
        value: item.id,
        label: item.name_category,
      }));

      setData(resWisata.data || []);
      setCategories(transformedCat
egories || []);
    } catch (err) {
      console.error("Error loading
data:", err);
      // ✅ Show swal with retry
option
      const result = await
Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat data
wisata atau kategori.",
        showCancelButton: true,
        confirmButtonText: "Coba
Lagi",
        cancelButtonText:
"Tampilkan Default",
      });

      if (result.isConfirmed) {
        // Retry
        fetchAllData();
        return;
      } else {
        // Load default data
        setData(defaultData);
        setCategories(defaultCatego
ries);
      }
    } finally {
      setIsLoading(false);
    }
  };

  useEffect(() => {
    fetchAllData();
  }, []);

  if (isLoading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory">
      <HeaderList
        title="Wisata Daerah
Toraja"
        image="https://i.pinimg.com
/1200x/cf/2c/c0/cf2cc00bef38e3e
3eaca168275fd8887.jpg"
      />
      <ListCard
        topic="tourist"
        data={data}
        categories={categories}
        title="Wisata Daerah
Masyarakat Toraja"
        description={desc}
```

```
      image="https://i.pinimg.com
/1200x/74/7b/d9/747bd9458f115
1a8c076128ad30ebbe0.jpg"
    />
    <Footer />
```

6. FAQ

Faq.js

```
// src/components/FAQList.jsx
import React, { useEffect,
useState } from "react";
import { FaChevronDown,
FaChevronUp } from "react-
icons/fa";
import HeaderList from
"../MainComponent/listCompone
nt/headerList";
import { motion,
AnimatePresence } from
"framer-motion";
import Swal from "sweetalert2";
import API from "../config/api";

const FAQList = () => {
  const [openId, setOpenId] =
useState(null);

  const [showForm,
setShowForm] = useState(false);
  const [email, setEmail] =
useState("");
  const [question, setQuestion] =
useState("");
  const [isSubmitting,
setIsSubmitting] =
useState(false);
  const [questions, setQuestions]
= useState([]);
  const [loading, setLoading] =
useState(true);
  const [error, setError] =
useState("");
  const toggleAnswer = (id) => {
    setOpenId(openId === id ?
null : id);
  };

  const handleSubmit = async (e)
=> {
    e.preventDefault();

    try {
      setIsSubmitting(true);
      const response = await
API.post("/question", {
        email,
        question,
      });

      Swal.fire({
        icon: "success",
        title: "Pertanyaan berhasil
dikirim!",
        html: `
          <p class="mt-2">Jawaban
akan dikirim ke email
<strong>${response.data.email}
</strong></p>
        `,
      });

      setEmail("");
      setQuestion("");
      setIsSubmitting(false);
      setShowForm(false);
    } catch (err) {
      console.error("[FRONTEND
ERROR]", err);
      setIsSubmitting(false);

      Swal.fire({
        icon: "error",
    </div>
  );
};

export default WisataPage;
```

```
        title: "Gagal mengirim
pertanyaan",
        text:
          err.response?.data?.messag
e ||
          "Terjadi kesalahan. Silakan
coba lagi nanti.",
      });
    }
  };

  const fetchQuestions = async ()
=> {
    setLoading(true);
    try {
      const res = await
API.get("/question", {
        params: {
          status: undefined,
          page: 1,
          limit: 100,
        },
      });
      console.log(res.data);
      setQuestions(res.data.data);
      setError("");
    } catch (err) {
      console.error("[GET
QUESTIONS ERROR]", err);
      setError("Gagal memuat data
pertanyaan.");
    } finally {
      setLoading(false);
    }
  };

  useEffect(() => {
    fetchQuestions();
  }, []);

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory">
      <HeaderList
        title="Pertanyaan dan
Jawaban"
        image="https://i.pinimg.com
/1200x/1d/3f/60/1d3f60d5b1ae11
bfc8aa92f769bfe95a.jpg"
      />
      <motion.div
        initial={{ opacity: 0, y: -30
}}
        animate={{ opacity: 1, y: 0
}}
        transition={{ duration: 0.8,
ease: "easeOut" }}
      >
        <AnimatePresence>
          {showForm && (
            <motion.div
              initial={{ opacity: 0, y: -
20 }}
              animate={{ opacity: 1,
y: 0 }}
              exit={{ opacity: 0, y: -
20 }}
              transition={{ duration:
0.4 }}
              className="max-w-5xl
mx-auto mt-10 bg-gray-100
border border-toraja-merah
rounded-xl shadow-md overflow-
hidden p-6 mb-8"
            >
              <div className="text-
center mb-6">
                <h3 className="text-
2xl font-semibold text-toraja-
merah mb-2">
                  Punya Pertanyaan?
                </h3>
                <p className="text-
toraja-merah text-sm">
                  Kirim pertanyaan
Anda tentang website kami dan
kami akan
                  segera membalas
                </p>
              </div>
```

```
            <form
onSubmit={handleSubmit}
className="space-y-4">
              <div
className="space-y-1">
                <input
                  type="email"
                  id="email"
                  value={email}
                  onChange={(e) =>
setEmail(e.target.value)}
                  placeholder="email
@contoh.com"
                  className="w-full
px-4 py-2 border border-gray-
300 bg-white rounded-lg
focus:ring-2 focus:ring-blue-500
focus:border-blue-500 transition-
all"
                  required
                />
              </div>

              <div
className="space-y-1">
                <textarea
                  id="question"
                  value={question}
                  onChange={(e) =>
setQuestion(e.target.value)}
                  placeholder="Tulis
pertanyaan Anda di sini..."
                  rows="4"
                  className="w-full
px-4 py-2 border text-gray-900
border-gray-300 rounded-lg
focus:ring-2 focus:ring-blue-500
focus:border-blue-500 transition-
all resize-y"
                  required
                />
              </div>

              <button
                type="submit"
                disabled={isSubmittin
g}
                className={`w-full
py-2 px-4 rounded-lg font-
medium text-white transition-all
${
                  isSubmitting
                    ? "bg-gray-400
cursor-not-allowed"
                    : "bg-toraja-merah
text-white hover:bg-transparent
hover:bg-white hover:border-
toraja-merah hover:text-toraja-
merah border border-white
active:scale-95"
                }`}
              >
                {isSubmitting ?
"Mengirim..." : "Kirim
Pertanyaan"}
              </button>
            </form>
          </motion.div>
        )}
      </AnimatePresence>
      <div className="max-w-
5xl mx-auto mt-10 bg-gray-100
p-6 rounded-2xl shadow-lg">
        <div className="mb-6
flex items-center justify-
between">
          <h1 className="text-3xl
font-bold text-toraja-
merah">FAQ</h1>
          <button
            onClick={() =>
setShowForm(!showForm)}
            className="bg-toraja-
merah text-white px-4 py-2
rounded-full hover:bg-toraja-
merah transition"
          >
            Punya Pertanyaan?
          </button>
        </div>
```

```
          {questions.map((q) => (
            <div
              key={q.id}
              className={`mb-4 p-4
rounded-xl shadow-sm
transition-all duration-300 ${
                openId === q.id
                  ? "bg-white border
border-toraja-merah"
                  : "bg-white"
              }`}
            >
              <div
                className="flex
justify-between items-center
cursor-pointer"
                onClick={() =>
toggleAnswer(q.id)}
              >
                <h2
                  className={`font-
semibold transition-all ${
                    openId === q.id ?
"text-toraja-merah" : "text-gray-
800"
                  }`}
                >
                  {q.question}
                </h2>
                {openId === q.id ? (
                  <FaChevronUp
className="text-toraja-merah"
/>
                ) : (
                  <FaChevronDown
className="text-toraja-merah"
/>
                )}
              </div>

              {/* Animated Answer
*/}
              <AnimatePresence>
                {openId === q.id && (
                  <motion.div
                    key="answer"
                    initial={{ opacity: 0,
height: 0 }}
                    animate={{ opacity:
1, height: "auto" }}
                    exit={{ opacity: 0,
height: 0 }}
                    transition={{
duration: 0.3 }}
                    className="overflo
w-hidden mt-3 text-sm text-gray-
600"
                  >
                    {q.answer?q.answer:
"Belum Terjawab"}
                  </motion.div>
                )}
              </AnimatePresence>
            </div>
          ))}
        </div>
      </motion.div>
    </div>
  );
};

export default FAQList;
```

7. Detail Semua Konten

DetailPage.js

```
import React, { useEffect,
useState } from "react";
import { useNavigate, useParams
} from "react-router-dom";
import DetailHeader from
"../MainComponent/detailCompo
nent/detailHeader";
```

```
import DetailContent from
"../MainComponent/detailCompo
nent/detailContent";
// import DetailGallery from
"../MainComponent/detailCompo
nent/detailGallery";
import Footer from
"../MainComponent/footer";
import API from "../config/api";
import LoaderPage from
"./loader"; // pastikan file loader
ada
import SliderImage from
"../MainComponent/listCompone
nt/sliderImages";

const DetailPage = () => {
  const { id, topic } =
useParams();
  const [data, setData] =
useState(null); // null default
lebih aman
  const [allData, setAllData] =
useState(null); // null default
lebih aman
  const [error, setError] =
useState("");
  const [title, setTitle] =
useState("");
  const [isLoading, setIsLoading]
= useState(true);

  const loadData = async () => {
    setIsLoading(true);
    try {
      const res = await
API.get(`/${topic}/${id}`);
      console.log(res.data,
"detail");
      setData(res.data);
      // await getData();
      setTitle(res.data.title);
      setError("");
    } catch (err) {
      console.error(err);
      setError("Gagal memuat
data.");
    } finally {
      setIsLoading(false);
    }
  };

  useEffect(() => {
    loadData();
  }, [id, topic]); // tambahkan
dependency jika ganti id/topic

  const navigate = useNavigate();

  if (isLoading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  // opsional: handle jika data null
atau error
  if (!data) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <p className="text-red-
500">{error || "Data tidak
tersedia."}</p>
      </div>
    );
  }

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory">
      <DetailHeader data={data}
/>
      <DetailContent data={data}
/>
```

```
      <SliderImage
data={data.images} title={title}
/>
      <Footer />
    </div>
  );
};

export default DetailPage;
```

8. Login Admin

Auth.js

```
import React, { useState } from
"react";
import API from "../config/api";
import { motion } from "framer-
motion";
import { useNavigate } from
"react-router-dom";

function Login() {
  const [step, setStep] =
useState("login"); // 'login' |
'requestToken' | 'reset'
  const [username, setUsername]
= useState("");
  const [password, setPassword]
= useState("");
  const [resetToken,
setResetToken] = useState("");
  const [newPassword,
setNewPassword] = useState("");
  const navigate = useNavigate();

  const handleLogin = async (e)
=> {
    e.preventDefault();
    try {
      const res = await
API.post("/auth/login", {
username, password });
      localStorage.setItem("token",
res.data.token);
      alert("Login berhasil");
      navigate(`/admin`);
      window.location.reload();

      // TODO: navigate ke
dashboard
    } catch (err) {
      alert("Login failed");
    }
  };

  const getToken = async () => {
    try {
      const res = await
API.post("/auth/forgot-
password", { username });
      setStep("reset");
      alert(
        "Reset token dikirim ke
Telegram. Cek Telegram &
masukkan di bawah."
      );
    } catch (err) {
      alert("Gagal mengirim reset
token");
    }
  };

  const handleReset = async () =>
{
    try {
      await API.post("/auth/reset-
password", {
        token: resetToken,
        newPassword,
      });
      alert("Password berhasil
direset. Silakan login lagi.");
      setStep("login");
    } catch (err) {
      alert("Reset password
gagal");
    }
```

```
  };

  return (
    <div
      className="relative flex items-center justify-center min-h-screen bg-cover bg-center"
      style={{
        backgroundImage: `url(https://res.cloudinary.com/dipdl14x/image/upload/v1751188952/pxoqhdsvzooxu7cngspx.jpg)`,
      }}
    >
      <div className="absolute inset-0 bg-gradient-to-r from-black/100 to-transparent z-10"></div>

      <motion.div
        initial={{ opacity: 0, y: -40 }}
        animate={{ opacity: 1, y: 0 }}
        transition={{ duration: 0.6, ease: "easeOut" }}
        className="relative z-20 w-full max-w-md p-8 space-y-8 bg-white/10 backdrop-blur-md rounded-lg border border-white/20 shadow-lg"
      >
        <h2 className="text-3xl font-bold text-center text-[#F5F5DC]">
          BUDAYA TORAJA
        </h2>

        {step === "login" && (
          <>
            <p className="text-center text-[#F5F5DC]/80">
              Isi Username & Password untuk Masuk
            </p>

            <div className="space-y-4">
              <input

placeholder="Username"
                onChange={(e) => setUsername(e.target.value)}
                className="w-full px-3 py-2 text-[#F5F5DC] placeholder-white/70 bg-white/10 border border-[#F5F5DC]/30 rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500 sm:text-sm backdrop-blur-sm"
              />
              <input
                type="password"

placeholder="Password"
                onChange={(e) => setPassword(e.target.value)}
                className="w-full px-3 py-2 text-[#F5F5DC] placeholder-white/70 bg-white/10 border border-[#F5F5DC]/30 rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500 sm:text-sm backdrop-blur-sm"
              />
              <button

onClick={handleLogin}
                className="inline-block w-full cursor-pointer rounded-xl border-[1.58px] border-zinc-600 bg-[#8B0000] px-5 py-3 font-medium text-[#F5F5DC] shadow-md transition-all duration-300 hover:translate-y-[-0.335rem] hover:shadow-xl"
              >
                Login
              </button>
```

```
            <button
              onClick={() =>
setStep("requestToken")}
              className="text-sm
text-[#FFD700] hover:underline"
            >
              Lupa password?
            </button>
          </div>
        </>
      )}

      {step === "requestToken"
&& (
        <>
          <p className="text-
center text-[#F5F5DC]/80">
            Masukkan Username
untuk Kirim Reset Token
          </p>
          <div className="space-
y-4">
            <input

placeholder="Username"
              onChange={(e) =>
setUsername(e.target.value)}
              className="w-full px-
3 py-2 text-[#F5F5DC]
placeholder-white/70 bg-
white/10 border border-
[#F5F5DC]/30 rounded-md
focus:outline-none focus:ring-2
focus:ring-blue-500 sm:text-sm
backdrop-blur-sm"
            />
            <button
              onClick={getToken}
              className="inline-
block w-full cursor-pointer
rounded-xl border-[1.58px]
border-zinc-600 bg-[#8B0000]
px-5 py-3 font-medium text-
[#F5F5DC] shadow-md
transition-all duration-300
hover:translate-y-[-0.335rem]
hover:shadow-xl"
            >
              Kirim Reset Token
            </button>
            <button
              onClick={() =>
setStep("login")}
              className="text-sm
text-[#FFD700] hover:underline"
            >
              Kembali ke Login
            </button>
          </div>
        </>
      )}

      {step === "reset" && (
        <>
          <p className="text-
center text-[#F5F5DC]/80">
            Masukkan Token &
Password Baru
          </p>
          <div className="space-
y-4">
            <input
              placeholder="Reset
Token"
              onChange={(e) =>
setResetToken(e.target.value)}
              className="w-full px-
3 py-2 text-[#F5F5DC]
placeholder-white/70 bg-
white/10 border border-
[#F5F5DC]/30 rounded-md
focus:outline-none focus:ring-2
focus:ring-blue-500 sm:text-sm
backdrop-blur-sm"
            />
            <input
              type="password"
              placeholder="Password
Baru"
              onChange={(e) =>
setNewPassword(e.target.value)}
```

```
            className="w-full px-
3 py-2 text-[#F5F5DC]
placeholder-white/70 bg-
white/10 border border-
[#F5F5DC]/30 rounded-md
focus:outline-none focus:ring-2
focus:ring-blue-500 sm:text-sm
backdrop-blur-sm"
          />
          <button

onClick={handleReset}
            className="inline-
block w-full cursor-pointer
rounded-xl border-[1.58px]
border-zinc-600 bg-[#8B0000]
px-5 py-3 font-medium text-
[#F5F5DC] shadow-md
transition-all duration-300
hover:translate-y-[-0.335rem]
hover:shadow-xl"
          >
            Reset Password
          </button>
          <button
            onClick={() =>
setStep("login")}
            className="text-sm
text-[#FFD700] hover:underline"
          >
            Kembali ke Login
          </button>
        </div>
      </>
    )}
  </motion.div>
 </div>
 );
}

export default Login;
```

9. Admin Home Page

Homepage.js

```
import { useEffect, useState }
from "react";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import ListCardAdmin from
"../../../MainComponent/adminCo
mponent/listCardAdmin";
import Swal from "sweetalert2";
import DashboardComponent
from
"../../../MainComponent/adminCo
mponent/dashboard";


export default function
HomeAdmin() {
 const [data, setData] =
useState([]);

 return (
  <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-8 bg-white">
    {/* <HeaderAdmin
title={"Kelola Data Tarian
Tradisional"} /> */}
    <DashboardComponent />
  </div>
 );
}
```

10. Sejarah

HistoryPage.js

```
import { useEffect, useState }
from "react";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import ListCardAdmin from
"../../../MainComponent/adminCo
mponent/listCardAdmin";
import LoaderPage from
"../../loader"; // pastikan file ini
ada

export default function History()
{
  const [data, setData] =
useState([]);
  const [newTitle, setNewTitle] =
useState("");
  const [newDesc, setNewDesc] =
useState("");
  const [error, setError] =
useState("");
  const [loading, setLoading] =
useState(false); // tambahkan
loading

  const getData = async () => {
    try {
      setLoading(true); // mulai
loading
      const res = await
API.get("/history");
      setData(res.data);
      setError("");
    } catch (err) {
      console.error("Error loading
data:", err);
      setError("Gagal memuat
data");
    } finally {
      setLoading(false); // selesai
loading
    }
  };

  useEffect(() => {
    getData();
  }, []);

  const handleCreate = async ()
=> {
    try {
      setLoading(true); // loading
saat create
      await API.post("/history", {
        title: newTitle,
        description: newDesc,
        image: "", // atau path
default
      });
      setNewTitle("");
      setNewDesc("");
      getData();
      setError("");
    } catch (err) {
      console.error("Error creating
data:", err);
      setError("Gagal menambah
data");
    } finally {
      setLoading(false);
    }
  };

  const handleDelete = async (id)
=> {
    try {
      setLoading(true); // loading
saat delete
      await
API.delete(`/history/${id}`);
      getData();
      setError("");
    } catch (err) {
      console.error("Error deleting
data:", err);
      setError("Gagal menghapus
data");
    } finally {
      setLoading(false);
```

```
  }
 };

 if (loading) {
  return (
   <div className="flex
justify-center items-center h-
screen">
    <LoaderPage />
   </div>
  );
 }

 return (
  <div className="relative h-
screen overflow-y-auto snap-y
```

DetailStory.js

```
import React, { useEffect,
useState } from "react";
import { useNavigate, useParams
} from "react-router-dom";
import API from
"../../../config/api";
import DetailAdmin from
"../../../MainComponent/adminCo
mponent/detailAdmin";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import Breadcrumb from
"../../../MainComponent/adminCo
mponent/breadcrumb";
import Swal from "sweetalert2";
import LoaderPage from
"../../loader";
import { storage } from
"../../../config/firebase";
import { ref, uploadBytes,
getDownloadURL } from
"firebase/storage";
const DetailHistory = () => {
 const { id } = useParams();
 const [data, setData] =
useState(null);
snap-mandatory font-montserrat
p-10 bg-white">
    <HeaderAdmin
title={"Kelola Data Sejarah"} />
    <ListCardAdmin
     topic={"sejarah"}
     data={data}
     image="https://res.cloudinar
y.com/diipdl14x/image/upload/v
1751567242/filosofi_dlh9dl.jpg"
     title="Sejarah dan Asal Usul
Budaya Toraja"
     description=""
    />
   </div>
  );
}

 const [loadingPage,
setLoadingPage] =
useState(true);
 const [loadingUpdate,
setLoadingUpdate] =
useState(false);
 const navigate = useNavigate();

 useEffect(() => {
  loadData();
 }, [id]);

 const loadData = async () => {
  try {
   setLoadingPage(true);a
   const res = await
API.get(`/history/${id}`);
   console.log(res.data, "data
detail");
   setData(res.data);
  } catch (err) {
   console.error(err);
   Swal.fire({
    icon: "error",
    title: "Gagal memuat data",
    text: "Periksa koneksi atau
coba lagi nanti.",
   });
```

```
    } finally {
      setLoadingPage(false);
    }
  };

  const updateText = async (title,
description, images) => {
    try {
      setLoadingUpdate(true);
      await
API.put(`/history/${id}`, { title,
description });
      console.log("Update title &
description sukses",images);

      const editedImages =
images.filter((img) => img.isEdit
&& img.id && img.file);
      for (const img of
editedImages) {
        await
updateSingleImage(img.id,
img.file);
      }

      const newImages =
images.filter((img) =>
img.isNew && img.file);
      for (const img of newImages)
{
        await addImage(img.file);
      }

      const deletedImages =
images.filter((img) =>
img.isDeleted && img.id);
      for (const img of
deletedImages) {
        await
deleteSingleImage(img.id);
      }

      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Data berhasil
diperbarui.",
      });
      navigate(`/admin/sejarah/`);

      loadData(); // refresh data
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Oops...",
        text: "Gagal memperbarui
data!",
      });
    } finally {
      setLoadingUpdate(false);
    }
  };

    const addImage = async (file)
=> {
    try {
      const storageRef =
ref(storage,
`historys/${Date.now()}-
${file.name}`);
      const snapshot = await
uploadBytes(storageRef, file);
      const downloadURL = await
getDownloadURL(snapshot.ref);

      await
API.post(`/history/${id}/image`,
{
        images: [downloadURL], //
karena backend expects array of
URLs
      });
    } catch (err) {
      console.error("Gagal
menambahkan gambar:", err);
    }
  };

  const updateSingleImage =
async (imageId, file) => {
```

```
    try {
      const storageRef =
ref(storage,
`historys/${Date.now()}-
${file.name}`);
      const snapshot = await
uploadBytes(storageRef, file);
      const downloadURL = await
getDownloadURL(snapshot.ref);

      await
API.put(`/history/${id}/image/${
imageId}`, {
        image: downloadURL, //
backend expects a single image
URL
      });
    } catch (err) {
      console.error("Gagal update
gambar:", err);
    }
  };

  const deleteSingleImage =
async (imageId) => {
    await
API.delete(`/history/${id}/image/
${imageId}`);
  };

  if (loadingPage||loadingUpdate)
{
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  if (!data) return <p>Data tidak
ditemukan</p>;

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-4 bg-white">
      <Breadcrumb
        items={[
          { label: "Dashboard", href:
"/admin" },
          { label: "Sejarah", href:
"/admin/sejarah" },
          { label: "Detail" },
        ]}
      />
      <HeaderAdmin
title={"Update Detail Data
Sejarah"} />
      <DetailAdmin
        data={data}
        updateText={updateText}
        categories={[]} // history
tidak ada kategori
        topic={"sejarah"}
        loadingUpdate={loadingUp
date}
      />
    </div>
  );
};

export default DetailHistory;
```

11. Makanan

FoodPage.js

```
import { useEffect, useState }
from "react";

import API from
"../../../config/api";
```

```
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import ListCardAdmin from
"../../../MainComponent/adminCo
mponent/listCardAdmin";
import Swal from "sweetalert2";
import LoaderPage from
"../../loader"; // pastikan
komponen loader ini ada

export default function
FoodPage() {
  const [data, setData] =
useState([]);
  const [error, setError] =
useState("");
  const [loading, setLoading] =
useState(false);

  const [categories, setCategories]
= useState([]);

  useEffect(() => {
    getCategories();
  }, []);

  const getCategories = async ()
=> {
    try {
      setLoading(true);
      const res = await
API.get(`/food/categories`);
      const transformedData =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformedDat
a);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat
kategori makanan.",
      });
    } finally {
      setLoading(false);
    }
  };

  const getData = async () => {
    try {
      setLoading(true); //
tambahkan loading saat fetch
data
      const res = await
API.get("/food");
      setData(res.data);
      setError("");
    } catch (err) {
      console.error("Error loading
data:", err);
      setError("Gagal memuat
data");
    } finally {
      setLoading(false);
    }
  };

  useEffect(() => {
    getData();
  }, []);

  if (loading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-10 bg-white">
```

```
    <HeaderAdmin
title={"Kelola Data Makanan
Tradisional"} />
    <ListCardAdmin
      topic={"makanan"}
      data={data}
      image="https://res.cloudinar
y.com/diipdl14x/image/upload/v
1751567242/filosofi_dlh9dl.jpg"
      title="Makanan Tradisional
Khas Budaya Toraja"
      description=""
      categories={categories}
    />
  </div>
 );
}
```

DetailFood.js

```
import React, { useEffect,
useState } from "react";
import { useNavigate, useParams
} from "react-router-dom";
import API from
"../../../config/api";
import DetailAdmin from
"../../../MainComponent/adminCo
mponent/detailAdmin";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import Breadcrumb from
"../../../MainComponent/adminCo
mponent/breadcrumb";
import Swal from "sweetalert2";
import AddCategory from
"../../../MainComponent/adminCo
mponent/addCategory";
import LoaderPage from
"../../loader";
import { storage } from
"../../../config/firebase";
import { ref, uploadBytes,
getDownloadURL } from
"firebase/storage";
const DetailFood = () => {
 const { id } = useParams();
 const navigate = useNavigate();

 const [data, setData] =
useState(null);
 const [categories, setCategories]
= useState([]);
 const [loadingPage,
setLoadingPage] =
useState(true);
 const [loadingUpdate,
setLoadingUpdate] =
useState(false);
 const [loadingDelete,
setLoadingDelete] =
useState(false);
 const [loadingCategory,
setLoadingCategory] =
useState(false);

 useEffect(() => {
  loadPageData();
 }, [id]);

 const loadPageData = async ()
=> {
  try {
   setLoadingPage(true);
   const [detailRes,
categoryRes] = await
Promise.all([
    API.get(`/food/${id}`),
    API.get(`/food/categories`),
   ]);

   setData(detailRes.data);
   const transformed =
categoryRes.data.map((item) =>
({
    value: item.id,
    label: item.name_category,
   }));
```

```
      setCategories(transformed);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Gagal memuat data",
        text: "Periksa koneksi atau
coba lagi nanti.",
      });
    } finally {
      setLoadingPage(false);
    }
  };

  const createFoodCategory =
async (title) => {
    try {
      setLoadingCategory(true);

      await
API.post("/food/category", {
name: title });
      await loadPageData();

      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Kategori Makanan
berhasil ditambahkan.",
      });
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Gagal",
        text:
err.response?.data?.message ||
"Gagal menambahkan kategori.",
      });
    } finally {
      setLoadingCategory(false);
    }
  };

  const updateText = async (title,
category, description, images) =>
{
    try {
      await
setLoadingUpdate(true);

      await API.put(`/food/${id}`,
{ title, category, description });

      const editedImages =
images.filter(
        (img) => img.isEdit &&
img.id && img.file
      );
      for (const img of
editedImages) {
        await
updateSingleImage(img.id,
img.file);
      }

      const newImages =
images.filter((img) =>
img.isNew && img.file);
      for (const img of newImages)
{
        await addImage(img.file);
      }

      const deletedImages =
images.filter((img) =>
img.isDeleted && img.id);
      for (const img of
deletedImages) {
        await
deleteSingleImage(img.id);
      }

      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Data berhasil
diperbarui.",
      });
      setLoadingUpdate(false);
```

```
      navigate(`/admin/makanan/`
);

    loadPageData(); // refresh
data
   } catch (err) {
    console.error(err);
    Swal.fire({
     icon: "error",
     title: "Gagal",
     text: "Gagal memperbarui
data!",
    });
   } finally {
    setLoadingUpdate(false);
   }
  };

  const addImage = async (file)
=> {
   try {
    const storageRef =
ref(storage,
`foods/${Date.now()}-
${file.name}`);
    const snapshot = await
uploadBytes(storageRef, file);
    const downloadURL = await
getDownloadURL(snapshot.ref);

    await
API.post(`/food/${id}/image`, {
     images: [downloadURL], //
karena backend expects array of
URLs
    });
   } catch (err) {
    console.error("Gagal
menambahkan gambar:", err);
   }
  };

  const updateSingleImage =
async (imageId, file) => {
   try {
    const storageRef =
ref(storage,
`foods/${Date.now()}-
${file.name}`);
    const snapshot = await
uploadBytes(storageRef, file);
    const downloadURL = await
getDownloadURL(snapshot.ref);

    await
API.put(`/food/${id}/image/${im
ageId}`, {
     image: downloadURL, //
backend expects a single image
URL
    });
   } catch (err) {
    console.error("Gagal update
gambar:", err);
   }
  };

  const deleteSingleImage =
async (imageId) => {
   await
API.delete(`/food/${id}/image/${
imageId}`);
  };

  const deleteFood = async () =>
{
   try {
    const confirm = await
Swal.fire({
     title: "Hapus Makanan?",
     text: "Tindakan ini tidak
dapat dibatalkan!",
     icon: "warning",
     showCancelButton: true,
     confirmButtonColor:
"#d33",
     cancelButtonColor:
"#3085d6",
     confirmButtonText: "Ya,
hapus!",
     cancelButtonText: "Batal",
```

```
    });

    if (confirm.isConfirmed) {
      setLoadingDelete(true);
      await
API.delete(`/food/${id}`);
      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Makanan berhasil
dihapus.",
      });
      navigate(`/admin/makanan/`
);
    }
  } catch (error) {
    console.error(error);
    Swal.fire({
      icon: "error",
      title: "Gagal!",
      text: "Terjadi kesalahan saat
menghapus makanan.",
    });
  } finally {
    setLoadingDelete(false);
  }
 };

 if
(loadingPage||loadingCategory||lo
adingDelete||loadingUpdate) {
  return (
    <div className="flex
justify-center items-center h-
screen">
      <LoaderPage />
    </div>
  );
 }

 if (!data) return <p>Data tidak
ditemukan</p>;

 return (
   <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-4 bg-white">
     <Breadcrumb
       items={[
         { label: "Dashboard", href:
"/admin" },
         { label: "Makanan", href:
"/admin/makanan" },
         { label: "Detail" },
       ]}
     />
     <HeaderAdmin
title={"Update Detail Data
Makanan"} />
     <div className="w-full h-
auto flex justify-between items-
start gap-6">
       <div className="w-
[60%]">
         <DetailAdmin
           data={data}
           deleteData={deleteFood}
           updateText={updateText}
           categories={categories}
           topic={"makanan"}
           loadingUpdate={loading
Update}
           loadingDelete={loadingD
elete}
         />
       </div>
       <div className="w-
[40%]">
         <AddCategory
addItem={createFoodCategory}
loading={loadingCategory} />
       </div>
     </div>
   </div>
 );
};

export default DetailFood;
```

AddFood.js

```
import React, { useEffect,
useState } from "react";
import { useParams, useNavigate
} from "react-router-dom";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import Breadcrumb from
"../../../MainComponent/adminCo
mponent/breadcrumb";
import Swal from "sweetalert2";
import AddItems from
"../../../MainComponent/adminCo
mponent/addItems";
import AddCategory from
"../../../MainComponent/adminCo
mponent/addCategory";
import LoaderPage from
"../../loader";
import { storage } from
"../../../config/firebase";
import { ref, uploadBytes,
getDownloadURL } from
"firebase/storage";
const AddFood = () => {
  const navigate = useNavigate();

  const [categories, setCategories]
= useState([]);
  const [loadingPage,
setLoadingPage] =
useState(true);
  const [loadingAdd,
setLoadingAdd] =
useState(false);
  const [loadingCategory,
setLoadingCategory] =
useState(false);

  useEffect(() => {
    getCategories();
  }, []);

  const getCategories = async ()
=> {
    try {
      setLoadingPage(true);
      const res = await
API.get(`/food/categories`);
      const transformed =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformed);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat
kategori makanan.",
      });
    } finally {
      setLoadingPage(false);
    }
  };

const createFood = async (title,
description, category, images) =>
{
  try {
    setLoadingAdd(true);

    // 1. Kirim data makanan ke
backend (tanpa gambar)
    const response = await
API.post("/food", {
      name: title,
      description: description,
      category: category,
    });

    const newFoodId =
response.data.id;
    const imageUrls = [];
```

```
    // 2. Upload gambar ke
Firebase dan ambil URL-nya
    if (images && images.length >
0) {
      for (const img of images) {
        if (!img.file) continue;

        try {
          const storageRef = ref(
            storage,
            `foods/${Date.now()}-
${img.file.name}`
          );
          const snapshot = await
uploadBytes(storageRef,
img.file);
          const downloadURL =
await
getDownloadURL(snapshot.ref);
          imageUrls.push(download
URL);
        } catch (err) {
          console.error(`Gagal
upload ke Firebase:
${img.file.name}`, err);
        }
      }
    }

    // 3. Kirim URL gambar ke
backend
    if (imageUrls.length > 0) {
      await
API.post(`/food/${newFoodId}/i
mage`, { images: imageUrls });
    }

    await Swal.fire({
      icon: "success",
      title: "Berhasil!",
      text: "Makanan dan gambar
berhasil ditambahkan.",
    });

    navigate(`/admin/makanan/`);
  } catch (err) {
    console.error("Error utama:",
err);
    Swal.fire({
      icon: "error",
      title: "Gagal",
      text:
err.response?.data?.message ||
"Gagal menambahkan
makanan.",
    });
  } finally {
    setLoadingAdd(false);
  }
};

  const createFoodCategory =
async (title) => {
    try {
      setLoadingCategory(true);



      await
API.post("/food/category",  {
name: title });
      await getCategories();

      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Kategori Makanan
berhasil ditambahkan.",
      });
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Gagal",
        text:
err.response?.data?.message ||
"Gagal menambahkan kategori.",
      });
    } finally {
      setLoadingCategory(false);
    }
  };
```

```
  if
(loadingPage||loadingAdd||loadin
gCategory) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-4 bg-white">
      <Breadcrumb
        items={[
          { label: "Dashboard", href:
"/admin" },
          { label: "Makanan", href:
"/admin/makanan" },
          { label: "Tambah
Makanan" },
        ]}
      />
      <HeaderAdmin
title={"Tambah Data Makanan"}
/>
      <div className="w-full h-
auto flex justify-between items-
start gap-6">
        <div className="w-
[60%]">
          <AddItems
            addItem={createFood}
            categories={categories}
            loading={loadingAdd}
          />
        </div>
        <div className="w-
[40%]">
          <AddCategory
addItem={createFoodCategory}
loading={loadingCategory} />
        </div>
      </div>
    </div>
  );
};

export default AddFood;
```

12. Tarian

DancePAge.js

```
import { useEffect, useState }
from "react";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import ListCardAdmin from
"../../../MainComponent/adminCo
mponent/listCardAdmin";
import Swal from "sweetalert2";
import LoaderPage from
"../../loader";

export default function
DancePage() {
  const [data, setData] =
useState([]);
  const [error, setError] =
useState("");
  const [loading, setLoading] =
useState(false);

  const [categories, setCategories]
= useState([]);

  useEffect(() => {
    getCategories();
  }, []);
```

```
  const getCategories = async ()
=> {
    try {
      setLoading(true);
      const res = await
API.get(`/dance/categories`);
      const transformedData =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformedDat
a);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat
kategori Tarian.",
      });
    } finally {
      setLoading(false);
    }
  };

  const getData = async () => {
    try {
      setLoading(true); //
tambahkan loading saat load data
juga
      const res = await
API.get("/dance");
      setData(res.data);
      setError("");
    } catch (err) {
      console.error("Error loading
data:", err);
      setError("Gagal memuat
data");
    } finally {
      setLoading(false);
    }
  };

  useEffect(() => {
    getData();
  }, []);

  if (loading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-10 bg-white">
      <HeaderAdmin
title={"Kelola Data Tarian
Tradisional"} />
      <ListCardAdmin
        topic={"tarian"}
        data={data}
        image="https://res.cloudinar
y.com/diipdl14x/image/upload/v
1751567242/filosofi_dlh9dl.jpg"
        title="Tarian Tradisional
Khas Budaya Toraja"
        description=""
        categories={categories}
      />
    </div>
  );
}
```

AddPage.js

```
import React, { useEffect,
useState } from "react";

import { useParams, useNavigate
} from "react-router-dom";
```

```
import API from
"../../../config/api";
import DetailAdmin from
"../../../MainComponent/adminCo
mponent/detailAdmin";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import Breadcrumb from
"../../../MainComponent/adminCo
mponent/breadcrumb";
import Swal from "sweetalert2";
import AddItems from
"../../../MainComponent/adminCo
mponent/addItems";
import AddCategory from
"../../../MainComponent/adminCo
mponent/addCategory";
import LoaderPage from
"../../loader";
import { storage } from
"../../../config/firebase";
import { ref, uploadBytes,
getDownloadURL } from
"firebase/storage";
const AddDance = () => {
  const { id } = useParams();
  const navigate = useNavigate();
  const [categories, setCategories]
= useState([]);

  // Ubah: dua state terpisah
  const [loadingDance,
setLoadingDance] =
useState(false);
  const [loadingCategory,
setLoadingCategory] =
useState(false);
  const [loadingPage,
setLoadingPage] =
useState(false); // untuk fetch
awal

  useEffect(() => {
    getCategories();
  }, []);

  const getCategories = async ()
=> {
    try {
      setLoadingPage(true);
      const res = await
API.get(`/dance/categories`);
      const transformedData =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformedDat
a);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat
kategori tarian.",
      });
    } finally {
      setLoadingPage(false);
    }
  };

  const createDance = async (title,
description, category, images) =>
{
    try {
      setLoadingDance(true);

      const response = await
API.post("/dance", {
        name: title,
        description: description,
        category: category,
      });
      const newDanceId =
response.data.id;

      const imageUrls = [];

      // Upload gambar ke Firebase
```

```
    if (images && images.length
> 0) {
      for (const img of images) {
        if (!img.file) continue;

        try {
          const storageRef = ref(
            storage,
            `dances/${Date.now()}-
${img.file.name}`
          );
          const snapshot = await
uploadBytes(storageRef,
img.file);
          const downloadURL =
await
getDownloadURL(snapshot.ref);
          imageUrls.push(downloa
dURL);
        } catch (err) {
          console.error(`Gagal
upload ke Firebase:
${img.file.name}`, err);
        }
      }
    }

    // Kirim URL gambar ke
backend
    if (imageUrls.length > 0) {
      await
API.post(`/dance/${newDanceId
}/image`, { images: imageUrls
});
    }

    await Swal.fire({
      icon: "success",
      title: "Berhasil!",
      text: "Tarian dan gambar
berhasil ditambahkan.",
    });

    navigate(`/admin/tarian/`);
  } catch (err) {
    console.error("Error utama:",
err);
    Swal.fire({
      icon: "error",
      title: "Gagal",
      text:
err.response?.data?.message ||
"Gagal menambahkan tarian.",
    });
  } finally {
    setLoadingDance(false);
  }
 };

 const createDanceCategory =
async (title) => {
  try {
    setLoadingCategory(true);

    await
API.post("/dance/category", {
name: title });
    await getCategories();

    await Swal.fire({
      icon: "success",
      title: "Berhasil!",
      text: "Kategori tarian
berhasil ditambahkan.",
    });
  } catch (err) {
    console.error("Error utama:",
err);
    Swal.fire({
      icon: "error",
      title: "Gagal",
      text:
        err.response?.data?.messag
e || "Gagal menambahkan
kategori tarian.",
    });
  } finally {
    setLoadingCategory(false);
  }
 };
```

```
  // Tampilkan loader jika sedang load awal atau proses
  if (loadingPage || loadingDance || loadingCategory) {
    return (
      <div className="flex justify-center items-center h-screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-screen overflow-y-auto snap-y snap-mandatory font-montserrat p-4 bg-white">
      <Breadcrumb
        items={[
          { label: "Dashboard", href: "/admin" },
          { label: "Tarian", href: "/admin/tarian" },
          { label: "Tambah Tarian" },
        ]}
      />
      <HeaderAdmin title={"Tambah Data Tarian"} />
      <div className="w-full h-auto flex justify-between items-start gap-6">
        <div className="w-[60%]">
          <AddItems
            addItem={createDance}
            categories={categories}
            loading={loadingDance}
          />
        </div>
        <div className="w-[40%]">
          <AddCategory
            addItem={createDanceCategory}
            loading={loadingCategory}
          />
        </div>
      </div>
    </div>
  );
};

export default AddDance;
```

DetailDance.js

```
import React, { useEffect, useState } from "react";
import { useNavigate, useParams } from "react-router-dom";
import API from "../../../config/api";
import DetailAdmin from "../../../MainComponent/adminComponent/detailAdmin";
import HeaderAdmin from "../../../MainComponent/adminComponent/headerAdmin";
import Breadcrumb from "../../../MainComponent/adminComponent/breadcrumb";
import Swal from "sweetalert2";
import AddCategory from "../../../MainComponent/adminComponent/addCategory";
import LoaderPage from "../../loader";
import { storage } from "../../../config/firebase";
import { ref, uploadBytes, getDownloadURL } from "firebase/storage";

const DetailDance = () => {
  const { id } = useParams();
  const navigate = useNavigate();
```

```
  const [data, setData] =
useState(null);
  const [categories, setCategories]
= useState([]);

  const [loadingPage,
setLoadingPage] =
useState(true);
  const [loadingCategory,
setLoadingCategory] =
useState(false);

  // Load detail data
  const loadData = async () => {
    try {
      setLoadingPage(true);
      const res = await
API.get(`/dance/${id}`);
      setData(res.data);
    } catch (err) {
      console.error(err);
      Swal.fire({ icon: "error",
title: "Gagal memuat data." });
    } finally {
      setLoadingPage(false);
    }
  };

  const getCategories = async ()
=> {
    try {
      const res = await
API.get(`/dance/categories`);
      const transformed =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformed);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat
kategori tarian.",
      });
    }
  };

  useEffect(() => {
    const loadAll = async () => {
      setLoadingPage(true);
      await
Promise.all([loadData(),
getCategories()]);
      setLoadingPage(false);
    };
    loadAll();
  }, [id]);

  const createDanceCategory =
async (title) => {
    try {
      setLoadingCategory(true);

      await
API.post("/dance/category", {
name: title });
      await getCategories();
      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Kategori tarian
berhasil ditambahkan.",
      });
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Gagal",
        text:
err.response?.data?.message ||
"Gagal menambahkan kategori.",
      });
    } finally {
      setLoadingCategory(false);
    }
  };
```

```
  const updateText = async (title, category, description, images) => {
    setLoadingPage(true);
    try {
      await API.put(`/dance/${id}`, { title, category, description });

      const edited = images.filter((img) => img.isEdit && img.id && img.file);
      for (const img of edited) {
        await updateSingleImage(img.id, img.file);
      }

      const added = images.filter((img) => img.isNew && img.file);
      for (const img of added) {
        await addImage(img.file);
      }

      const deleted = images.filter((img) => img.isDeleted && img.id);
      for (const img of deleted) {
        await deleteSingleImage(img.id);
      }

      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Data berhasil diperbarui.",
      });
      setLoadingPage(false);
      navigate(`/admin/tarian/`);

      loadData(); // refresh
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Oops...",
        text: "Gagal memperbarui data!",
      });
      setLoadingPage(false);
    }
  };

  const addImage = async (file) => {
    try {
      const storageRef = ref(storage, `dances/${Date.now()}-${file.name}`);
      const snapshot = await uploadBytes(storageRef, file);
      const downloadURL = await getDownloadURL(snapshot.ref);

      await API.post(`/dance/${id}/image`, {
        images: [downloadURL], // karena backend expects array of URLs
      });
    } catch (err) {
      console.error("Gagal menambahkan gambar:", err);
    }
  };

  const updateSingleImage = async (imageId, file) => {
    try {
      const storageRef = ref(storage, `dances/${Date.now()}-${file.name}`);
      const snapshot = await uploadBytes(storageRef, file);
      const downloadURL = await getDownloadURL(snapshot.ref);
```

```
      await
API.put(`/dance/${id}/image/${i
mageId}`, {
        image: downloadURL, //
backend expects a single image
URL
      });
    } catch (err) {
      console.error("Gagal update
gambar:", err);
    }
  };

  const deleteSingleImage =
async (imageId) => {
    await
API.delete(`/dance/${id}/image/
${imageId}`);
  };

  const deleteDance = async () =>
{
    try {
      const result = await
Swal.fire({
        title: "Hapus Tarian?",
        text: "Tindakan ini tidak
dapat dibatalkan!",
        icon: "warning",
        showCancelButton: true,
        confirmButtonColor:
"#d33",
        cancelButtonColor:
"#3085d6",
        confirmButtonText: "Ya,
hapus!",
      });
      if (result.isConfirmed) {
        await
API.delete(`/dance/${id}`);
        await Swal.fire({
          title: "Berhasil!",
          text: "Tarian berhasil
dihapus.",
          icon: "success",
        });
        navigate(`/admin/tarian/`);
      }
    } catch (error) {
      console.error(error);
      Swal.fire({
        title: "Gagal!",
        text: "Terjadi kesalahan saat
menghapus.",
        icon: "error",
      });
    }
  };

  if
(loadingPage||loadingCategory) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-4 bg-white">
      <Breadcrumb
        items={[
          { label: "Dashboard", href:
"/admin" },
          { label: "Tarian", href:
"/admin/tarian" },
          { label: "Detail" },
        ]}
      />
      <HeaderAdmin
title={"Update Detail Data
Tarian "} />
      <div className="w-full h-
auto flex justify-between items-
start gap-6">
        <div className="w-
[60%]">
```

```
          <DetailAdmin
            data={data}
            deleteData={deleteDance
}
            updateText={updateText}
            categories={categories}
            topic={"Tarian"}
          />
        </div>
        <div className="w-
[40%]">
          <AddCategory
            addItem={createDanceCa
tegory}
            loading={loadingCategor
y}
          />
        </div>
      </div>
    </div>
  );
};

export default DetailDance;
```

13. Wisata

TourPage.js

```
import { useEffect, useState }
from "react";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import ListCardAdmin from
"../../../MainComponent/adminCo
mponent/listCardAdmin";
import LoaderPage from
"../../loader"; // pastikan file ini
ada
import Swal from "sweetalert2";

export default function
TourPage() {
  const [data, setData] =
useState([]);
  const [newTitle, setNewTitle] =
useState("");
  const [newDesc, setNewDesc] =
useState("");
  const [error, setError] =
useState("");
  const [loading, setLoading] =
useState(false);

  const [categories, setCategories]
= useState([]);

  useEffect(() => {
    getCategories();
    getData();
  }, []);

  const getCategories = async ()
=> {
    try {
      setLoading(true);
      const res = await
API.get(`/tourist/categories`);
      const transformedData =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformedDat
a);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat
kategori wisata.",
      });
    } finally {
      setLoading(false);
    }
  };

  const getData = async () => {
    try {
```

```
      setLoading(true); //
tambahkan loading di sini juga
      const res = await
API.get("/tourist");
      setData(res.data);
      setError("");
    } catch (err) {
      console.error("Error loading
data:", err);
      setError("Gagal memuat
data");
    } finally {
      setLoading(false);
    }
  };

  if (loading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-10 bg-white">
      <HeaderAdmin
title={"Kelola Data Wisata
Daerah Toraja"} />
      <ListCardAdmin
        topic={"wisata"}
        data={data}
        image="https://res.cloudinar
y.com/diipdl14x/image/upload/v
1751567242/filosofi_dlh9dl.jpg"
        title="Wisata Daerah
Budaya Toraja"
        description=""
        categories={categories}
      />
    </div>
  );
}
```

AddTour.js

```
import React, { useEffect,
useState } from "react";
import { useNavigate } from
"react-router-dom";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import Breadcrumb from
"../../../MainComponent/adminCo
mponent/breadcrumb";
import Swal from "sweetalert2";
import AddItems from
"../../../MainComponent/adminCo
mponent/addItems";
import AddCategory from
"../../../MainComponent/adminCo
mponent/addCategory";
import LoaderPage from
"../../loader";
import { storage } from
"../../../config/firebase";
import { ref, uploadBytes,
getDownloadURL } from
"firebase/storage";
const AddTour = () => {
  const navigate = useNavigate();
  const [categories, setCategories]
= useState([]);
  const [loading, setLoading] =
useState(false);

  useEffect(() => {
    getCategories();
  }, []);
```

```
  const getCategories = async ()
=> {
    try {
      setLoading(true);
      const res = await
API.get(`/tourist/categories`);
      const transformedData =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformedDat
a);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat
kategori wisata.",
      });
    } finally {
      setLoading(false);
    }
  };

  const createTourist = async
(title, description, category,
images) => {
    try {
      setLoading(true);

      console.log(title);
      // 1. Kirim data wisata ke
backend
      const response = await
API.post("/tourist", {
        title: title,
        description: description,
        category: category,
      });

      const newTouristId =
response.data.id;
      const imageUrls = [];

      // 2. Upload gambar ke
Firebase dan simpan URL-nya
      if (images && images.length
> 0) {
        for (const img of images) {
          if (!img.file) continue;

          try {
            const storageRef = ref(
              storage,
              `tourists/${Date.now()}-
${img.file.name}`
            );
            const snapshot = await
uploadBytes(storageRef,
img.file);
            const downloadURL =
await
getDownloadURL(snapshot.ref);
            imageUrls.push(downloa
dURL);
          } catch (err) {
            console.error(`Gagal
upload ke Firebase:
${img.file.name}`, err);
          }
        }
      }

      // 3. Kirim URL gambar ke
backend
      if (imageUrls.length > 0) {
        await
API.post(`/tourist/${newTouristI
d}/image`, { images: imageUrls
});
      }

      // 4. Notifikasi sukses dan
navigasi
      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Wisata dan gambar
berhasil ditambahkan.",
      });
```

```
      navigate(`/admin/wisata/`);
    } catch (err) {
      console.error("Error utama:",
err);
      Swal.fire({
        icon: "error",
        title: "Gagal",
        text:
err.response?.data?.message ||
"Gagal menambahkan wisata.",
      });
    } finally {
      setLoading(false);
    }
  };

  const createTouristCategory =
async (title) => {
    try {
      setLoading(true);

      await
API.post("/tourist/category", {
name: title });
      await getCategories();
      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Kategori wisata
berhasil ditambahkan.",
      });
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Gagal",
        text:
          err.response?.data?.messag
e || "Gagal menambahkan
kategori wisata.",
      });
    } finally {
      setLoading(false);
    }
  };

  if (loading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-4 bg-white">
      <Breadcrumb
        items={[
          { label: "Dashboard", href:
"/admin" },
          { label: "Wisata", href:
"/admin/wisata" },
          { label: "Tambah Wisata"
},
        ]}
      />
      <HeaderAdmin
title="Tambah Data Wisata" />
      <div className="w-full h-
auto flex justify-between items-
start gap-6">
        <div className="w-
[60%]">
          <AddItems
            addItem={createTourist}
            categories={categories}
            loading={loading}
          />
        </div>
        <div className="w-
[40%]">
          <AddCategory
addItem={createTouristCategory
} />
        </div>
      </div>
```

```
    </div>
  );
};

export default AddTour;
```

DetailTour.js

```
import React, { useEffect,
useState } from "react";
import { useNavigate, useParams
} from "react-router-dom";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import Breadcrumb from
"../../../MainComponent/adminCo
mponent/breadcrumb";
import Swal from "sweetalert2";
import DetailAdmin from
"../../../MainComponent/adminCo
mponent/detailAdmin";
import AddCategory from
"../../../MainComponent/adminCo
mponent/addCategory";
import LoaderPage from
"../../loader";
import { storage } from
"../../../config/firebase";
import { ref, uploadBytes,
getDownloadURL } from
"firebase/storage";
const DetailTour = () => {
  const { id } = useParams();
  const navigate = useNavigate();
  const [data, setData] =
useState(null);
  const [categories, setCategories]
= useState([]);
  const [loading, setLoading] =
useState(false);

  useEffect(() => {
    getCategories();
  }, []);

  useEffect(() => {
    loadData();
  }, [id]);

  const loadData = async () => {
    try {
      setLoading(true);
      const res = await
API.get(`/tourist/${id}`);
      setData(res.data);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
        text: "Gagal memuat detail
data wisata.",
      });
    } finally {
      setLoading(false);
    }
  };

  const getCategories = async ()
=> {
    try {
      setLoading(true);
      const res = await
API.get(`/tourist/categories`);
      const transformed =
res.data.map((item) => ({
        value: item.id,
        label: item.name_category,
      }));
      setCategories(transformed);
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Error",
```

```
        text: "Gagal memuat
kategori wisata.",
      });
    } finally {
      setLoading(false);
    }
  };

  const createTourCategory =
async (title) => {
    try {
      setLoading(true);

      await
API.post("/tourist/category", {
name: title });
      await getCategories();
      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Kategori wisata
berhasil ditambahkan.",
      });
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Gagal",
        text:
          err.response?.data?.messag
e || "Gagal menambahkan
kategori wisata.",
      });
    } finally {
      setLoading(false);
    }
  };

  const updateText = async (title,
category, description, images) =>
{
    try {
      setLoading(true);
      await
API.put(`/tourist/${id}`, { title,
category, description });

      const editedImages =
images.filter(
        (img) => img.isEdit &&
img.id && img.file
      );
      for (const img of
editedImages) {
        await
updateSingleImage(img.id,
img.file);
      }

      const newImages =
images.filter((img) =>
img.isNew && img.file);
      for (const img of newImages)
{
        await addImage(img.file);
      }

      const deletedImages =
images.filter((img) =>
img.isDeleted && img.id);
      for (const img of
deletedImages) {
        await
deleteSingleImage(img.id);
      }

      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Data berhasil
diperbarui.",
      });
      navigate("/admin/wisata/");

      loadData();
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Oops...",
        text: "Gagal memperbarui
data!",
```

```
    });
   } finally {
    setLoading(false);
   }
  };

  const addImage = async (file)
=> {
   try {
    const storageRef =
ref(storage,
`tourists/${Date.now()}-
${file.name}`);
    const snapshot = await
uploadBytes(storageRef, file);
    const downloadURL = await
getDownloadURL(snapshot.ref);

    await
API.post(`/tourist/${id}/image`,
{
     images: [downloadURL], //
karena backend expects array of
URLs
    });
   } catch (err) {
    console.error("Gagal
menambahkan gambar:", err);
   }
  };

  const updateSingleImage =
async (imageId, file) => {
   try {
    const storageRef =
ref(storage,
`tourists/${Date.now()}-
${file.name}`);
    const snapshot = await
uploadBytes(storageRef, file);
    const downloadURL = await
getDownloadURL(snapshot.ref);

    await
API.put(`/tourist/${id}/image/${i
mageId}`, {
      image: downloadURL, //
backend expects a single image
URL
    });
   } catch (err) {
    console.error("Gagal update
gambar:", err);
   }
  };

  const deleteSingleImage =
async (imageId) => {
   await
API.delete(`/tourist/${id}/image/
${imageId}`);
  };

  const deleteTour = async () =>
{
   try {
    const result = await
Swal.fire({
      title: "Hapus Wisata?",
      text: "Apakah Anda yakin
ingin menghapus wisata ini?
Tindakan ini tidak dapat
dibatalkan!",
      icon: "warning",
      showCancelButton: true,
      confirmButtonColor:
"#d33",
      cancelButtonColor:
"#3085d6",
      confirmButtonText: "Ya,
hapus!",
      cancelButtonText: "Batal",
    });

    if (result.isConfirmed) {
      await
API.delete(`/tourist/${id}`);
      await Swal.fire({
       title: "Berhasil!",
       text: "Wisata berhasil
dihapus.",
       icon: "success",
```

```
        });
        navigate("/admin/wisata/");
      }
    } catch (err) {
      console.error(err);
      Swal.fire({
        title: "Gagal!",
        text: "Terjadi kesalahan saat menghapus wisata.",
        icon: "error",
      });
    }
  };

  if (loading) {
    return (
      <div className="flex justify-center items-center h-screen">
        <LoaderPage />
      </div>
    );
  }

  if (!data) return <p>Data tidak ditemukan</p>;

  return (
    <div className="relative h-screen overflow-y-auto snap-y snap-mandatory font-montserrat p-4 bg-white">
      <Breadcrumb
        items={[
          { label: "Dashboard", href: "/admin" },
          { label: "Wisata", href: "/admin/wisata" },
          { label: "Detail" },
        ]}
      />
      <HeaderAdmin title="Update Detail Data Wisata" />
      <div className="w-full h-auto flex justify-between items-start gap-6">
        <div className="w-[60%]">
          <DetailAdmin
            data={data}
            deleteData={deleteTour}
            updateText={updateText}
            categories={categories}
            topic="Wisata"
          />
        </div>
        <div className="w-[40%]">
          <AddCategory addItem={createTourCategory} />
        </div>
      </div>
    </div>
  );
};

export default DetailTour;
```

14. Filosofi

FilosofiPage.js

```
import { useEffect, useState } from "react";
import API from "../../../config/api";
import HeaderAdmin from "../../../MainComponent/adminComponent/headerAdmin";
import ListCardAdmin from "../../../MainComponent/adminComponent/listCardAdmin";
import LoaderPage from "../../loader"; // pastikan file ini ada
import Swal from "sweetalert2";
```

```
export default function PhilosophyPage() {
  const [data, setData] = useState([]);
  const [newTitle, setNewTitle] = useState("");
  const [newDesc, setNewDesc] = useState("");
  const [error, setError] = useState("");
  const [loading, setLoading] = useState(false);

  const [categories, setCategories] = useState([]);

  // useEffect(() => {
  //   getCategories();
  // }, []);

  // const getCategories = async () => {
  //   try {
  //     setLoading(true);
  //     const res = await API.get(`/tourist/categories`);
  //     const transformedData = res.data.map((item) => ({
  //       value: item.id,
  //       label: item.name_category,
  //     }));
  //     setCategories(transformedData);
  //   } catch (err) {
  //     console.error(err);
  //     Swal.fire({
  //       icon: "error",
  //       title: "Error",
  //       text: "Gagal memuat kategori.",
  //     });
  //   } finally {
  //     setLoading(false);
  //   }
  // };

  const getData = async () => {
    try {
      setLoading(true); // mulai loading
      const res = await API.get("/philosophy");
      setData(res.data);
      setError("");
    } catch (err) {
      console.error("Error loading data:", err);
      setError("Gagal memuat data");
    } finally {
      setLoading(false); // selesai loading
    }
  };

  useEffect(() => {
    getData();
  }, []);

  if (loading) {
    return (
      <div className="flex justify-center items-center h-screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-screen overflow-y-auto snap-y snap-mandatory font-montserrat p-10 bg-white">
      <HeaderAdmin title={"Kelola Data Wisata Daerah Toraja"} />
      <ListCardAdmin
        topic={"filosofi"}
        data={data}
```

```
      image="https://res.cloudinary.com/diipdl14x/image/upload/v1751567242/filosofi_dlh9dl.jpg"
      title="Filosofi Masyarakat Daerah Toraja"
      description=""
```

addPhilosophy.js

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import API from "../../../config/api";
import HeaderAdmin from "../../../MainComponent/adminComponent/headerAdmin";
import Breadcrumb from "../../../MainComponent/adminComponent/breadcrumb";
import Swal from "sweetalert2";
import AddItemsFilosofi from "../../../MainComponent/adminComponent/addItemsFilosofi";
import LoaderPage from "../../loader";

const AddPhilosophy = () => {
  const navigate = useNavigate();
  const [loading, setLoading] = useState(false);

  const createPhilosophy = async (title, description) => {
    try {
      setLoading(true);


      await API.post("/philosophy", {name:title, description});

      await Swal.fire({
        icon: "success",
        title: "Berhasil!",
        text: "Data Filosofi berhasil ditambahkan.",
      });

      navigate(`/admin/filosofi/`);
    } catch (err) {
      console.error("Error:", err);
      Swal.fire({
        icon: "error",
        title: "Gagal",
        text: err.response?.data?.message || "Gagal menambahkan data Filosofi.",
      });
    } finally {
      setLoading(false);
    }
  };

  if (loading) {
    return (
      <div className="flex justify-center items-center h-screen">
        <LoaderPage />
      </div>
    );
  }

  return (
    <div className="relative h-screen overflow-y-auto snap-y snap-mandatory font-montserrat p-4 bg-white">
      <Breadcrumb
        items={[
          categories={categories}
        />
      </div>
  );
}
```

```
        { label: "Dashboard", href:
"/admin" },
        { label: "Filosofi", href:
"/admin/filosofi" },
        { label: "Tambah Filosofi"
},
      ]}
    />
    <HeaderAdmin
title={"Tambah Data Filosofi"}
/>
    <div className="w-full h-
auto flex justify-center items-
start gap-6">
      <div className="w-
[60%]">
        <AddItemsFilosofi
addItem={createPhilosophy}
loading={loading} />
      </div>
    </div>
  </div>
 );
};

export default AddPhilosophy;
```

detailPhilosophy.js

```
import React, { useEffect,
useState } from "react";
import { useNavigate, useParams
} from "react-router-dom";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import Breadcrumb from
"../../../MainComponent/adminCo
mponent/breadcrumb";
import Swal from "sweetalert2";
import DetailAdminFilosofi from
"../../../MainComponent/adminCo
mponent/detailAdminFilosofi";
import LoaderPage from
"../../loader";

const DetailPhilosophy = () => {
 const { id } = useParams();
 const navigate = useNavigate();
 const [data, setData] =
useState(null);
 const [loading, setLoading] =
useState(false);

 // Load detail data
 const loadData = async () => {
   try {
    setLoading(true);
    const res = await
API.get(`/philosophy/${id}`);
    setData(res.data);
   } catch (err) {
    console.error(err);
    Swal.fire({
     icon: "error",
     title: "Error",
     text: "Gagal memuat data.",
    });
   } finally {
    setLoading(false);
   }
 };

 useEffect(() => {
  loadData();
 }, [id]);

 // Update title & description
 const updateText = async (title,
description) => {
   try {
    setLoading(true);
    await
API.put(`/philosophy/${id}`, {
title, description });
    await Swal.fire({
```

```
        icon: "success",
        title: "Berhasil!",
        text: "Data berhasil
diperbarui.",
      });
      loadData();
    } catch (err) {
      console.error(err);
      Swal.fire({
        icon: "error",
        title: "Oops...",
        text: "Gagal memperbarui
data!",
      });
    } finally {
      setLoading(false);
    }
  };

  const deletePhilosophy = async
() => {
    try {
      const result = await
Swal.fire({
        title: "Hapus Filosofi?",
        text: "Apakah Anda yakin
ingin menghapus Filosofi ini?
Tindakan ini tidak dapat
dibatalkan!",
        icon: "warning",
        showCancelButton: true,
        confirmButtonColor:
"#d33",
        cancelButtonColor:
"#3085d6",
        confirmButtonText: "Ya,
hapus!",
        cancelButtonText: "Batal",
      });

      if (result.isConfirmed) {
        await
API.delete(`/philosophy/${id}`);
        await Swal.fire({
          title: "Berhasil!",
          text: "Filosofi berhasil
dihapus.",
          icon: "success",
        });
        navigate(`/admin/filosofi/`);
      }
    } catch (error) {
      console.error(error);
      Swal.fire({
        title: "Gagal!",
        text: "Terjadi kesalahan saat
menghapus Filosofi.",
        icon: "error",
      });
    }
  };

  if (loading) {
    return (
      <div className="flex
justify-center items-center h-
screen">
        <LoaderPage />
      </div>
    );
  }

  if (!data) return <p>Data tidak
ditemukan</p>;

  return (
    <div className="relative h-
screen overflow-y-auto snap-y
snap-mandatory font-montserrat
p-4 bg-white">
      <Breadcrumb
        items={[
          { label: "Dashboard", href:
"/admin" },
          { label: "Filosofi", href:
"/admin/filosofi" },
          { label: "Detail" },
        ]}
      />
```

```
    <HeaderAdmin
title={"Update Detail Data
Filosofi"} />
    <div className="w-full h-
auto flex justify-center items-
start gap-6">
      <div className="w-
[60%]">
        <DetailAdminFilosofi
          data={data}
          deleteData={deletePhilos
ophy}
          updateText={updateText}
          topic={"filosofi"}
        />
      </div>
    </div>
  </div>
 );
};

export default DetailPhilosophy;
```

15. Pertanyaan

faqPage.js

```
// src/components/FAQList.jsx
import React, { useEffect,
useState } from "react";
import { FaChevronDown,
FaChevronUp } from "react-
icons/fa";
import { motion,
AnimatePresence } from
"framer-motion";
import Swal from "sweetalert2";
import API from
"../../../config/api";
import HeaderAdmin from
"../../../MainComponent/adminCo
mponent/headerAdmin";
import { FaInbox } from "react-
icons/fa"; // Tambahkan import
import { LuCopy } from "react-
icons/lu";
import { TbMailShare } from
"react-icons/tb";
import {
IoMdCloseCircleOutline } from
"react-icons/io";
import { FiSave } from "react-
icons/fi";
import { MdDelete } from "react-
icons/md";
const FAQPage = () => {
  const [openId, setOpenId] =
useState(null);
  const [questions, setQuestions]
= useState([]);
  const [loading, setLoading] =
useState(true);
  const [error, setError] =
useState("");
  const [answerInputs,
setAnswerInputs] = useState({});
  const [submittingAnswerId,
setSubmittingAnswerId] =
useState(null);

  const [filter, setFilter] =
useState("all");
  const unansweredCount =
questions.filter((q) =>
!q.answer).length;

  const filteredQuestions =
[...questions]
    .sort((a, b) => {
      // Belum terjawab
(null/empty) akan naik ke atas
      if (!a.answer && b.answer)
return -1;
      if (a.answer && !b.answer)
return 1;
      return 0;
```

```
    })
    .filter((q) => {
      if (filter === "unanswered")
return !q.answer;
      if (filter === "answered")
return q.answer;
      return true;
    });

  const toggleAnswer = (id) => {
    setOpenId(openId === id ?
null : id);
  };

  const fetchQuestions = async ()
=> {
    setLoading(true);
    try {
      const res = await
API.get("/question", {
        params: {
          status: undefined,
          page: 1,
          limit: 100,
        },
      });
      setQuestions(res.data.data);
      setError("");
    } catch (err) {
      console.error("[GET
QUESTIONS ERROR]", err);
      setError("Gagal memuat data
pertanyaan.");
    } finally {
      setLoading(false);
    }
  };

  const handleDelete = async (id)
=> {
    try {
      const confirm = await
Swal.fire({
        title: "Hapus Pertanyaan?",
        text: "Tindakan ini tidak
dapat dibatalkan!",
        icon: "warning",
        showCancelButton: true,
        confirmButtonColor:
"#d33",
        cancelButtonColor:
"#3085d6",
        confirmButtonText: "Ya,
hapus!",
        cancelButtonText: "Batal",
      });

      if (confirm.isConfirmed) {
        console.log(id);
        await
API.delete(`/question/${id}`);
        await Swal.fire({
          icon: "success",
          title: "Berhasil!",
          text: "Pertanyaan berhasil
dihapus.",
        });
      }
      await fetchQuestions();
    } catch (error) {
      console.error(error);
      Swal.fire({
        icon: "error",
        title: "Gagal!",
        text: "Terjadi kesalahan saat
menghapus Pertanyaan.",
      });
    } finally {
    }
  };

  const handleAnswerChange =
(id, value) => {
    setAnswerInputs((prev) => ({
      ...prev,
      [id]: value,
    }));
  };

  const handleAnswerSubmit =
async (id) => {
```

```
    const answer =
answerInputs[id];
    if (!answer || answer.trim()
=== "") {
      Swal.fire("Gagal", "Jawaban
tidak boleh kosong", "warning");
      return;
    }

    try {
      setSubmittingAnswerId(id);
      const res = await
API.put(`/question/${id}`, {
        answer,
        status: "answered", // atau
apa pun status yang kamu pakai
      });

      Swal.fire("Berhasil",
"Jawaban berhasil dikirim",
"success");

      // Refresh pertanyaan
      await fetchQuestions();
      setOpenId(null);
    } catch (err) {
      console.error("[UPDATE
QUESTION ERROR]", err);
      Swal.fire("Gagal", "Gagal
mengirim jawaban", "error");
    } finally {
      setSubmittingAnswerId(null);
    }
  };

  const handleUpdateSend = (id)
=> {
    setQuestions((prevQuestions)
=>
      prevQuestions.map((q) =>
(q.id === id ? { ...q, is_send: true
} : q))
    );
  };

  const handleUpdateClose = (id)
=> {
    setQuestions((prevQuestions)
=>
      prevQuestions.map((q) =>
(q.id === id ? { ...q, is_send:
false } : q))
    );
  };
  useEffect(() => {
    fetchQuestions();
  }, []);

  return (
    <div className="h-screen
overflow-y-auto snap-y snap-
mandatory p-10">
      <HeaderAdmin title={"Data
Pertanyaan "} />
      <motion.div
        initial={{ opacity: 0, y: -30
}}
        animate={{ opacity: 1, y: 0
}}
        transition={{ duration: 0.8,
ease: "easeOut" }}
      >
        <div className="max-w-
5xl mx-auto mt-10 bg-gray-100
p-6 rounded-2xl shadow-lg">
          <div className="flex
flex-col sm:flex-row sm:items-
center sm:justify-between mb-6
gap-4">
            <h1 className="text-xl
font-bold text-toraja-
merah">FAQ</h1>

            {/* Inbox Icon with
Badge */}
            <div className="flex
items-center gap-2">
              <FaInbox
className="text-red-500 text-
base" />
```

```
            <span className="text-red-600 font-semibold">
              {unansweredCount} belum dijawab
            </span>
          </div>
        </div>

        {/* Filter Buttons */}
        <div className="flex gap-2 mb-6">
          {[
            { label: "Semua", value: "all" },
            { label: "Belum Terjawab", value: "unanswered" },
            { label: "Sudah Terjawab", value: "answered" },
          ].map((btn) => (
            <button
              key={btn.value}
              onClick={() => setFilter(btn.value)}
              className={`px-4 py-2 rounded-full text-sm font-semibold transition ${
                filter === btn.value
                  ? "bg-toraja-merah text-white"
                  : "bg-white text-toraja-merah border border-toraja-merah"
              }`}
            >
              {btn.label}
            </button>
          ))}
        </div>

        {filteredQuestions.map((q) => (
          <div
            key={q.id}
            className={`mb-4 p-4 rounded-xl shadow-sm transition-all text-sm duration-300 h-auto ${
              openId === q.id ? "border border-toraja-merah" : ""
            } ${!q.answer ? "bg-red-100" : "bg-white"}`}
          >
            <div
              className="flex flex-col justify-between items-start cursor-pointer w-full"
              onClick={() => toggleAnswer(q.id)}
            >
              <h2
                className={`font-semibold transition-all ${
                  openId === q.id
                    ? "text-toraja-merah"
                    : !q.answer
                    ? "text-red-600"
                    : "text-gray-800"
                }`}
              >
                {q.email}
              </h2>
              <div
                className="flex w-full justify-between items-center cursor-pointer mt-2"
                onClick={() => toggleAnswer(q.id)}
              >
                <h3
                  className={`font-semibold transition-all text-xl ${
                    openId === q.id
                      ? "text-toraja-merah"
                      : !q.answer
                      ? "text-red-600"
                      : "text-gray-800"
                  }`}
                >
                  {q.question}
```

```
            </h3>

            {openId === q.id ? (
              <FaChevronUp
classNam="text-toraja-merah"
/>
            ) : (
              <FaChevronDown
classNam="text-toraja-merah"
/>
            )}
          </div>
        </div>

        {/* Animated Answer
Form */}
        <AnimatePresence>
          {openId === q.id && (
            <motion.div
              key="answer"
              initial={{ opacity: 0,
height: 0 }}
              animate={{ opacity:
1, height: "auto" }}
              exit={{ opacity: 0,
height: 0 }}
              transition={{
duration: 0.3 }}
              className="overflo
w-hidden mt-3 text-sm text-gray-
600"
            >
              {!q.is_send ? (
                <>
                  <textarea
                    className="w-
full p-3 border rounded-md mb-
3"
                    placeholder="Tul
is jawaban di sini..."
                    rows={4}
                    value={answerIn
puts[q.id] || q.answer ? q.answer :
""}
                    onChange={(e)
=>
                      handleAnswerC
hange(q.id, e.target.value)
                    }
                  />
                  <div
classNam="flex justify-start
gap-4 items-center w-full">
                    <button
                    classNam="bg
-toraja-merah text-white px-4 py-
2 rounded hover:bg-white
hover:text-toraja-merah
transition-all border border-
toraja-merah"
                    onClick={() =>
handleAnswerSubmit(q.id)}
                    disabled={sub
mittingAnswerId === q.id}
                  >
                    <div
classNam="flex items-center
gap-4">
                      <p
classNam=" text-sm">
                        {" "}
                        {submittingA
nswerId === q.id
                         ?
"Menyimpan..."
                         : "Simpan
Jawaban"}
                      </p>
                      <FiSave
classNam="text-xl " />
                    </div>
                  </button>
                  {q.answer && (
                    <a
                      href={`https://
mail.google.com/mail/?view=cm
&fs=1&to=${encodeURICompo
nent(
                        q.email
                      )}&su=${enco
deURIComponent(
```

```
                        "Balasan
Resmi dari Tim Toraja Budaya
atas Pertanyaan Anda"
                      )}&body=${e
ncodeURIComponent(
                        "Pertanyaan
Anda: " +
                          q.question +
                          "\nJawaban
Kami: " +
                          q.answer
                      )}`}
                      className="b
g-toraja-merah text-white px-4
py-2 rounded hover:bg-white
hover:text-toraja-merah
transition-all border border-
toraja-merah"
                      target="_blank
"
                      rel="noopener
noreferrer"
                    >
                      <div
className="flex items-center
gap-4">
                        <p
className="text-sm">Kirim Ke
Email</p>
                        <TbMailShar
e className="text-xl" />
                      </div>
                    </a>
                  )}
                  <div
className="flex justify-end w-
[60%] items-center">
                    <button
                      className="b
g-toraja-merah text-white px-4
py-2 rounded hover:bg-white
hover:text-toraja-merah
transition-all border border-
toraja-merah"
                      onClick={()
=> handleDelete(q.id)}
                      disabled={sub
mittingAnswerId === q.id}
                    >
                      <div
className="flex items-center
gap-4">
                        <p
className=" text-
sm">Hapus</p>
                        <MdDelete
className="text-xl " />
                      </div>
                    </button>
                  </div>
                </div>
              </>
            ) : (
              <>
                <div
className="mb-3 h-auto">
                  <label
className="font-medium text-
gray-700 block mb-1">
                    Subject:
                  </label>
                  <div
className="flex items-center
gap-2">
                    <div
className="bg-gray-100 p-2
rounded w-full">{`Menjawab
pertanyaan Anda`}</div>
                    <button
                      onClick={()
=>
                        navigator.clip
board.writeText(
                          "Menjawab
pertanyaan Anda"
                        )
                      }
                      className="
bg-red-200 p-4 rounded-xl"
                    >
```

```
                    <LuCopy
classNameName="text-toraja-merah
text-xl font-bold" />
                  </button>
                 </div>
                </div>

                <div
classNameName="mb-3">
                 <label
classNameName="font-medium text-
gray-700 block mb-1">
                  Isi Jawaban:
                 </label>
                 <div
classNameName="flex items-center
gap-2">
                  <div
classNameName="bg-gray-100 p-2
rounded w-full whitespace-pre-
line">
                   {q.answer}
                  </div>
                  <button
                   onClick={()
=>
                    navigator.clip
board.writeText(q.answer)
                   }
                   classNameName="
bg-red-200 p-4 rounded-xl"
                  >
                   <LuCopy
classNameName="text-toraja-merah
text-xl font-bold" />
                  </button>
                 </div>
                </div>
                <div
classNameName="flex justify-start
gap-4 items-center w-full">
                 <a
                  href={`https://
mail.google.com/mail/?view=cm
&fs=1&to=${
                   q.email
                  }&su=${encode
URIComponent(
                    "Balasan
Resmi dari Tim Toraja Budaya
atas Pertanyaan Anda"
                  )}&body=${enc
odeURIComponent(q.answer)}`}
                  classNameName="bg
-toraja-merah text-white px-4 py-
2 rounded hover:bg-white
hover:text-toraja-merah
transition-all border border-
toraja-merah"
                  target="_blank"
                  rel="noopener
noreferrer"
                 >
                  <div
classNameName="flex items-center
gap-4">
                   <p
classNameName=" text-sm">Kirim
Email</p>
                   <TbMailShare
classNameName="text-xl " />
                  </div>
                 </a>
                 <button
                  classNameName="bg
-toraja-merah text-white px-4 py-
2 rounded hover:bg-white
hover:text-toraja-merah
transition-all border border-
toraja-merah"
                  onClick={() =>
handleUpdateClose(q.id)}
                  disabled={sub
mittingAnswerId === q.id}
                 >
                  <div
classNameName="flex items-center
gap-4">
                   <p
classNameName=" text-
sm">Tutup</p>
```

```
                        <IoMdCloseCi
rcleOutline className="text-xl "
/>
                      </div>
                    </button>
                  </div>
                </>
              )}
            </motion.div>
          )}
        </AnimatePresence>
      </div>
    ))}
  </div>
 </motion.div>
</div>
);
};

export default FAQPage;
```