LAMPIRAN

DISTRIBUSI NILAI r_{tabel} SIGNIFIKANSI 5% dan 1%

N	The Level of Significance		N	The Level of Significance	
	5%	1%		5%	1%
3	0.997	0.999	38	0.320	0.413
4	0.950	0.990	39	0.316	0.408
5	0.878	0.959	40	0.312	0.403
6	0.811	0.917	41	0.308	0.398
7	0.754	0.874	42	0.304	0.393
8	0.707	0.834	43	0.301	0.389
9	0.666	0.798	44	0.297	0.384
10	0.632	0.765	45	0.294	0.380
11	0.602	0.735	46	0.291	0.376
12	0.576	0.708	47	0.288	0.372
13	0.553	0.684	48	0.284	0.368
14	0.532	0.661	49	0.281	0.364
15	0.514	0.641	50	0.279	0.361
16	0.497	0.623	55	0.266	0.345
17	0.482	0.606	60	0.254	0.330
18	0.468	0.590	65	0.244	0.317
19	0.456	0.575	70	0.235	0.306
20	0.444	0.561	75	0.227	0.296
21	0.433	0.549	80	0.220	0.286
22	0.432	0.537	85	0.213	0.278
23	0.413	0.526	90	0.207	0.267
24	0.404	0.515	95	0.202	0.263
25	0.396	0.505	100	0.195	0.256
26	0.388	0.496	125	0.176	0.230
27	0.381	0.487	150	0.159	0.210
28	0.374	0.478	175	0.148	0.194
29	0.367	0.470	200	0.138	0.181
30	0.361	0.463	300	0.113	0.148
31	0.355	0.456	400	0.098	0.128
32	0.349	0.449	500	0.088	0.115
33	0.344	0.442	600	0.080	0.105
34	0.339	0.436	700	0.074	0.097
35	0.334	0.430	800	0.070	0.091
36	0.329	0.424	900	0.065	0.086
37	0.325	0.418	1000	0.062	0.081

src/app/admin/page.tsx

```
"use client";

import { DataSiswa } from
"@/app/lib/interface";
import useSWR from "swr";
import ModalAdd from "./ModalAdd";
import DeleteSiswaButton from
"./DeleteSiswaButton";
import ModalUpdate from "./ModalUpdate";
import { useSearchParams } from
"next/navigation";
import ComponentPagination from
"./Pagination";
import { Button, Spinner } from "flowbite-
react";
import SearchInput from "./SearchInput";

export default function TebleStudents() {
const params = useSearchParams();
const queryParams = params.get("query");
const pageParams = params.get("page");

const fetcher = () =>
fetch(`/api/siswa?page=${pageParams ||
1}&query=${queryParams}`).then(
(res) => res.json()
);
const { data, error, isLoading } = useSWR(
`/api/siswa?page=${pageParams ||
1}&query=${queryParams}`,
fetcher
);

if (error)
return (
<div className="flex justify-center items-
center w-screen h-[60vh] text-center mt-4">
Failed to load
</div>
);
if (isLoading)
return (
<div className="w-screen h-[60vh] flex
items-center justify-center">
<div className="flex items-center justify-
center gap-4 text-center mt-4">
<Spinner color={"warning"} />
<p className="mt-2">loading table</p>
</div>
</div>
);
if (data.success === false) {
return (
<div className="flex flex-col gap-4 justify-
center items-center w-screen text-center mt-
4">
<h3>{data.message}</h3>
<Button className="bg-yellow-500
hover:bg-yellow-600">
Kembali ke halaman sebelumnya
</Button>
</div>
);
}
const totalPages = data.totalPages;
const totalSiswa = data.totalSiswa;

return (
<>
<div className="md:col-start-1 md:col-end-
3">
<div className="w-full flex gap-4 items-
center mb-4">
<ModalAdd />
<SearchInput />
</div>
<div className="relative min-h-fit max-h-
[20rem] overflow-y-scroll overflow-x-auto
rounded-xl mb-2">
<table className="w-full text-sm text-left
rtl:text-right text-gray-500 dark:text-gray-
400">
<thead className="text-xs text-gray-700
uppercase bg-gray-50 dark:bg-gray-700
dark:text-gray-400">
<tr>
<th scope="col" className="px-6 py-3">
Name
</th>
<th scope="col" className="px-6 py-3">
Jenis Kelamin
</th>
<th scope="col" className="px-6 py-3">
Nomor
</th>
<th scope="col" className="px-6 py-3">
Sudah Menilai
</th>
<th scope="col" className="px-6 py-3">
Action
```

```
</th>
</tr>
</thead>
<tbody>
{data.data.length === 0 ? (
<tr className="bg-white border-b dark:bg-
gray-800 dark:border-gray-700">
<td colSpan={5} className="px-6 py-4 text-
center">
Maaf, Siswa dengan nama "{queryParams}"
tidak ditemukan
</td>
</tr>
) : (
data.data?.map((item: DataSiswa, index:
number) => {
return (
<tr
key={index}
className="bg-white border-b dark:bg-gray-
800 dark:border-gray-700">
<th
scope="row"
className="w-[5rem] px-6 py-4 font-medium
text-gray-900 whitespace-nowrap dark:text-
white">
{item.name}
</th>
<td className="px-6 py-
4">{item.gender}</td>
<td className="px-6 py-
4">{item.phoneNumber}</td>
<td className="px-6 py-4">
{!item.status ? "Belum" : "Sudah"}
</td>
<td className="px-6 py-4 flex gap-4">
<ModalUpdate dataSiswa={item} />
<DeleteSiswaButton id={item.id} />
</td>
</tr>
);
})
)}
</tbody>
</table>
</div>
<div className="flex justify-center">
<ComponentPagination
totalPages={totalPages} />
</div>
<p className="mt-2 text-white font-semibold
text-center">
{totalSiswa} siswa terdaftar
</p>
</div>
</>
);
}
```

src/app/quesioner/page.tsx

```
"use client";

import { useEffect, useState } from "react";
import { useFormState } from "react-dom";
import Quesioner from "./Quesioner";
import { useRouter } from "next/navigation";
import WelcomeModal from
"./WelcomeModal";
import toast, { Toaster } from "react-hot-toast";
import {
checkPhoneNumberInQuesionerPage,
quesionerSubmit,
} from "@/app/lib/action";
import ButtonSubmit from "./ButtonSubmit";
import { Spinner } from "flowbite-react";

export default function PageQuesioner() {
const initialState = {
success: false,
message: "",
redirect: false,
};
const [state, formAction] =
useFormState(quesionerSubmit, initialState);
const [sessionId, setSessionId] =
useState<number>();
const [heRespondent, setheRespondent] =
useState<boolean>(false);
const router = useRouter();

async function checkStatusByID(id: number)
{
const result = await
checkPhoneNumberInQuesionerPage(id);
if (result?.status) {
sessionStorage.removeItem("id");
sessionStorage.removeItem("phone-number");
sessionStorage.removeItem("name");
toast.success("Jawaban anda berhasil
disimpan");
```

```
router.push("/hasil-analisis");
}
}

useEffect(() => {
const getSessionId =
Number(sessionStorage.getItem("id"));
const getSessionNumber =
sessionStorage.getItem("phone-number");
const getSessionName =
sessionStorage.getItem("name");
setSessionId(getSessionId);
checkStatusByID(getSessionId);
if (!getSessionName || !getSessionNumber
|| !getSessionId) {
router.push("/signin-quesioner");
} else {
setheRespondent(true);
}
}, []);

useEffect(() => {
if (
state.success === false &&
state.redirect === false &&
state.message.length > 0
) {
toast.error(state.message);
}
if (state.success === true && state.redirect
=== true) {
toast.success(state.message);
setTimeout(() => {
router.push("/hasil-analisis");
sessionStorage.removeItem("id");
sessionStorage.removeItem("phone-number");
sessionStorage.removeItem("name");
}, 3000);
}
}, [state]);

return heRespondent ? (
<>
<Toaster
position="top-center"
toastOptions={{ duration: 3000, style:
{ marginTop: "1rem" } }}
/>
<WelcomeModal />
<form className="grid grid-cols-1 gap-6 px-
4 xl:px-8" action={formAction}>
<input type="hidden" name="id-siswa"
defaultValue={sessionId} />
<div>
<h1 className="text-xl font-semibold text-
center mb-4">
Fasilitas Mobil Kursus
</h1>
<div className="grid grid-cols-1 md:grid-
cols-2 gap-4 ">
<Quesioner name="kebersihan-mobil"
kepentingan="Seberapa penting menurut anda
akan kebersihan mobil kursus"
kepuasan="Seberapa bersih mobil kursus yang
anda lihat dan rasakan" />
<Quesioner name="kelengkapan-performa-
alat-mobil"
kepentingan="Seberapa penting menurut anda
akan kelengkapan dan performa alat pada
mobil kursus"
kepuasan="Seberapa lengkap dan baik
performa alat mobil kursus menurut anda" />
<Quesioner name="performa-mobil"
kepentingan="Seberapa penting menurut anda
akan performa mobil kursus"
kepuasan="Seberapa baik performa mobil
kursus menurut anda" />
</div>
</div>

<div>
<h1 className="text-xl font-semibold text-
center mb-4 text-white">
Layanan Staff Lembaga kursus
</h1>
<div className="grid grid-cols-1 md:grid-
cols-2 gap-4 ">
<Quesioner
name="etika-sopan-santun-staff"
kepentingan="Seberapa penting menurut anda
staff kantor melayani siswa dengan penuh
sopan santun dan etika yang baik"
kepuasan="Seberapa baik etika dan sopan
santun staff kantor dalam melayani anda"
/>
<Quesioner name="skill-komunikasi-staff-
kantor"
kepentingan="Seberapa penting menurut anda
staff kantor memiliki skill komunikasi yang
baik"
kepuasan="Seberapa baik skill komunikasi
staff kantor menurut anda" />
```

```
<Quesioner
name="pelayanan-informasi-jadwal-belajar"
kepentingan="Seberapa penting menurut anda kemudahan dalam meperoleh informasi terkait kursus dan jadwal belajar"
kepuasan="Seberapa mudah menurut anda dalam memperoleh informasi terkait kursus dan jadwal belajar"
/>
</div>
</div>

<div>
<h1 className="text-xl font-semibold text-center mb-4">
Layanan Mentor kursus Mengemudi
</h1>
<div className="grid grid-cols-1 md:grid-cols-2 gap-4 ">
<Quesioner name="etika-sopan-santun-mentor-mengemudi"
kepentingan="Seberapa penting menurut anda mentor mengajari siswa dengan penuh sopan santun dan etika yang baik"
kepuasan="Seberapa baik etika & sopan santun mentor yang telah mengajari anda" />
<Quesioner name="skill-komunikasi-mentor-mengemudi"
kepentingan="Seberapa penting menurut anda mentor memiliki skill komunikasi yang baik"
kepuasan="Seberapa baik skill komunikasi mentor menurut anda" />
<Quesioner name="pengawasan-penuh-mentor-mengemudi"
kepentingan="Seberapa penting menurut anda mentor memberi pengawasan penuh kepada siswa saat proses belajar mengajar"
kepuasan="Seberapa penuh pengawasan mentor saat mengajari anda" />
<Quesioner name="pembawaan-materi-belajar-mentor-mengemudi"
kepentingan="Seberapa penting menurut anda mentor mampu memberi pemahaman yang mudah terkait materi belajar"
kepuasan="Seberapa mudah kah yang anda rasakan dalam memahami materi belajar yang diajarkan oleh mentor" />
</div>
</div>
<ButtonSubmit />
</form>
</>
) : (
<div className="w-screen h-screen flex items-center justify-center">
<div className="flex items-center justify-center gap-4 text-center mt-4">
<Spinner color={"warning"} size="xl" />
</div>
</div>
)
}
```

src/app/layout.tsx

```
import type { Metadata } from "next";
import localFont from "next/font/local";
import "./globals.css";
import Navbar from "./components/Navbar";
import Footers from "./components/Footer";
const geistSans = localFont({
src: "./fonts/GeistVF.woff",
variable: "--font-geist-sans",
weight: "100 900",
});
const geistMono = localFont({
src: "./fonts/GeistMonoVF.woff",
variable: "--font-geist-mono",
weight: "100 900",
});

export const metadata: Metadata = {
title: "Lestari Belajar",
description: "Generated by create next app",
};

export default async function RootLayout({
children,
}: Readonly<{
children: React.ReactNode;
}>) {

return (
<html lang="en">
<body
className={`${geistSans.variable} ${geistMono.variable} antialiased`}
>
<header className="bg-[#274C77]">
<Navbar />
</header>
<main className="bg-[#274C77]">{children}</main>
```

```
<footer className="bg-[#274C77]">
<Footers />
</footer>
</body>
</html>
);
}
```

src/app/lib/action.tsx

```
"use server";

import { revalidatePath } from "next/cache";
import { prisma } from "./prismaClient";
const bcrypt = require('bcrypt');
import { Gender } from '@prisma/client'
import { CheckNumberPhone, Item, Login,
tambahDataSiswaInterface,
TambahScheduleInterface, TaskFormat } from
"./interface";
import { checkPhoneNumberZod,
quesionerValidation,
tambahDataSiswaSchema,
updateDataSiswaSchema } from
"./schemaZod";
import { redirect } from "next/navigation";
import { promise } from "zod";

export async function
quesionerSubmit(prevState: any, formData:
FormData) {

const answerQuesioner =
quesionerValidation.safeParse({

// mobil kursus start

kepentingan_kebersihan_mobil:
Number(formData.get('kepentingan-
kebersihan-mobil')),
kinerja_kebersihan_mobil :
Number(formData.get('kinerja-kebersihan-
mobil')),
komentar_kepentingan_kebersihan_mobil :
formData.get('komentar-kepentingan-
kebersihan-mobil'),
komentar_kinerja_kepentingan_kebersihan_m
obil : formData.get('komentar-kinerja-
kebersihan-mobil'),
kepentingan_kelengkapan_performa_alat_mo
bil: Number(formData.get('kepentingan-
kelengkapan-performa-alat-mobil')),
kinerja_kelengkapan_performa_alat_mobil :
Number(formData.get('kinerja-kelengkapan-
performa-alat-mobil')),
komentar_kepentingan_kelengkapan_perform
a_alat_mobil : formData.get('komentar-
kepentingan-kelengkapan-performa-alat-
mobil'),
komentar_kinerja_kepentingan_kelengkapan_
performa_alat_mobil :
formData.get('komentar-kinerja-kelengkapan-
performa-alat-mobil'),
kepentingan_performa_mobil:
Number(formData.get('kepentingan-performa-
mobil')),
kinerja_performa_mobil :
Number(formData.get('kinerja-performa-
mobil')),
komentar_kepentingan_performa_mobil :
formData.get('komentar-kepentingan-
performa-mobil'),
komentar_kinerja_kepentingan_performa_mo
bil : formData.get('komentar-kinerja-
performa-mobil'),

// mobil kursus end

// staff lembaga kursus start

kepentingan_etika_sopan_santun:
Number(formData.get('kepentingan-etika-
sopan-santun-staff')),
kinerja_etika_sopan_santun :
Number(formData.get('kinerja-etika-sopan-
santun-staff')),
komentar_kepentingan_etika_sopan_santun :
formData.get('komentar-kepentingan-etika-
sopan-santun-staff'),
komentar_kinerja_kepentingan_etika_sopan_s
antun : formData.get('komentar-kinerja-etika-
sopan-santun-staff'),
kepentingan_pelayanan_informasi_jadwal_bel
ajar: Number(formData.get('kepentingan-
pelayanan-informasi-jadwal-belajar')),
kinerja_pelayanan_informasi_jadwal_belajar :
Number(formData.get('kinerja-pelayanan-
informasi-jadwal-belajar')),
komentar_kepentingan_pelayanan_informasi_
jadwal_belajar : formData.get('komentar-
```

```
kepentingan-pelayanan-informasi-jadwal-
belajar'),
komentar_kinerja_kepentingan_pelayanan_inf
ormasi_jadwal_belajar :
formData.get('komentar-kinerja-pelayanan-
informasi-jadwal-belajar'),
kepentingan_skill_komunikasi_staff:
Number(formData.get('kepentingan-skill-
komunikasi-staff-kantor')),
kinerja_skill_komunikasi_staff :
Number(formData.get('kinerja-skill-
komunikasi-staff-kantor')),
komentar_kepentingan_skill_komunikasi_staf
f : formData.get('komentar-kepentingan-skill-
komunikasi-staff-kantor'),
komentar_kinerja_kepentingan_skill_komunik
asi_staff : formData.get('komentar-kinerja-
skill-komunikasi-staff-kantor'),
// staff lembaga kursus end

// mentor kursus start

kepentingan_etika_sopan_santun_mentor:
Number(formData.get('kepentingan-etika-
sopan-santun-mentor-mengemudi')),
kinerja_etika_sopan_santun_mentor :
Number(formData.get('kinerja-etika-sopan-
santun-mentor-mengemudi')),
komentar_kepentingan_etika_sopan_santun_
mentor : formData.get('komentar-kepentingan-
etika-sopan-santun-mentor-mengemudi'),
komentar_kinerja_kepentingan_etika_sopan_s
antun_mentor : formData.get('komentar-
kinerja-etika-sopan-santun-mentor-
mengemudi'),

kepentingan_skill_komunikasi_mentor:
Number(formData.get('kepentingan-skill-
komunikasi-mentor-mengemudi')),
kinerja_skill_komunikasi_mentor :
Number(formData.get('kinerja-skill-
komunikasi-mentor-mengemudi')),
komentar_kepentingan_skill_komunikasi_men
tor : formData.get('komentar-kepentingan-
skill-komunikasi-mentor-mengemudi'),
komentar_kinerja_kepentingan_skill_komunik
asi_mentor : formData.get('komentar-kinerja-
skill-komunikasi-mentor-mengemudi'),

kepentingan_pengawasan_penuh_mentor:
Number(formData.get('kepentingan-
pengawasan-penuh-mentor-mengemudi')),
kinerja_pengawasan_penuh_mentor :
Number(formData.get('kinerja-pengawasan-
penuh-mentor-mengemudi')),
komentar_kepentingan_pengawasan_penuh_m
entor : formData.get('komentar-kepentingan-
pengawasan-penuh-mentor-mengemudi'),
komentar_kinerja_kepentingan_pengawasan_
penuh_mentor : formData.get('komentar-
kinerja-pengawasan-penuh-mentor-
mengemudi'),
kepentingan_pembawaan_materi_belajar_men
tor: Number(formData.get('kepentingan-
pembawaan-materi-belajar-mentor-
mengemudi')),
kinerja_pembawaan_materi_belajar_mentor :
Number(formData.get('kinerja-pembawaan-
materi-belajar-mentor-mengemudi')),
komentar_kepentingan_pembawaan_materi_b
elajar_mentor : formData.get('komentar-
kepentingan-pembawaan-materi-belajar-
mentor-mengemudi'),
komentar_kinerja_kepentingan_pembawaan_
materi_belajar_mentor :
formData.get('komentar-kinerja-pembawaan-
materi-belajar-mentor-mengemudi'),
// staff lembaga kursus end

id_siswa : Number(formData.get("id-siswa"))

});

if (!answerQuesioner.success) {
return{
success: false,
errors:
answerQuesioner.error.flatten().fieldErrors,
message: 'Terdapat inputan yang belum diisi,
pengiriman jawaban anda gagal',
redirect: false
}
}
// const scriptRegex =
/(<script.*?>[\s\S]*?<\/script>|\b(eval|setTime
out|setInterval)\(.*?\)|data:image\/svg\+xml;.*?
base64.*?>)/gi;
// const isInvalid =
scriptRegex.test(answerQuesioner.data?.kome
ntar_etika_sopan_santun as string) ||
```

```
scriptRegex.test(answerQuesioner.data?.komentar_pelayanan_informasi_jadwal_belajar as string) ||
scriptRegex.test(answerQuesioner.data?.komentar_skill_komunikasi_staff as string) ||
scriptRegex.test(answerQuesioner.data?.komentar_kebersihan_mobil as string) ||
scriptRegex.test(answerQuesioner.data?.komentar_performa_mobil as string)||
scriptRegex.test(answerQuesioner.data?.komentar_etika_sopan_santun_mentor as string) ||
scriptRegex.test(answerQuesioner.data?.komentar_kelengkapan_performa_alat_mobil as string) ||
scriptRegex.test(answerQuesioner.data?.komentar_pembawaan_materi_belajar_mentor as string) ||
scriptRegex.test(answerQuesioner.data?.komentar_pengawasan_penuh_mentor as string) ||
scriptRegex.test(answerQuesioner.data?.komentar_skill_komunikasi_mentor as string);

// if(isInvalid){
// return{
// success: false,
// message : "Karakter tidak valid terdeteksi. Harap jangan berlebihan dalam memberikan tanda baca.",
// redirect: false
// }
// }
try {
const komentarKebersihanMobil = [answerQuesioner?.data.komentar_kepentingan_kebersihan_mobil, answerQuesioner?.data.komentar_kinerja_kebersihan_mobil];
const komentarKelengkapanPerformaAlatMobil = [answerQuesioner?.data.komentar_kepentingan_kelengkapan_performa_alat_mobil, answerQuesioner?.data.komentar_kinerja_kelengkapan_performa_alat_mobil];
const komentarPerformaMobil = [answerQuesioner?.data.komentar_kepentingan_performa_mobil, answerQuesioner?.data.komentar_kinerja_performa_mobil];
const komentarEtikaSopanSantun = [answerQuesioner?.data.komentar_kepentingan_etika_sopan_santun, answerQuesioner?.data.komentar_kinerja_etika_sopan_santun];
const komentarSkillKomunikasiStaff = [answerQuesioner?.data.komentar_kepentingan_skill_komunikasi_staff, answerQuesioner?.data.komentar_kinerja_skill_komunikasi_staff];
const komentarPelayananInformasiJadwalBelajar = [answerQuesioner?.data.komentar_kepentingan_pelayanan_informasi_jadwal_belajar, answerQuesioner?.data.komentar_kinerja_pelayanan_informasi_jadwal_belajar];
const komentarEtikaSopanSantunMentor = [answerQuesioner?.data.komentar_kepentingan_etika_sopan_santun_mentor, answerQuesioner?.data.komentar_kinerja_etika_sopan_santun_mentor];
const komentarSkillKomunikasiMentor = [answerQuesioner?.data.komentar_kepentingan_skill_komunikasi_mentor, answerQuesioner?.data.komentar_kinerja_skill_komunikasi_mentor];
const komentarPembawaanMateriBelajarMentor = [answerQuesioner?.data.komentar_kepentingan_pembawaan_materi_belajar_mentor, answerQuesioner?.data.komentar_kinerja_pembawaan_materi_belajar_mentor];
const komentarPengawasanPenuhMentor = [answerQuesioner?.data.komentar_kepentingan_pengawasan_penuh_mentor, answerQuesioner?.data.komentar_kinerja_pengawasan_penuh_mentor];

await prisma.kebersihan_mobil.create({
data : {
kepentingan: answerQuesioner.data.kepentingan_kebersihan_mobil,
kinerja: answerQuesioner.data.kinerja_kebersihan_mobil,
komentar_kepentingan: komentarKebersihanMobil[0] || null,
komentar_kinerja: komentarKebersihanMobil[1] || null,
}
});

await prisma.kelengkapan_alat_mobil.create({
```

```
data : {
kepentingan:
answerQuesioner.data.kepentingan_kelengkap
an_performa_alat_mobil,
kinerja:
answerQuesioner.data.kinerja_kelengkapan_p
erforma_alat_mobil,
komentar_kepentingan:
komentarKelengkapanPerformaAlatMobil[0] ||
null,
komentar_kinerja:
komentarKelengkapanPerformaAlatMobil[1] ||
null,
}
});

await prisma.performa_mobil.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_performa_
mobil,
kinerja:
answerQuesioner.data.kinerja_performa_mobi
l,
komentar_kepentingan:
komentarPerformaMobil[0] || null,
komentar_kinerja: komentarPerformaMobil[1]
|| null,
}
});

await prisma.etika_SopanSantun.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_etika_sopa
n_santun,
kinerja:
answerQuesioner.data.kinerja_etika_sopan_sa
ntun,
komentar_kepentingan:
komentarEtikaSopanSantun[0] || null,
komentar_kinerja:
komentarEtikaSopanSantun[1] || null,
}
});

await prisma.pelayanan_administrasi.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_pelayanan
_informasi_jadwal_belajar,
kinerja:
answerQuesioner.data.kinerja_pelayanan_info
rmasi_jadwal_belajar,
komentar_kepentingan:
komentarPelayananInformasiJadwalBelajar[0]
|| null,
komentar_kinerja:
komentarPelayananInformasiJadwalBelajar[1]
|| null,
}
});

await
prisma.skill_komunikasi_baik_staff.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_skill_kom
unikasi_staff,
kinerja:
answerQuesioner.data.kinerja_skill_komunika
si_staff,
komentar_kepentingan:
komentarSkillKomunikasiStaff[0] || null,
komentar_kinerja:
komentarSkillKomunikasiStaff[1] || null,
}
});

await
prisma.etika_sopan_santun_mentor.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_etika_sopa
n_santun_mentor,
kinerja:
answerQuesioner.data.kinerja_etika_sopan_sa
ntun_mentor,
komentar_kepentingan:
komentarEtikaSopanSantunMentor[0] || null,
komentar_kinerja:
komentarEtikaSopanSantunMentor[1] || null,
}
});

await prisma.skill_komunikasi_baik.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_skill_kom
unikasi_mentor,
```

```
kinerja:
answerQuesioner.data.kinerja_skill_komunikasi_mentor,
komentar_kepentingan:
komentarSkillKomunikasiMentor[0] || null,
komentar_kinerja:
komentarSkillKomunikasiMentor[1] || null,
}
});

await prisma.pengawasan_penuh.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_pengawasan_penuh_mentor,
kinerja:
answerQuesioner.data.kinerja_pengawasan_penuh_mentor,
komentar_kepentingan:
komentarPengawasanPenuhMentor[0] || null,
komentar_kinerja:
komentarPengawasanPenuhMentor[1] || null,
}
});

await
prisma.pembawaan_materi_belajar_mentor.create({
data : {
kepentingan:
answerQuesioner.data.kepentingan_pembawaan_materi_belajar_mentor,
kinerja:
answerQuesioner.data.kinerja_pembawaan_materi_belajar_mentor,
komentar_kepentingan:
komentarPembawaanMateriBelajarMentor[0]
|| null,
komentar_kinerja:
komentarPembawaanMateriBelajarMentor[1]
|| null,
}
});

await
prisma.siswaKursusMengemudi.update({
where: {
id: answerQuesioner.data.id_siswa,
},
data:{
status: true
}
})

return{
message:"Jawaban anda berhasil disimpan",
success: true,
redirect:true
}
} catch (error) {
return{
message:"Inputan anda gagal tersimpan",
success: false,
redirect:false
}
}finally{
revalidatePath("/")
}
}

export async function
getAllQuesionerAnswer(){
function getAverangeValue(arr: Item[], name:
string): {kepentingan: number, kinerja:
number, nama: string, komentar_kepentingan:
(string|null)[], komentar_kinerja:
(string|null)[]}{
const totalKepentingan = arr.reduce((sum:
number, item) => sum + item.kepentingan, 0)
const rataRataKepentingan =
(totalKepentingan / arr.length).toFixed(2);
const totalKinerja = arr.reduce((sum: number,
item) => sum + item.kinerja, 0)
const rataRataKinerja = (totalKinerja /
arr.length).toFixed(2);
let fiveKomentar_kepentingan: (string|null)[]
= [];
arr.map((data, index) => {
index >= (arr.length - 171) ?
fiveKomentar_kepentingan.push(data.komentar_kepentingan) : null
})
let fiveKomentar_kinerja: (string|null)[] = [];
arr.map((data, index) => {
index >= (arr.length - 171) ?
fiveKomentar_kinerja.push(data.komentar_kinerja) : null
})
return {
nama : name,
kepentingan : Number(rataRataKepentingan),
kinerja: Number(rataRataKinerja),
```

```
komentar_kepentingan:
fiveKomentar_kepentingan,
komentar_kinerja: fiveKomentar_kinerja
}
}
const kebersihan_mobil = await
prisma.kebersihan_mobil.findMany();
const kelengkapan_performa_alat_mobil =
await
prisma.kelengkapan_alat_mobil.findMany();
const performa_mobil = await
prisma.performa_mobil.findMany();
const etika_sopan_santun_staff = await
prisma.etika_SopanSantun.findMany();
const
etika_sopan_santun_mentor_mengemudi =
await
prisma.etika_sopan_santun_mentor.findMany()
;
const skill_komunikasi_baik_staff = await
prisma.skill_komunikasi_baik_staff.findMany
();
const skill_komunikasi_baik_mentor = await
prisma.skill_komunikasi_baik.findMany();
const pengawasan_penuh_mentor = await
prisma.pengawasan_penuh.findMany();
const
pembawaan_materi_belajar_mentor_mengem
udi = await
prisma.pembawaan_materi_belajar_mentor.fin
dMany();
const pelayanan_informasi_jadwal_belajar =
await
prisma.pelayanan_administrasi.findMany();

const kordinatKebersihanMobil =
getAverangeValue(kebersihan_mobil,
"Kebersihan Mobil")
const
kordinatKelengkapanPerformaAlatMobil =
getAverangeValue(kelengkapan_performa_ala
t_mobil, "Kelengkapan Performa Alat Mobil")
const kordinatPerformaMobil =
getAverangeValue(performa_mobil, "Performa
Mobil")
const kordinatEtikaSopanSantunStaff =
getAverangeValue(etika_sopan_santun_staff,
"Etika dan Sopan Santun")
const
kordinatPelayananInformasiJadwalBelajar =
getAverangeValue(pelayanan_informasi_jadw
al_belajar, "Pelayanan Informasi Jadwal
Belajar")
const kordinatSkillKomunikasiBaikStaff =
getAverangeValue(skill_komunikasi_baik_staf
f, "Skill Komunikasi Staff")
const
kordinatEtikaSopanSantunMentorMengemudi
=
getAverangeValue(etika_sopan_santun_mento
r_mengemudi, "Etika dan Sopan Santun
Mentor Mengemudi")
const kordinatSkillKomunikasiBaikMentor =
getAverangeValue(skill_komunikasi_baik_me
ntor, "Skill Komunikasi Mentor")
const kordinatPengawasanPenuhMentor =
getAverangeValue(pengawasan_penuh_mentor,
"Pengawasan Penuh Mentor")
const
kordinatPembawaanMateriBelajarMentorMen
gemudi =
getAverangeValue(pembawaan_materi_belajar
_mentor_mengemudi, "Kemudahan
memahami materi")

const sumbuY = [
kordinatKebersihanMobil.kepentingan,
kordinatKelengkapanPerformaAlatMobil.kepe
ntingan,
kordinatPerformaMobil.kepentingan,
kordinatEtikaSopanSantunStaff.kepentingan,
kordinatSkillKomunikasiBaikStaff.kepentinga
n,
kordinatPelayananInformasiJadwalBelajar.kep
entingan,
kordinatEtikaSopanSantunMentorMengemudi.
kepentingan,
kordinatSkillKomunikasiBaikMentor.kepentin
gan,
kordinatPengawasanPenuhMentor.kepentinga
n,
kordinatPembawaanMateriBelajarMentorMen
gemudi.kepentingan]

const sumbux = [
kordinatKebersihanMobil.kinerja,
kordinatKelengkapanPerformaAlatMobil.kine
rja,
kordinatPerformaMobil.kinerja,
kordinatEtikaSopanSantunStaff.kinerja,
kordinatSkillKomunikasiBaikStaff.kinerja,
```

```
kordinatPelayananInformasiJadwalBelajar.kinerja,
kordinatEtikaSopanSantunMentorMengemudi.kinerja,
kordinatSkillKomunikasiBaikMentor.kinerja,
kordinatPengawasanPenuhMentor.kinerja,
kordinatPembawaanMateriBelajarMentorMengemudi.kinerja]

function getSumbuYX(sumbuY: number[], sumbuX: number[]): {sumbuY : number, sumbuX: number}{
const totalSumbuY = sumbuY.reduce((sum: number, item) => sum + item, 0)
const rataRataSumbuY = (totalSumbuY / sumbuY.length).toFixed(2);
const totalSumbuX = sumbuX.reduce((sum: number, item) => sum + item, 0)
const rataRataSumbuX = (totalSumbuX / sumbuX.length).toFixed(2);
return {
sumbuY : Number(rataRataSumbuY),
sumbuX: Number(rataRataSumbuX)
}
}

const garisPerbatasan = getSumbuYX(sumbuY, sumbux);

return{
kebersihanMobil: kordinatKebersihanMobil,
kelengkapanPerformaAlatMobil: kordinatKelengkapanPerformaAlatMobil,
performaMobil: kordinatPerformaMobil,
EtikaSopanSantunStaff: kordinatEtikaSopanSantunStaff,
skillKomunikasiBaikStaff: kordinatSkillKomunikasiBaikStaff,
pelayananInformasiJadwalBelajar: kordinatPelayananInformasiJadwalBelajar,
EtikaSopanSantunMentorMengemudi: kordinatEtikaSopanSantunMentorMengemudi,
skillKomunikasiBaikMentor: kordinatSkillKomunikasiBaikMentor,
pengawasanPenuh: kordinatPengawasanPenuhMentor,
pembawaanMateriBelajarMentorMengemudi: kordinatPembawaanMateriBelajarMentorMengemudi,
garisPerbatasan: garisPerbatasan
}
}

export async function tambahDataSiswa(prevState: tambahDataSiswaInterface, formData: FormData): Promise<tambahDataSiswaInterface>{
const siswa = {
nama : formData.get("nama-siswa"),
gender : formData.get("gender"),
number : formData.get("phoneNumber"),
}
try{
const dataSiswa = tambahDataSiswaSchema.safeParse({
nama: siswa.nama as string,
gender: siswa.gender as string,
phoneNumber: siswa.number as string
})

if(dataSiswa.success === false){
return {
success: false,
errors: dataSiswa.error.flatten().fieldErrors
}
}

await prisma.siswaKursusMengemudi.create({
data: {
name: dataSiswa.data.nama,
gender: dataSiswa.data.gender as Gender,
phoneNumber: String(dataSiswa.data.phoneNumber),
status: false,
}
})
return {
success: true,
data : {
nama: siswa.nama as string,
gender: siswa.gender as string,
phoneNumber: Number(siswa.number)
}
}
}catch(error){
if(error instanceof Error){
return {
success: false,
error: error.message
}
}else{
```

```
return {
success: false,
error: "Terjadi Kesalahan"
}
}
}finally{
revalidatePath('/admin')
}
}

export async function
updateDataSiswa( prevState:
tambahDataSiswaInterface, formData:
FormData):
Promise<tambahDataSiswaInterface> {
const siswa = {
id : formData.get("id"),
nama : formData.get("nama-siswa"),
gender : formData.get("gender"),
number : formData.get("phoneNumber"),
}
try{
const dataSiswa =
updateDataSiswaSchema.safeParse({
id: Number(siswa.id),
nama: siswa.nama as string,
gender: siswa.gender as string,
phoneNumber: siswa.number,
})

if(dataSiswa.success === false){
return {
success: false,
errors: dataSiswa.error.flatten().fieldErrors
}
}

await
prisma.siswaKursusMengemudi.update({
where:{
id: dataSiswa.data.id
},
data: {
name: dataSiswa.data.nama,
gender: dataSiswa.data.gender as Gender,
phoneNumber:
String(dataSiswa.data.phoneNumber),
status: false,
}
})

return {
success: true,
data : {
nama: siswa.nama as string,
gender: siswa.gender as string,
phoneNumber: Number(siswa.number)
}
}
}catch(error){
if(error instanceof Error){
return {
success: false,
error: error.message
}
}else{
return {
success: false,
error: "Terjadi Kesalahan"
}
}
}finally{
revalidatePath('/admin')
}
}

export async function deleteSiswa(id:
number){
try {
const data = await
prisma.siswaKursusMengemudi.delete({
where: {
id: id
}
})
return {
success: true
}
} catch (error) {
if(error instanceof Error){
console.error(error)
return {
success: false
}
}else{
return{
success: false
}
}
}finally{
revalidatePath('/admin')
}
```

```
}

export async function getDataSiswa(page:
number){
try {
const totalData = await
prisma.siswaKursusMengemudi.count()
const take = 10
const totalPage = Math.ceil(totalData/take)
if(page < 1){
throw new Error("Halaman tidak diketahui")
}
if(page > totalPage){
throw new Error("Halaman tidak diketahui")
}
const skip = (page - 1)*take;
const datas = await
prisma.siswaKursusMengemudi.findMany({
take: take,
skip: skip
})
return {success: true, data: datas, totalSiswa:
totalData,totalPages : totalPage}
} catch (error) {
if(error instanceof Error){
return {success: false, message : error.message}
}
}
}

export async function
getDataStudentsFilter(query: string, page:
number){

// logic data skip result 0, 5, 10, 15, ...
let skipData;
if(page >= 1){
skipData = (page - 1) * 10;
}

// logic if input search is empty
if(!query){
const dataCount = await
prisma.siswaKursusMengemudi.count()
const getStudents = await
prisma.siswaKursusMengemudi.findMany({
skip: skipData,
take: 10
})
if(!getStudents){
return {
success: false
}
}

return{
success: true,
data: getStudents,
totalSiswa: dataCount,
totalPages : Math.ceil(dataCount/10)
}
}

//logic if query params isn't empty
let dataCount;
if(query){
dataCount = await
prisma.siswaKursusMengemudi.count({
where : {
name : {
contains : query,
mode: "insensitive"
}
}
})}

const getStudentsQuery = await
prisma.siswaKursusMengemudi.findMany({
where: {
name: {
contains: query,
mode: "insensitive"
}
},
skip: skipData,
take: 10
})

if(!getStudentsQuery){
return {
success: false
}
}

if(dataCount){
return{
success: true,
data: getStudentsQuery,
totalSiswa: dataCount,
totalPages : Math.ceil(dataCount/10)
}
}else{
```

```
return{
success: true,
data: getStudentsQuery,
totalSiswa: dataCount,
totalPages: 0
}
}
}

export async function
getUserFromDb(username: string, password:
string): Promise<Login>{
try {
const [result] = await
prisma.admin.findMany({
where: {
username: username,
password: password
},
})

if(!result){
throw Error('Username dan password anda
salah')
}
return {
success: true,
data: {
username: result.username,
password: result.password
}
};
} catch (error) {
if(error instanceof Error){
return {
success: false,
error: error.message
};
} else {
return {
success: false,
error: 'An unknown error occurred',
};
}
}
}

export async function
checkPhoneNumberSignIn(prevState:
CheckNumberPhone, formData: FormData) :
Promise<CheckNumberPhone>{
const phoneNumber = await
Number(formData.get("phone-number"));
const resultCheckPhoneNumberZod =
checkPhoneNumberZod.safeParse({
phoneNumber: phoneNumber
})
if(resultCheckPhoneNumberZod.success ===
false){
return {
success: false,
errorMessage:
resultCheckPhoneNumberZod?.error.flatten().
fieldErrors.phoneNumber
}
}
const addZero =
`${0}${resultCheckPhoneNumberZod.data.ph
oneNumber}`;
const resultCheckDataByPrisma = await
prisma.siswaKursusMengemudi.findFirst({
where: {
phoneNumber: addZero,
status: false
}
})

if(!resultCheckDataByPrisma){
return {
success: false,
errorMessage: "Sepertinya nomor anda tidak
terdaftar atau sudah pernah digunakan untuk
mengisi quesioner, silahkan hubungi admin!"
}
}

return {
success: true,
id: resultCheckDataByPrisma.id,
name: resultCheckDataByPrisma.name,
phoneNumber:
String(resultCheckDataByPrisma.phoneNumb
er)
}
}

export async function
checkPhoneNumberInQuesionerPage(id:
number){
try{
const result = await
prisma.siswaKursusMengemudi.findFirst({
```

```
where: {
id: id
}
})
return result
}catch(error){
if(error instanceof Error){
console.log(error)
}
}
}

export async function loginAdmin(username:
string, password: string): Promise<Login>{
const hasil = await prisma.admin.findFirst({
where: {
username: username,
password: password
}
})
if(!hasil){
return {
success: false,
}
}else{
try {
return {
success:true
}
} catch (error) {
if (error instanceof Error) {
return {
success: false,
error: error.message
}
}
return {
error: "An unknown error occurred"
}
}
}}

export const checkDataInputanAdmin = async
(username: string, password: string) => {
const dataDariDatabase = await
prisma.admin.findMany()

if(dataDariDatabase[0]?.username !==
username){
return {
valid: false,
message: "Username anda salah"
}
}

const match = await bcrypt.compare(password,
dataDariDatabase[0]?.password);
if (!match) {
return {
valid: false,
message: "Password anda salah"
}
}else{
await prisma.admin.update({
where : {
id: 1
},
data:{
status: true
}
})
return {
valid: true,
username: dataDariDatabase[0].username
}
}
}

export async function
checkStatusAdminServer(username: string){
const findAdmin = await
prisma.admin.findFirst({
where:{
username: username,
status: true
}
})

if(findAdmin){
return {active: true}
}else{
return{active: false}
}
}

export async function signOutdmin(username:
string){
const findAdmin = await
prisma.admin.findFirst({
where:{
username: username,
status: true
```

```
}
})

if(findAdmin){
const changeStatus = await
prisma.admin.update({
where:{
id: 1
},
data: {
status: false
}
})
}else{
return{active: false}
}
}

// bagian belajar

export async function
tambahSchedule(prevState: any, formData:
FormData):Promise<TambahScheduleInterfac
e>{
const tangkapSchedule = {
nama: formData.get("schedule"),
waktu: formData.get("time"),
tanggal: formData.get("date"),
}
try {
await prisma.products.create({
data:{
nama: tangkapSchedule.nama as string,
price: tangkapSchedule.waktu as string,
image: tangkapSchedule.tanggal as string
}
})
return{
success: true,
errorMessage: null
}
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
}
}
}finally{
revalidatePath("/todoList")
redirect("/todoList")
}
}

export async function ubahSchedule(prevState:
any, formData:
FormData):Promise<TambahScheduleInterfac
e>{
const tangkapSchedule = {
nama: formData.get("name"),
waktu: formData.get("time"),
tanggal: formData.get("date"),
id: Number(formData.get("id")),
}
console.log(tangkapSchedule);
try {
await prisma.products.update({
where: {
id: tangkapSchedule.id
},
data:{
nama: tangkapSchedule.nama as string,
price: tangkapSchedule.waktu as string,
image: tangkapSchedule.tanggal as string
}
})
return{
success: true,
errorMessage: null
}
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
}
}
}finally{
revalidatePath("/todoList")
redirect("/todoList")
```

```
}
}

export async function
getSchedule():Promise<TaskFormat>{
try {
const datas = await
prisma.products.findMany()
return {
data:datas,
success: true
}
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
}
}
}
}

export async function getTaskById(idTask:
number){
try {
const data = await
prisma.products.findUnique({
where:{
id: idTask
}
})
return {
data: data,
success: true
}
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
}
}
}
}

export async function deleteSchedule(id:
number){
try {
await prisma.products.delete({
where:{
id:id
}
})
revalidatePath("/todoList")
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
}
}
}
}

export async function deleteScheduleFinish(id:
number){
try {
await prisma.taskFinish.delete({
where:{
id:id
}
})
revalidatePath("/todoList")
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
```

```
}
}
}
}

export async function taskFinish(id: number,
name:string, timeFinish:string){
try {
console.log(id, name, timeFinish);
await prisma.products.delete({
where:{
id:id
}
})

await prisma.taskFinish.create({
data:{
nama:name,
timeFinish: timeFinish
}
})
revalidatePath("/todoList")
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
}
}
}
}

export async function getTaskFinish(){
try {
const taskFinish = await
prisma.taskFinish.findMany();
return{
data:taskFinish,
success: true
}
} catch (error) {
if(error instanceof Error){
console.error(error);
return {
success: false,
errorMessage: error.message
}
}else{
return{
success: false,
errorMessage: "Error tidak diketahui"
}
}
}
}

export async function getKepentinganData(){
try {
const [arr1, arr2, arr3, arr4, arr5, arr6, arr7,
arr8, arr9, arr10] = await Promise.all([
prisma.kebersihan_mobil.findMany({
select: {
kepentingan: true,
},
}),
prisma.kelengkapan_alat_mobil.findMany({
select: {
kepentingan: true,
},
}),
prisma.performa_mobil.findMany({
select: {
kepentingan: true,
},
}),

prisma.etika_SopanSantun.findMany({
select: {
kepentingan: true,
},
}),
prisma.skill_komunikasi_baik_staff.findMany
({
select: {
kepentingan: true,
},
}),

prisma.pelayanan_administrasi.findMany({
select: {
kepentingan: true,
},
}),
prisma.etika_sopan_santun_mentor.findMany(
{
select: {
```

```
kepentingan: true,
},
}),
prisma.skill_komunikasi_baik_staff.findMany
({
select: {
kepentingan: true,
},
}),
prisma.pengawasan_penuh.findMany({
select: {
kepentingan: true,
},
}),
prisma.pembawaan_materi_belajar_mentor.fin
dMany({
select: {
kepentingan: true,
},
}),
]);
const data1 = arr1.map((item) =>
item.kepentingan);
const data2 = arr2.map((item) =>
item.kepentingan);
const data3 = arr3.map((item) =>
item.kepentingan);
const data4 = arr4.map((item) =>
item.kepentingan);
const data5 = arr5.map((item) =>
item.kepentingan);
const data6 = arr6.map((item) =>
item.kepentingan);
const data7 = arr7.map((item) =>
item.kepentingan);
const data8 = arr8.map((item) =>
item.kepentingan);
const data9 = arr9.map((item) =>
item.kepentingan);
const data10 = arr10.map((item) =>
item.kepentingan);
// Menggabungkan hasil ke dalam format yang
diinginkan
const result = data1.map((kepentingan1, index)
=> ({
kepentingan1,
kepentingan2: data2[index],
kepentingan3: data3[index],
kepentingan4: data4[index],
kepentingan5: data5[index],
kepentingan6: data6[index],
kepentingan7: data7[index],
kepentingan8: data8[index],
kepentingan9: data9[index],
kepentingan10: data10[index],
}));
return result;
} catch (error) {
console.error(error);
}
}

export async function getKinerjaData(){
try {
const [arr1, arr2, arr3, arr4, arr5, arr6, arr7,
arr8, arr9, arr10] = await Promise.all([
prisma.kebersihan_mobil.findMany({
select: {
kinerja: true,
},
}),
prisma.kelengkapan_alat_mobil.findMany({
select: {
kinerja: true,
},
}),
prisma.performa_mobil.findMany({
select: {
kinerja: true,
},
}),

prisma.etika_SopanSantun.findMany({
select: {
kinerja: true,
},
}),
prisma.skill_komunikasi_baik_staff.findMany
({
select: {
kinerja: true,
},
}),

prisma.pelayanan_administrasi.findMany({
select: {
kinerja: true,
},
}),
prisma.etika_sopan_santun_mentor.findMany(
{
select: {
```

```
kinerja: true,
},
}),
prisma.skill_komunikasi_baik_staff.findMany
({
select: {
kinerja: true,
},
}),
prisma.pengawasan_penuh.findMany({
select: {
kinerja: true,
},
}),
prisma.pembawaan_materi_belajar_mentor.fin
dMany({
select: {
kinerja: true,
},
}),
]);
const data1 = arr1.map((item) => item.kinerja);
const data2 = arr2.map((item) => item.kinerja);
const data3 = arr3.map((item) => item.kinerja);
const data4 = arr4.map((item) => item.kinerja);
const data5 = arr5.map((item) => item.kinerja);
const data6 = arr6.map((item) => item.kinerja);
const data7 = arr7.map((item) => item.kinerja);
const data8 = arr8.map((item) => item.kinerja);
const data9 = arr9.map((item) => item.kinerja);
const data10 = arr10.map((item) =>
item.kinerja);
// Menggabungkan hasil ke dalam format yang
diinginkan
const result = data1.map((kinerja1, index) =>
({
kinerja1,
kinerja2: data2[index],
kinerja3: data3[index],
kinerja4: data4[index],
kinerja5: data5[index],
kinerja6: data6[index],
kinerja7: data7[index],
kinerja8: data8[index],
kinerja9: data9[index],
kinerja10: data10[index],
}));
return result;
} catch (error) {
console.error(error);
}

export async function getData(){
try {
let data = await
prisma.kebersihan_mobil.findMany({
select:{
kepentingan:true,
kinerja:true
}
})
return data;
} catch (error) {
console.error(error);
}
}
interface DataTingkatKesesuaian{
name : string;
process: string;
result: number;
}
interface Formatkopi{
skor_keputusan: number,
tingkat_kesesuaian: DataTingkatKesesuaian[];
}
export async function getTingkatKesesuaian():
Promise<Formatkopi | null>{
try{
const getAllData = await Promise.all([
prisma.kebersihan_mobil.findMany({
select: {
kepentingan: true,
kinerja: true,
},
}),
prisma.kelengkapan_alat_mobil.findMany({
select: {
kepentingan: true,
kinerja: true,
},
}),
prisma.performa_mobil.findMany({
select: {
kepentingan: true,
kinerja: true,
},
}),

prisma.etika_SopanSantun.findMany({
select: {
kepentingan: true,
```

```
kinerja: true,
},
}),
prisma.skill_komunikasi_baik_staff.findMany
({
select: {
kepentingan: true,
kinerja: true,
},
}),

prisma.pelayanan_administrasi.findMany({
select: {
kepentingan: true,
kinerja: true,
},
}),
prisma.etika_sopan_santun_mentor.findMany(
{
select: {
kepentingan: true,
kinerja: true,
},
}),
prisma.skill_komunikasi_baik.findMany({
select: {
kepentingan: true,
kinerja: true,
},
}),
prisma.pengawasan_penuh.findMany({
select: {
kepentingan: true,
kinerja: true,
},
}),
prisma.pembawaan_materi_belajar_mentor.fin
dMany({
select: {
kepentingan: true,
kinerja: true,
},
}),
])

let arrDataTingkatKesesuaian :
DataTingkatKesesuaian[] = [];
let arrTingkatKesesuaian = [];
const arrServicesName = ["Kebersihan mobil",
"Kelengkapan & performa alat mobil",
"Performa Mobil", "Etika & sopan santun
karyawan", "Skill komunikasi karyawan",
"Pelayanan informasi & jadwal belajar",
"Etika & sopan santun mentor", "Skill
komunikasi mentor", "Pegawasan penuh
mentor", "Kompetensi mentor"];
getAllData.map((itemNilai, index)=>{
// hp = Hasil penjumlahan nilai
const hpnKepentingan =
itemNilai.reduce((sum, item)=> sum +
item.kepentingan, 0);
const hpnKinerja = itemNilai.reduce((sum,
item)=> sum + item.kinerja, 0);
const hpTingkatKesesuaian =
((hpnKinerja/hpnKepentingan)*100)+0.00033
33333;
arrTingkatKesesuaian.push(hpTingkatKesesua
ian);
const formatDataReturn = {
name : arrServicesName[index],
process: `(${hpnKinerja} /
${hpnKepentingan}) x ${100}% =
${hpTingkatKesesuaian}`,
result:
Number(hpTingkatKesesuaian.toFixed(2))
}
arrDataTingkatKesesuaian.push(formatDataRe
turn);
})
const skorKeputusan =
arrDataTingkatKesesuaian.reduce((sum,
item)=> sum + item.result,
0)/arrDataTingkatKesesuaian.length;

const result:Formatkopi = {
skor_keputusan:
Number(skorKeputusan.toFixed(2)),
tingkat_kesesuaian: arrDataTingkatKesesuaian
}
return result;
}catch(error){
if(error instanceof Error){
console.error(error);
throw error;
}
return null;
}
}

export async function ujiValiditas(jenis:
string){
try{
```

```
if(jenis === "kepentingan"){
const getAllData = await Promise.all([
prisma.kebersihan_mobil.findMany({
select: {
kepentingan: true,
},
}),
prisma.kelengkapan_alat_mobil.findMany({
select: {
kepentingan: true,
},
}),
// prisma.performa_mobil.findMany({
// select: {
// kepentingan: true,
// },
// }),
// prisma.etika_SopanSantun.findMany({
// select: {
// kepentingan: true,
// },
// }),
//
prisma.skill_komunikasi_baik_staff.findMany
({
// select: {
// kepentingan: true,
// },
// }),
// prisma.pelayanan_administrasi.findMany({
// select: {
// kepentingan: true,
// },
// }),
//
prisma.etika_sopan_santun_mentor.findMany(
{
// select: {
// kepentingan: true,
// },
// }),
// prisma.skill_komunikasi_baik.findMany({
// select: {
// kepentingan: true,
// },
// }),
// prisma.pengawasan_penuh.findMany({
// select: {
// kepentingan: true,
// },
// }),
//
prisma.pembawaan_materi_belajar_mentor.fin
dMany({
// select: {
// kepentingan: true,
// },
// }),
])
// let kuefisienValues = [];
// for (let k = 0; k < getAllData.length; k++) {
// let total_x = getAllData[k].reduce((acc, val)
=> acc + val.kepentingan, 0);
// let y = [];
// let total_y = 0;
// for (let i = 0; i < getAllData[k].length; i++)
{
// const sum = getAllData[i].reduce((acc, val)
=> acc + val.kepentingan, 0);
// y.push(sum);
// total_y += y[i];
// }
// let xy = [];
// let total_xy = 0;
// for (let i:number = 0; i <=
getAllData[k].length; i++) {
// xy.push(getAllData[i][i].kepentingan * y[i]);
// total_xy += xy[i];
// }
// let x2 = [];
// let total_x2 = 0;
// for (let i = 0; i <= getAllData[k].length; i++)
{
// x2.push(getAllData[i][i].kepentingan ** 2);
// total_x2 += x2[i];
// }
// let y2 = [];
// let total_y2 = 0;
// for (let i = 0; i <= getAllData[k].length; i++)
{
// y2.push(y[i] ** 2);
// total_y2 += y2[i];
// }
// const perhitunganAtas =
getAllData[k].length * total_xy - total_x *
total_y;
// const perhitunganBawahKiri =
getAllData[k].length * total_x2 - total_x ** 2;
// const perhitunganBawahKanan =
getAllData[k].length * total_y2 - total_y ** 2;
// const nilaiKuadrat = Math.sqrt(
```

```
// perhitunganBawahKiri *
perhitunganBawahKanan
// );
// const nilaiKoefisien = perhitunganAtas /
nilaiKuadrat;
//
kuefisienValues.push(Number(nilaiKoefisien.t
oFixed(3)));
// }
// return kuefisienValues;
let data: {x1: number[],x2: number[],x3:
number[],x4: number[],x5: number[],x6:
number[],x7: number[],x8: number[],x9:
number[],x10: number[]} = {
x1: [],
x2: [],
x3: [],
x4: [],
x5: [],
x6: [],
x7: [],
x8: [],
x9: [],
x10: [],
};
const keys = Object.values(data);
for(let k = 0; k<getAllData.length;k++){
for(let i = 0; i<171;i++){
keys[k].push(getAllData[k][i].kepentingan)
}
}
return data;
}
if(jenis === "kinerja"){
const getAllData = await Promise.all([
prisma.kebersihan_mobil.findMany({
select: {
kinerja: true,
},
}),
prisma.kelengkapan_alat_mobil.findMany({
select: {
kinerja: true,
},
}),
prisma.performa_mobil.findMany({
select: {
kinerja: true,
},
}),
prisma.etika_SopanSantun.findMany({
select: {
kinerja: true,
},
}),
prisma.skill_komunikasi_baik_staff.findMany
({
select: {
kinerja: true,
},
}),

prisma.pelayanan_administrasi.findMany({
select: {
kinerja: true,
},
}),
prisma.etika_sopan_santun_mentor.findMany(
{
select: {
kinerja: true,
},
}),
prisma.skill_komunikasi_baik.findMany({
select: {
kinerja: true,
},
}),
prisma.pengawasan_penuh.findMany({
select: {
kinerja: true,
},
}),
prisma.pembawaan_materi_belajar_mentor.fin
dMany({
select: {
kinerja: true,
},
}),
])
}
}catch(error){
if(error instanceof Error){
console.error(error);
throw error;
}
// return null;
}
}
```

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : MARWAN	Pembimbing I : Wahyuddin, S. Kom, M. Kom.
NIM : 220280157	Pembimbing II : Mughaffir Yunus, ST, MT.
Judul Skripsi : IMPLEMENTASI METODE IMPORTANCE PERFORMACE ANALYSIS DALAM MENGANALISA KEBUTUHAN DAN KEPUASAN TERHADAP FASILITAS DAN PELAYANAN SISWA DI LESTARI BELAJAR	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 - jabarkan kuisioner yg akan digunakan pada Aplikasi		Konsultasi 1 - Latar Belakang	
Konsultasi 2 - Masukkan rumus IPA pada BAB II		Konsultasi 2 - User interface (perbaiki lagi)	
Konsultasi 3 perjelas teori yg digunakan		Konsultasi 3 - Rancangan aplikasi diperbaiki (jangan desain UI)	
Konsultasi 4		Konsultasi 4 - perbaiki penomoran sub bab - Perbaiki rumusan masalah	
Konsultasi 5		Konsultasi 5 Acc Seminar proposal	

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

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

SKR USPS1
~~HASIL~~

NAMA	: MARWAN	Pembimbing I : Wahyudin, S.Kom, M.Kom.
NIM	: 220 280 157	Pembimbing II : Mughaffir Yunus, ST, MT.
Judul Skripsi	: IMPLEMENTASI METODE IMPORTANCE PERFORMANCE ANALYSIS DALAM MENGANALISIS KEPUASAN SISWA TERHADAP KUALITAS PELAYANAN DI LESTARI BELAJAR BERBASIS WEB	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 - Ikuti panduan - Kata asing digunakan		Konsultasi 1 perbaiki sesuai pada penyampaian	
Konsultasi 2 - Abstrak Indonesia - " inggris		Konsultasi 2 Gambar gunakan pada web sesuai dengan yang ada dilembaga	
Konsultasi 3 ACC ujian hasil		Konsultasi 3 Tambahkan referensi terkait penelitian	
Konsultasi 4		Konsultasi 4 Acc Aplikasi Lanjutkan buat dokumen	
Konsultasi 5		Konsultasi 5 Acc Seminar Hasil	

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

Lanjutan...

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6		Konsultasi 6 - Perbaiki dalam penulisan - Tambahkan keterangan pada pada uraian pembahasan	
Konsultasi 7		Konsultasi 7 Acc Ujian Tutup	
Konsultasi 8		Konsultasi 8	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Parepare, 2024

Mengetahui
Ketua Program Studi

Marlina, S.Kom.,M.Kom.
NBM. 1162 680

Mahasiswa

MARWAN
NIM. 220 280 157

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

SERTIFIKAT

Nomor : 409/PRK/LB-TI/II.3.AU/D/2023

Diberikan Kepada :

Mahasiswa

NIM. 2202801571

Telah Menyelesaikan

PRAKTIKUM III

Laboratorium Teknik Informatika

Program Studi Teknik Informatika Fakultas Teknik

Universitas Muhammadiyah Prepare

Pada tanggal 16 Januari 2023 - 29 Maret 2023

Prepare, 15 Juni 2023

KEPALA
LABORATORIUM

Ir. Untung Suwardoyo, S.Kom., MT., IPP.
NBM. 1288 973

KETUA PRODI
TEKNIK INFORMATIKA

Marlina, S.Kom., M.Kom
NBM. 1162 680

Nilai Praktikum laboratorium Teknik Informatika
Fakultas Teknik Universitas Muhammadiyah Parepare Tahun 2023

MATERI	NILAI ANGKA
PEMROGRAMAN WEB 2	82,5
PEMROGRAMAN JAVA	59,5
PEMROGRAMAN DELPHI 10	56
JARINGAN KOMPUTER 2	85
RATA-RATA	70,75
NILAI HURUF	B

SERTIFIKAT

Nomor : 193/PRK/LB-TI/II.3.AU/D/2022

diberikan kepada :

MARWAN

NIM. 2202801571

telah menyelesaikan

PRAKTIKUM IV

Laboratorium Teknik Informatika
Program Studi Teknik Informatika Fakultas Teknik
Universitas Muhammadiyah Pareparе

pada tanggal 14 September 2022 - 25 Desember 2022

Parepare, 02 Januari 2023

KEPALA
LABORATORIUM

Ir. UNTUNG SUWARDOYO, S.Kom., MT.,IPP.

KETUA PRODI
TEKNIK INFORMATIKA

MARLINA. S.Kom., M.Kom

Nilai Praktikum laboratorium Teknik Informatika
Fakultas Teknik Universitas Muhammadiyah Parepare Tahun 2022

MATERI	NILAI ANGKA
DESAIN 3D	80
UI/UX DESIGN	80
PEMROGRAMAN PYTHON	95
MATEMATIKA INFORMATIKA	70
RATA-RATA	81,25
NILAI HURUF	A