LAMPIRAN

Lampiran 1 Surat Kesediaan Membimbing TA Pembimbing 1

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini :

Nama	: Ida Afriliana, ST, M.Kom
NIDN	: 0624047703
NIPY	: 12.013.168
Jabatan Struktural	: Ka.Prodi D3 Teknik Komputer
Jabatan Fungsional	: Lektor

Dengan ini menyatakan bersedia untuk menjadi pembimbing I pada Tugas Akhir mahasiswa berikut :

Nama	: Sabina Izatul Zahrani
NIM	: 21040109
Program Studi	: DIII Teknik Komputer

Judul TA : SISTEM INFORMASI TOKO ROTI BERBASIS *WEBSITE*

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 10 Juni 2024

Mengetahui
Ka. Prodi DIII Teknik Komputer,

Ida Afriliana.ST.M.Kom
NIPY. 12.013.168

Dosen Pembimbing I,

Ida Afriliana.ST.M.Kom
NIPY. 12.013.168

Lampiran 2 Surat Kesediaan Membimbing TA Pembimbing 2

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini :

Nama	: Abdul Basit, S.Kom,. MT
NIDN	: 0608129106
NIPY	: 01.015.198
Jabatan Struktural	: Sekretaris Prodi D3 Teknik Komputer
Jabatan Fungsional	: Asisten Ahli

Dengan ini menyatakan bersedia untuk menjadi pembimbing II pada Tugas Akhir mahasiswa berikut :

Nama	: Sabina Izatul Zahrani
NIM	: 21040109
Program Studi	: DIII Teknik Komputer

Judul TA : SISTEM INFORMASI TOKO ROTI BERBASIS *WEBSITE*

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 21 Juni 2024

Mengetahui
Ka. Prodi DIII Teknik Komputer,

Ida Afriliana.ST.M.Kom
NIPY. 12.013.168

Dosen Pembimbing II,

Abdul Basit, S.Kom,. MT
NIPY. 01.015.198

Lampiran 3 Surat Observasi

POLITEKNIK HARAPAN BERSAMA

D-3 Teknik Komputer

No. : 013.03/KMP.PHB/VII/2024
Lampiran : -
Perihal : Permohonan Izin Observasi Tugas Akhir (TA)

Kepada Yth.
Pimpinan Qiana Pastry Brebes
Jl. Let. Jend. Sutoyo No.9-13, Brebes Tengah, Brebes, Kec. Brebes, Kabupaten Brebes, Jawa Tengah 52212

Dengan Hormat,
Sehubungan dengan tugas mata kuliah Tugas Akhir (TA) yang akan diselenggarakan di semester VI (Genap) Program Studi D III Teknik Komputer Politeknik Harapan Bersama Tegal, Maka dengan ini kami mengajukan izin observasi pengambilan data di Qiana Pastry Brebes yang Bapak / Ibu Pimpin, untuk kepentingan dalam pembuatan produk Tugas Akhir, dengan Mahasiswa sebagai berikut:

No.	NIM	Nama	No. HP
1	21040109	SABINA IZATUL ZAHRANI	0895391403539

Demikian surat permohonan ini kami sampaikan atas izin dan kerjasamanya kami sampaikan terima kasih.

Tegal, 08 Juli 2024
Ka. Prodi DIII Teknik Komputer
Politeknik Harapan Bersama Tegal

Ida Afriliana, ST, M.Kom
NIPY. 12.013.168

Jl. Mataram No. 9 Kota Tegal 52143, Jawa Tengah, Indonesia.
(0283)352000
komputer@poltektegal.ac.id
poltektegal.ac.id

Lampiran 4 Wawancara Observasi

1. Apakah toko roti ini sudah memiliki sistem untuk mengelola toko roti ini?

   Narasumber : toko ini masih belum memiliki sistem pengelolaan.

2. Bagaimana pembeli memesan roti disini? Apakah melalui *website*, datang langsung ke toko, atau bisa lewat *chat whatsapp*?

   Narasumber : toko ini masih belum memiliki *website*, jadi pembeli harus datang ke toko untuk memesan atau bisa memesan lewat *whatsapp*.

3. Jika ada pembeli yang pesanannya banyak itu bagaimana pencatatannya?

   Narasumber : jika ada yang pesan akan dicatat di papan.

4. Untuk pembayarannya, apakah hanya bisa *cash* atau bisa bayar *online*?

   Narasumber : pembayaran bisa dilakukan secara *cash* maupun *online* pakai QRIS.

5. Untuk laporan penjualannya apakah masih dicatat secara manual atau sudah pakai sistem?

   Narasumber : laporan penjualan pakai aplikasi dan dicatat di buku juga, kemudian diserahkan langsung kerumah atasan.

6. Untuk pengelolaan data produk itu apakah dilakukan secara manual atau menggunakan sistem?

   Narasumber : pengelolaan data produk masih manual, biasanya dicatat di buku mengenai produknya apa saja dan stoknya berapa.

Lampiran 5 Source Code

```
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use App\Models\Product;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\Validator;

class ProductController extends Controller
{
    public function index()
    {
        $products = Product::all();
        foreach ($products as $product) {
            $product->image_url = url('uploads/images/'
. $product->image);
        }
        return response()->json(['status' => 'success',
'data' => $products]);    }

    public function store(Request $request)
{
    $validator = Validator::make($request->all(), [
        'name' => 'required|string',
        'price' => 'required|numeric',
        'image' =>
'required|image|mimes:jpeg,png,jpg,gif|max:2048',
```

```
            'stock' => 'required|integer',
            'weight' => 'required|numeric',
            'description' => 'required|string',
            'category_id' =>
'required|exists:categories,id',
        ]);

        if ($validator->fails()) {
            return response()->json(['status' => 'error',
'message' => $validator->errors()], 400);
        }

        // Menyimpan gambar baru dalam variabel
        $imageName = time() . '.' . $request->image-
>getClientOriginalExtension();
        $request->image-
>move(public_path('uploads/images'), $imageName);

        // Membuat produk baru dengan data yang diterima
dan menyimpan nama gambar baru
        $product = Product::create(array_merge($request-
>except('image'), ['image' => $imageName]));

        // Tambahkan URL lengkap untuk gambar produk
        $product->image_url = url('uploads/images/' .
$product->image);

        return response()->json(['status' => 'success',
'data' => $product], 201);
    }

        public function show($id)
        {
            $product = Product::findOrFail($id);
            $product->image_url = url('uploads/images/' .
$product->image);
            return response()->json(['status' => 'success',
'data' => $product]);
        }

        public function update(Request $request, $id)
    {
        $product = Product::findOrFail($id);

        $validator = Validator::make($request->all(), [
            'name' => 'required|string',
            'price' => 'required|numeric',
```

```
        'image' =>
'nullable|image|mimes:jpeg,png,jpg,gif|max:2048', //
Pastikan validasi gambar ditambahkan
        'stock' => 'required|integer',
        'weight' => 'required|numeric',
        'description' => 'required|string',
        'category_id' =>
'required|exists:categories,id',
    ]);

    if ($validator->fails()) {
        return response()->json(['status' => 'error',
'message' => $validator->errors()], 400);
    }

    // Hapus gambar lama jika ada gambar baru yang
diunggah
    if ($request->hasFile('image')) {
        // Hapus gambar lama jika ada
        if ($product->image) {
            Storage::delete('public/uploads/images/' .
$product->image);
        }
        // Simpan gambar baru dari variabel yang sudah
ada
        $imageName = time() . '.' . $request->image-
>getClientOriginalExtension();
        $request->image-
>move(public_path('uploads/images'), $imageName);
        $product->image = $imageName;
    }

    $product->update($request->except('image')); //
Menyimpan data kecuali gambar

    // Tambahkan URL lengkap untuk gambar produk
    $product->image_url = url('uploads/images/' .
$product->image);

    return response()->json(['status' => 'success',
'data' => $product]);
}

    public function destroy($id)
    {
        $product = Product::findOrFail($id);
        $product->delete();
```

```
        return response()->json(['status' => 'success',
'message' => 'Product deleted successfully']);
    }
}

import { useEffect, useState } from 'react';
import { HiMinusSm, HiPlusSm } from "react-icons/hi";
import { useNavigate, useParams } from 'react-router-
dom';
import { useDispatch, useSelector } from 'react-redux';
import { fetchCategories, fetchProducts } from
'../redux/slices/productUserSlice';
import { toast } from 'react-toastify';
import { addToCart } from '../redux/slices/cartSlice';

const ProductDetail = () => {
    const { id } = useParams();
    const navigate = useNavigate();
    const dispatch = useDispatch();
    const { token, user } = useSelector((state) =>
state.auth);
    const products = useSelector((state) =>
state.productUser.products);
    const [product, setProduct] = useState(null);
    const [quantity, setQuantity] = useState(1);

    useEffect(() => {
        dispatch(fetchProducts());
        dispatch(fetchCategories());
    }, [dispatch]);

    useEffect(() => {
        const selectedProduct = products.find(p => p.id
=== parseInt(id));
        setProduct(selectedProduct);
    }, [id, products]);

    const decreaseQuantity = () => {
        if (quantity > 1) {
            setQuantity(quantity - 1);
        }
    };

    const increaseQuantity = () => {
        setQuantity(quantity + 1);
    };
```

```
    const handleAddToCart = () => {
        if (!token) {
            toast.error('Silakan login terlebih
dahulu.');
            navigate('/login');
            return;
        }

        if (user.role === 'admin') {
            toast.error('Admin tidak dapat menambahkan
produk ke keranjang.');
            return;
        }

        if (product.stock <= 0) {
            toast.error('Produk ini sudah habis.');
            return;
        }

        dispatch(addToCart({ product_id: product.id,
qty: quantity }))
            .then(() => {
                toast.success('Produk berhasil
ditambahkan ke keranjang!');
            })
            .catch(() => {
                toast.error('Terjadi kesalahan. Silakan
coba lagi.');
            });
    };

    if (!product) {
        return <p>Loading...</p>;
    }

    return (
        <div className="p-4 pt-20">
            <div className="flex flex-col md:flex-row
items-center">
                <div className="md:w-1/2 mb-4 md:mb-0">
                    <img src={product.image_url}
alt={product.name} className="w-full h-96 md:h-full
object-cover rounded-lg" />
                </div>
                <div className="md:w-1/2 md:pl-4 w-
full">
```

```
                    <h2 className="text-2xl font-bold
mb-2 md:mt-0 text-center md:text-
left">{product.name}</h2>
                    <div className="flex justify-
between">
                        <p className="text-lg font-
semibold mb-2">Rp. {product.price.toLocaleString()}</p>
                        <p className="text-lg font-
semibold mb-2">Stock: {product.stock}</p>
                    </div>
                    <p className="mb-4 text-center
md:text-left">{product.description}</p>
                    <div className="flex items-center
justify-between mb-4">
                        <button
onClick={decreaseQuantity} className="bg-[#FFE0B5] px-3
py-2 rounded-md hover:bg-[#eab162]"><HiMinusSm
size={20} /></button>
                        <p className="text-xl px-
2">{quantity}</p>
                        <button
onClick={increaseQuantity} className="bg-[#FFE0B5] px-3
py-2 rounded-md hover:bg-[#eab162]"><HiPlusSm size={20}
/></button>
                    </div>
                    <button onClick={handleAddToCart}
className="w-full bg-secondary text-white py-2 px-4
rounded-md hover:bg-hover mx-auto md:mx-0 block">Add To
Cart</button>
                </div>
            </div>
        </div>
    );
};

export default ProductDetail;

import { useState, useEffect, useRef } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import { HiFilter } from 'react-icons/hi';
import { fetchProducts, fetchCategories,
setSelectedCategory } from
'../redux/slices/productUserSlice';
import ProductCard from '../components/ProductCard';
import Pagination from '../components/Pagination';
import { useNavigate } from 'react-router-dom';
```

```
const Products = () => {
  const dispatch = useDispatch();
  const navigate = useNavigate();
  const { products, selectedCategory } =
useSelector((state) => state.productUser);
  const { categories } = useSelector((state) =>
state.productUser);

  const [productsList, setProductsList] = useState([]);
  const [isMenuOpen, setIsMenuOpen] = useState(false);
  const filterButtonRef = useRef(null);

  const [currentPage, setCurrentPage] = useState(1);
  const productsPerPage = 8;

  useEffect(() => {
    dispatch(fetchProducts());
    dispatch(fetchCategories());
    dispatch(setSelectedCategory({ id: 'All', name:
'All' }));
  }, [dispatch]);

  useEffect(() => {
    // Filter produk berdasarkan kategori yang dipilih
    if (selectedCategory.id  === 'All') {
      setProductsList(products);
    } else {
      setProductsList(products.filter((product) =>
product.category_id === selectedCategory.id));
    }
    setCurrentPage(1); // Reset halaman ke 1 saat
kategori berubah
```

```
  }, [products, selectedCategory]);

  const toggleMenu = () => {
    setIsMenuOpen(!isMenuOpen);
  };

  const handleCategoryClick = (category) => {
    dispatch(setSelectedCategory(category)); //
Dispatch action untuk mengubah selectedCategory
    setIsMenuOpen(false);
  };

  const handleProductClick = (product) => {
    navigate(`/productdetail/${product.id}`);
  };

  const indexOfLastProduct = currentPage *
productsPerPage;
  const indexOfFirstProduct = indexOfLastProduct -
productsPerPage;
  const currentProducts =
productsList.slice(indexOfFirstProduct,
indexOfLastProduct);

  const totalPages = Math.ceil(productsList.length /
productsPerPage);

  const onPageChange = (pageNumber) => {
    setCurrentPage(pageNumber);
  };

  return (
    <div className="p-4 pt-20 relative">
      <div className="pb-2 flex justify-between">
            <span className="font-semibold text-
xl">{selectedCategory.name}</span>
            <button
                className="inline-flex items-center p-2
text-sm text-gray-500 rounded-lg md:hidden hover:bg-
gray-100 focus:outline-none focus:ring-2 focus:ring-
gray-200"
                onClick={toggleMenu}
                ref={filterButtonRef}>
                <HiFilter className="h-6 w-6 cursor-
pointer" />
            </button>
        </div>
```

```
        <div className="grid grid-cols-1 sm:grid-cols-2
md:grid-cols-3 lg:grid-cols-4 gap-4">
            {currentProducts.map((product, index) => (
            <div key={index} onClick={() =>
handleProductClick(product)}>
                <ProductCard product={product} />
            </div>
            ))}
        </div>
        <Pagination currentPage={currentPage}
totalPages={totalPages} onPageChange={onPageChange} />
        {/* mobile menu */}
        {isMenuOpen && (
            <div
            className="md:hidden bg-white shadow-lg
absolute left-0 rounded-md w-full"
            style={{ top:
filterButtonRef.current?.offsetTop +
filterButtonRef.current?.offsetHeight + 'px' }}
            >
            <div className="flex flex-col items-center
p-4">
                <button
                className={`text-lg font-semibold
hover:bg-[#D8AE7E] transition duration-150 ease-in-out
py-2 w-full text-center rounded-xl ${
                    selectedCategory === 'All' ? 'bg-
[#D8AE7E]' : ''
                }`}
                onClick={() => {
                    handleCategoryClick({ id: 'All',
name: 'All' });
                    setIsMenuOpen(false);
                }}
                >
                All
                </button>
                {categories.map((category) => (
                <button
                    key={category.id}
                    className={`text-lg font-semibold
hover:bg-[#D8AE7E] transition duration-150 ease-in-out
py-2 w-full text-center rounded-xl ${
                    selectedCategory === category.name
? 'bg-[#D8AE7E]' : ''
                    }`}
                    onClick={() => {
```

```
                        handleCategoryClick(category);
                        setIsMenuOpen(false);
                        }}
                    >
                        {category.name}
                    </button>
                    ))}
                </div>
                </div>
            )}
        </div>
    );
};

export default Products;
```

Lampiran 6 Foto Dokumentasi