



เอกสารคู่มือระบบ

ระบบสั่งอาหารผ่านเว็บไซต์

(Quickbites: Online Food Ordering System)

จดทำโดย

6604062630111 นายจุ่นพลภัทร์ สาเกภูล

6604062630251 นายธีรชวนนท์ ศรีธรรมยศ

6604062630358 นายพงษ์พัฒน์ บางข่า

6604062630561 นายอชิตพล แทนโป

6604062630579 นายอธิป yawangkaew

เสนอ

ผู้ช่วยศาสตราจารย์ สุทธิ์ ประสมพันธ์

รายงานเล่มนี้เป็นส่วนหนึ่งของรายวิชา 040613306 วิศวกรรมซอฟต์แวร์

ภาควิชาวิทยาการคอมพิวเตอร์และสารสนเทศ คณะวิทยาศาสตร์ประยุกต์

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ปีการศึกษา 2/2567

สารบัญ

หน้า

Quickbites Webpage

- แกนหลักของระบบ	1
- คอมโพเนนท์	9
- หน้าเว็บ	21

Admin Dashboard

- แกนหลักของระบบ	34
- คอมโพเนนท์	43
- หน้าล็อกอิน	49
- หน้าหลัก	52
- ระบบวิเคราะห์ข้อมูลยอดขาย	57
- ระบบจัดการข้อมูลพนักงาน	64
- ระบบจัดการรายการอาหาร	76
- ระบบจัดการออร์เดอร์	92

Controller

102

Design Pattern

- Provider	118
- Observer	120

Quickbites Webpage

แกนหลักของระบบ

App.jsx

คำอธิบาย

- ควบคุมการแสดงผลของหน้าเว็บแต่ละหน้า
- การตรวจสอบสถานะของโต๊ะที่เปิดให้ใช้บริการ
- การจัดส่งทางระหว่างหน้าต่าง ๆ

โค้ดสำคัญ

ส่วนที่ 1: การกำหนด URL ให้ตรงตามหมายเลขโต๊ะที่ลูกค้าสแกน

```
useEffect(() => {
  if (tableNumber) {
    const currentParams = new URLSearchParams(location.search);
    currentParams.set('table', tableNumber);
    navigate(`?${currentParams.toString()}`, { replace: true });
  }
}, [tableNumber, location.search, navigate]);
```

ส่วนที่ 2: การตรวจสอบว่าโต๊ะในร้านอาหารลูกเบิดให้ใช้บริการหรือไม่

```
<ToastContainer position="top-right" autoClose={2000} />
{!hideNavbarRoutes.includes(location.pathname) && <Navbar />}
{loading ? (
  <div className="text-center mt-10 text-2xl font-bold text-white">
    <p>Checking table status...</p>
  </div>
) : available ?
```

- โดยถ้าต้องการเปิดให้ใช้บริการ จะคืนค่าเส้นทางของหน้าต่าง ๆ ภายในระบบให้สามารถเข้าถึงได้

```
(  

  <Routes>  

    <Route path="/" element={<Home />} />  

    <Route path="/cart" element={<Cart />} />  

    <Route path="/table/:tableNumber" element={<Table />} />  

    <Route path="/about" element={<About />} />  

    <Route path="/contact" element={<Contact />} />  

    <Route path="/orderSummary" element={<OrderSummary />} />  

    <Route path="/:food/:foodId" element={<Food />} />  

    <Route path="/ThankYou" element={<Thank />} />  

  </Routes>  

)
```

- แต่ถ้าต้องการปิดอยู่ จะแสดงหน้า “ไม่พร้อมให้บริการ” ให้แก่ผู้ใช้งาน

```
(  

  <div className="text-center mt-10 text-2xl font-bold text-red-600">  

    <p>The table is currently unavailable. Please try again later.</p>  

  </div>  

)
```

StoreContext.jsx

คำอธิบาย

ไฟล์จัดการ state หลัก เช่น ทะกร้าสินค้า รายการอาหาร คำสั่งชื่อ คำค้นหา หมายเลขโทรศัพท์ และข้อมูลผู้ใช้ รวมถึงการสื่อสารกับ API เพื่อดึงข้อมูลอาหาร และข้อมูลการใช้งานโดยทำให้คอมโภเนนท์ทั่วทั้งแอปสามารถเข้าถึงและปรับปรุงข้อมูลร่วมกันได้อย่างราบรื่น

โค้ดสำคัญ

ส่วนเสริม: การกำหนด state สำหรับข้อมูลต่าง ๆ เช่น สัญลักษณ์สกุลเงิน รายการในทะกร้า คำสั่งชื่อ คำค้นหา หมายเลขโทรศัพท์ และข้อมูลผู้ใช้ รวมถึง URL ของ API จากตัวแปรต่าง ๆ

```
const currency = "฿";
const backendURL = import.meta.env.VITE_BACKEND_URL;

const [cartItems, setCartItems] = useState({});
const [orderData, setOrderData] = useState([]);
const [totalFoodCount, setTotalFoodCount] = useState(0);
const [search, setSearch] = useState("");
const [tableNumber, setTableNumber] = useState(localStorage.getItem("tableNumber") || null);
const [userID, setUserID] = useState(localStorage.getItem("userID") || "");
const [foods_list, setFoods_list] = useState([]);

const [available, setAvailable] = useState(false);
const [loading, setLoading] = useState(true);

const [searchParams] = useSearchParams();

const socket = useState(() => io(backendURL, { transports: ['websocket', 'polling'], withCredentials: true })[0];

const storeUserID = (newUserID) => {
  if (newUserID) {
    localStorage.setItem("userID", newUserID);
  }
};

const removeUserID = () => {
  localStorage.removeItem("userID");
};
```

ส่วนที่ 1: พังก์ชันการจัดการข้อมูลเกี่ยวกับโต๊ะและผู้ใช้ โดยดึงหมายเลขโต๊ะจาก URL และตรวจสอบความพร้อมใช้งานของโต๊ะผ่านการติดต่อกับ backend

```

useEffect(() => {
  const tableFromURL = searchParams.get("table");
  if (tableFromURL && tableNumber !== tableFromURL) {
    setTableNumber(tableFromURL);
    localStorage.setItem("tableNumber", tableFromURL);
  }
}, [searchParams]);

useEffect(() => {
  if (tableNumber !== null) {
    checkTable();
  }
}, [userID, tableNumber]);

const checkTable = async () => {
  if (!tableNumber) return;

  setLoading(true);
  try {
    if (userID) {
      const response = await axios.get(`${backendURL}/api/table/get`, {params: { table: tableNumber }});

      if (!response.data.success) {
        removeUserID();
        setUserID("");
        setAvailable(false);
      } else {
        setAvailable(true);
      }
      return;
    }

    const response = await axios.post(`${backendURL}/api/table/join`, {
      table: tableNumber,
      userID: userID,
    });

    if (response.data.success) {
      setUserID(response.data.userID);
      storeUserID(response.data.userID);
      setAvailable(true);
    } else {
      setAvailable(false);
    }
  } catch (error) {
    console.error(error);
    setAvailable(false);
  } finally {
    setLoading(false);
  }
};

useEffect(() => {
  socket.on("tableUpdatedStatus", (data) => {
    if (data.table == tableNumber) {
      if (data.available) {
        if (!userID) {
          checkTable();
        } else {
          setAvailable(true);
        }
      } else {
        removeUserID();
        setUserID("");
        setAvailable(false);
      }
    }
  });
});

return () => {
  socket.off("tableUpdatedStatus");
};
}, [socket, tableNumber, userID]);

```

ส่วนที่ 2: พัฒนาการจัดการข้อมูลตระกร้าสินค้าและคำสั่งซื้อ โดยเพิ่มรายการอาหารลงในตระกร้าจากนั้นจึงจะคำนวณจำนวนและยอดรวม

```

const setToCart = (itemId, itemCount, requirement = '') => {
  let cartData = structuredClone(cartItems);
  if (itemCount === 0) {
    removeItem(itemId);
  } else {
    cartData[itemId] = { quantity: itemCount, requirement };
    setCartItems(cartData);
  }
};

const getCartCount = () => {
  return Object.values(cartItems).reduce((acc, item) => acc + item.quantity, 0);
};

const updateQuantity = (itemId, quantity) => {
  if (cartItems[itemId]) {
    setCartItems((prevCart) => ({
      ...prevCart,
      [itemId]: { ...prevCart[itemId], quantity },
    }));
  }
};

const removeItem = (id) => {
  setCartItems((prevCart) => {
    const updatedCart = { ...prevCart };
    delete updatedCart[id];
    return updatedCart;
  });
};

const getCartAmount = () => {
  return Object.entries(cartItems).reduce((totalAmount, [itemId, { quantity }]) => {
    const itemInfo = foods_list.find((food) => food._id === itemId);
    if (itemInfo) {
      totalAmount += itemInfo.price * quantity;
    }
    return totalAmount;
  }, 0);
};

const placeOrder = () => {
  const order = {
    orderId: new Date().getTime(),
    tableNumber,
    items: Object.keys(cartItems).map((itemId) => {
      const itemInfo = foods_list.find((food) => food._id === itemId);
      return {
        itemId,
        name: itemInfo.name,
        price: itemInfo.price,
        image: itemInfo.image,
        quantity: cartItems[itemId].quantity,
        totalPrice: itemInfo.price * cartItems[itemId].quantity,
        requirement: cartItems[itemId].requirement || '',
        status: "Ordering",
      };
    }),
  };
  setOrderData((prevOrders) => [...prevOrders, order]);
  setCartItems({});
};

const updateStatus = (orderId, itemId, newStatus) => {
  const updatedOrders = orderData.map((order) => {
    if (order.orderId === orderId) {
      const updatedItems = order.items.map((item) => {
        if (item.itemId === itemId) {
          return { ...item, status: newStatus };
        }
        return item;
      });
      return { ...order, items: updatedItems };
    }
    return order;
  });
  return updatedOrders;
};

setOrderData(updatedOrders);

const getTotalFoodCount = () => {
  return orderData.reduce((count, order) => {
    order.items.forEach((item) => {
      count += item.quantity;
    });
  }, 0);
};

const clearOrders = () => {
  setOrderData([]);
};

```

ส่วนที่ 3: พัฒนาการดึงข้อมูลรายการอาหารจากฐานข้อมูล

```
const getFoodData = async () => {
  try {
    const response = await axios.get(` ${backendURL}/api/product/list`);
    if (response.data.success) {
      setFoods_list(response.data.product);
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};

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

ส่วนเสริม: การส่ง ข้อมูล พิงก์ชัน และ state ไปให้คอมโพเนนท์ต่าง ๆ ภายในระบบ นำไปใช้งาน

```
const contextValue = {
    foods_list,
    cartItems,
    setCartItems,
    setToCart,
    search,
    setSearch,
    getCartCount,
    updateQuantity,
    removeItem,
    getCartAmount,
    currency,
    placeOrder,
    updateStatus,
    orderData,
    getTotalFoodCount,
    clearOrders,
    tableNumber,
    setTableNumber,
    backendURL,
    totalFoodCount,
    setTotalFoodCount,
    userID,
    setUserID,
    checkTable,
    available,
    loading,
};

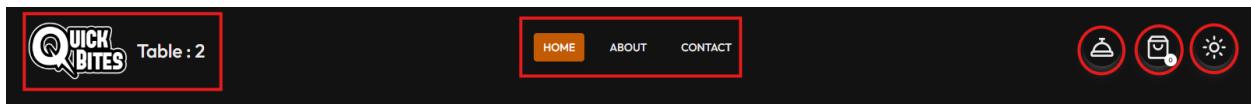
return (
    <StoreContext.Provider value={contextValue}>
        {props.children}
    </StoreContext.Provider>
);
```

คอมโพเนนท์

Navbar.jsx

คำอธิบาย

ແຄບນໍາທາງດ້ານບນຂອງເວັບໄຊຕີ່ຈະແສດງ ໂລໂກ້ ມາຍເລີໂຕ້ຈະ ປຸ່ມກດໄປຢັງໜ້າ “ໜ້າຫລັກ” “ເກື່ອງກັບເຮົາ” ແລະ “ຕິດຕ່ອເຮົາ” ປຸ່ມເຮີຍພັນກົງການ ປຸ່ມກດໄປຢັງໜ້າຮາຍກາຮາກອາຫາຣທີ່ຖູກສັ່ງ ແລະປຸ່ມກດເພື່ອສລັບໂໜມດແສງ/ມືດ



1

2

3 4 5

ໂຄດສໍາຄັງ

ສ່ວນທີ 1: ການແສດງໂລໂກ້ ແລະມາຍເລີໂຕ້ຈະ

```
<div className="flex items-center justify-between pt-4 sm:py-5 px-4 sm:px-8 font-medium">
  {location.pathname === '/' ? (
    <div className='flex gap-4 pt-2 '>
      <AlignLeft onClick={() => setVisible(true)} className='size-6 sm:size-10 cursor-pointer md:hidden text-Text'/>
      <Link to="/" className='flex md:flex-col lg:flex-row items-center gap-3'>
        <img className='w-16 sm:w-24 md:w-32 ' src={assets.logo} alt="" />
        <h1 className='w-36 text-sm xl:text-md sm:text-lg md:text-2xl text-Text md:text-center whitespace nowrap'>Table : {tableNumber}</h1>
      </Link>
    </div>
  ) : (
    <div>
      <Link to="/" className='flex md:flex-col lg:flex-row items-center gap-3'>
        <ArrowLeft className='size-7 sm:size-10 cursor-pointer md:hidden text-Text'/>
        <img className='w-16 sm:w-24 md:w-32 ' src={assets.logo} alt="" />
        <h1 className='w-36 text-sm xl:text-md sm:text-lg md:text-2xl text-Text text-center whitespace nowrap'>Table : {tableNumber}</h1>
      </Link>
    </div>
  )}
</div>
```

ສ່ວນທີ 2: ການແສດງປຸ່ມກດໄປຢັງໜ້າຫລັກທີ່ 3 ຜ້າ (“ໜ້າຫລັກ” “ເກື່ອງກັບເຮົາ” ແລະ “ຕິດຕ່ອເຮົາ”)

```
<ul className="hidden md:flex gap-5 text-sm text-Text justify-center w-full mx-auto">
  <NavLink to="/" className={({ isActive }) => `flex flex-col items-center gap-1 px-3 py-2 rounded ${isActive ? 'bg-BG_sec text-white shadow-lg' : ''}`}>
    <p>HOME</p>
    <hr className="w-2/4 border-none h-[1.5px] bg-gray-700 hidden" />
  </NavLink>
  <NavLink to="/about" className={({ isActive }) => `flex flex-col items-center gap-1 px-3 py-2 rounded ${isActive ? 'bg-BG_sec text-white shadow-lg' : ''}`}>
    <p>ABOUT</p>
    <hr className="w-2/4 border-none h-[1.5px] bg-gray-700 hidden" />
  </NavLink>
  <NavLink to="/contact" className={({ isActive }) => `flex flex-col items-center gap-1 px-3 py-2 rounded ${isActive ? 'bg-BG_sec text-white shadow-lg' : ''}`}>
    <p>CONTACT</p>
    <hr className="w-2/4 border-none h-[1.5px] bg-gray-700 hidden" />
  </NavLink>
</ul>
```

ส่วนที่ 3: การแสดงปุ่มกดเรียกพนักงาน

```
<div onClick={handleCall} className="cursor-pointer">
  <div className='w-9 h-9 sm:w-12 sm:h-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center'>
    <ConciergeBell className="size-6 sm:size-8 text-Text" alt="Concierge Bell"/>
  </div>
</div>
```

- โดยที่มีการเชื่อมต่อ API เข้ากับระบบหลังบ้าน (Admin Dashboard)

```
const handleCall = async () => {
  try {
    const response = await axios.post(backendURL + '/api/table/call' , {table : tableNumber});
    if (response.data.success) {
      toast.success(response.data.message, {
        position: "top-center",
        autoClose: 3000,
        hideProgressBar: true,
        closeOnClick: true,
        pauseOnHover: false,
        draggable: true,
      });
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};
```

ส่วนที่ 4: การแสดงปุ่มกดไปยังหน้ารายการอาหารที่ลูกสั่ง

```
<Link to="/orderSummary" className="relative">
  <div className='w-9 h-9 sm:w-12 sm:h-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center'>
    <ShoppingBag className="size-6 sm:size-8 text-Text" alt="Shopping Bag" />
  </div>
  <p className="absolute right-0 bottom-0 transform translate-x-1/6 translate-y-1/6 w-4 text-center leading-4 bg-BG_Blk text-BG aspect-square rounded-full text-[8px] shadow-md">{totalFoodCount}</p>
</Link>
```

ส่วนที่ 5: การแสดงปุ่มกดเพื่อสลับโหมดแสง/มืด

```
<div onClick={() => setDarkMode(!darkMode)} className="cursor-pointer">
    <div className="w-9 h-9 sm:w-12 sm:h-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center">
        {darkMode ? <Sun className="size-6 sm:size-8 text-Text" /> : <Moon className="size-6 sm:size-8 text-Text" />}
    </div>
</div>
```

```
useEffect(() => {
    const savedMode = localStorage.getItem('themeMode');
    if (savedMode === 'dark') {
        setDarkMode(true);
    }
}, []);

useEffect(() => {
    localStorage.setItem('themeMode', darkMode ? 'dark' : 'light');

    if (darkMode) {
        document.body.classList.add('dark');
        document.body.classList.remove('light');
    } else {
        document.body.classList.add('light');
        document.body.classList.remove('dark');
    }
}, [darkMode]);
```

ส่วนเสริม: การแสดงແຄນນຳທາງ ໃນກຣັບທີ່ເຂົ້າໃຈ່ງນເວີບໄຊຕື່ຜ່ານມືອຄື້ອ ປຶ້ງຈະມີການເພີ່ມຮາຍລະເອີຍດ
ຂອງໄວດີລູກຄ້າໃຫ້ແກ່ແຕ່ລະຄນ

```
/* Table data */


Table : <span className="font-bold">{tableNumber}</span></h1>

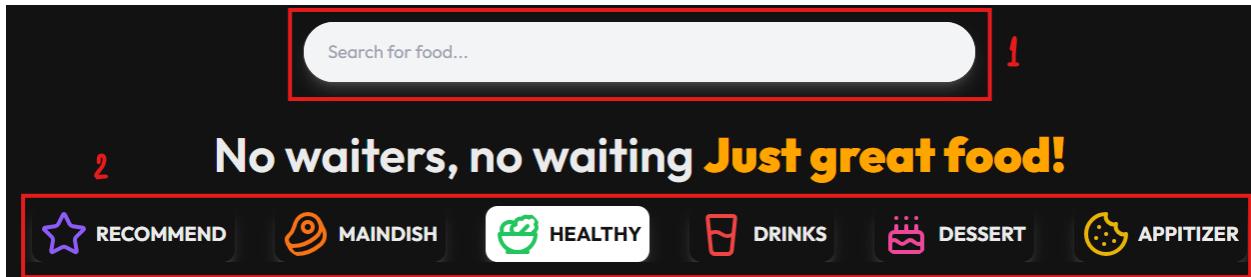

# UserID : <span className="font-bold">{userID}</span></h1>


```

ExploreMenu.jsx

คำอธิบาย

ส่วนควบคุมการแสดงผลรายการอาหาร ตามหมวดหมู่ หรือตามคำค้นหาชื่ออาหาร



โค้ดสำคัญ

ส่วนที่ 1: แบบการค้นหารายการอาหาร

```
/* Search Bar */

```

ส่วนที่ 2: แบบกรองรายการอาหารตามหมวดหมู่

```
/* Scrollable Menu */


{menu_list.map((item, index) => (
  <div
    key={index}
    onClick={() => setCategory((prev) => (prev === item.menu_name ? 'All' : item.menu_name))}
    className={'flex items-center gap-2 bg-BG_px-2 py-1 rounded-xl shadow-lg shadow-Text/20 transition-all ease-in-out cursor-pointer '}
      {category === item.menu_name ? 'bg-BG_Black text-BG' : 'text-Text'}
    >
    {/* Icon Section */}
    <div className="hover:scale-105 relative w-12 h-12 overflow-hidden rounded-full shrink-0">
      {React.createElement(item.menu_image || ChefHat, { className: `w-full h-full object-center ${item.color}` })}
    </div>
    {/* Text Section */}
    <div className="flex flex-col">
      <p className="text-md sm:lg md:text-xl font-bold uppercase">{item.menu_name}</p>
    </div>
  </div>
))
</div>


```

FoodDisplay.jsx

คำอธิบาย

ส่วนแสดงรายการอาหาร โดยพิจารณาจากคำค้นหาและหมวดหมู่ที่เลือกไว้ รวมถึงแคลอรี่
จำนวนรายการอาหารที่เลือก และแสดงราคาโดยรวมของอาหาร

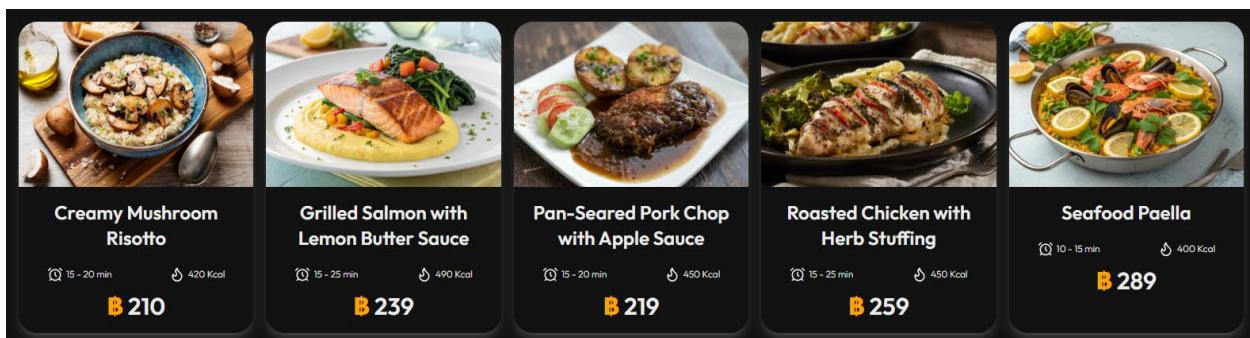
โค้ดสำคัญ

ส่วนที่ 1: การคัดกรองรายการอาหารตามการค้นหา หรือตามหมวดหมู่ที่เลือก

```
// Filter foods based on search term and selected category
const filteredFoods = foods_list.filter(item =>
  item.name.toLowerCase().includes(search.toLowerCase()) &&
  (category === 'All' || category === item.category || (item.recommend && category === 'Recommend'))
);

// Group items by category
const groupedFoods = filteredFoods.reduce((categories, item) => {
  const cat = item.category || 'Uncategorized';
  if (!categories[cat]) {
    categories[cat] = [];
  }
  categories[cat].push(item);
  return categories;
}, {});
```

ส่วนที่ 2: การแสดงผลรายการอาหาร



```

{Object.keys(groupedFoods).map((categoryName) => (
  <div key={categoryName}>
    /* Category Header with Icon */
    <h2 className="text-xl sm:text-3xl mt-5 sm:mt-1 font-bold text-Text px-4 sm:px-8 flex items-center gap-2">
      <div className="hover:scale-105 relative w-12 h-12 overflow-hidden rounded-full shrink-0">
        {(() => {
          const item = menu_list.find(item => item.menu_name === categoryName) || {};
          return React.createElement(item.menu_image || ChefHat, {
            className: `w-full h-full object-center ${item.color || ''}`
          });
        })()
      </div>
    {categoryName}
  </h2>

  /* Food Items data-aos="fade-up"*/
  <div className='grid grid-cols-2 md:grid-cols-3 lg:grid-cols-4 xl:grid-cols-5 gap-4 gap-y-6 sm:py-5 px-4 sm:px-8'>
    {groupedFoods[categoryName].map((item, index) => (
      <FoodItem
        key={index}
        id={item._id}
        time={item.time}
        Kcal={item.Kcal}
        name={item.name}
        price={item.price}
        image={item.image[0]}
      />
    )));
  </div>
</div>
))}
));
)
)}
)
)}
```

ส่วนที่ 3: การแสดงผลแบบสรุปจำนวนรายการอาหารที่สั่ง และค่าใช้จ่าย



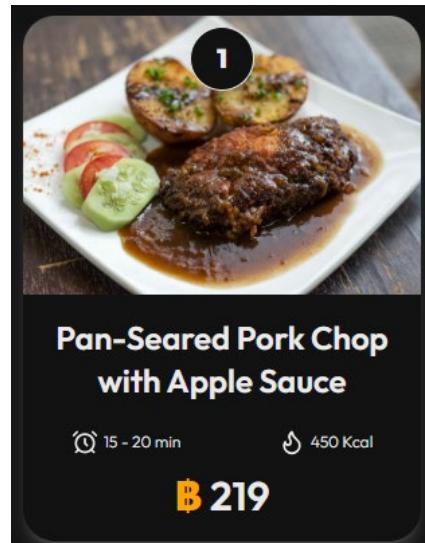
```

getCartCount() > 0 && (
  <div className="fixed bottom-0 left-1/2 transform -translate-x-1/2 w-full p-5 pb-8 md:p-10 bg-BG shadow-xl shadow-t shadow-Text/20 rounded-lg">
    <Link to="/cart" className="flex items-center justify-center">
      <button
        className="flex items-center justify-between w-full md:w-[50%] bg-Button text-BG px-6 sm:px-8 py-3 text-sm sm:text-base rounded-lg hover:bg-orange-500 active:bg-orange-700 transition duration-300">
        <div className="flex gap-2 md:gap-8 items-center">
          <ShoppingCart className="size-12" />
          <p className="text-xl md:text-2xl lg:text-3xl">{getCartCount()} In cart</p>
        </div>
        <p className="text-xl md:text-2xl lg:text-3xl">{currency} {getCartAmount().toFixed(2)}</p>
      </button>
    </Link>
  </div>
)}
```

FoodItem.jsx

คำอธิบาย

ส่วนแสดงรายละเอียดของแต่ละรายการอาหารภายในกล่อง



โค้ดสำหรับ

```

<Link className="shadow-lg shadow-Text/20 text-Text cursor-pointer mt-4 rounded-3xl" to={`/food/${id}`}>
  <div className="relative rounded-t-3xl overflow-hidden">
    <img src={image} className="w-full h-40 sm:h-52 hover:scale-110 transition-all ease-in-out" alt={name} />
    {cartItems[id] && (
      <div className="absolute top-2 start-1/2 transform -translate-x-1/2 w-10 h-10 sm:w-12 sm:h-12 bg-BG rounded-full border shadow-lg flex items-center justify-center">
        <p className="text-xl font-black text-Text">{cartItems[id].quantity || 0}</p> /* Display quantity */
      </div>
    )}
  </div>

  <div className="p-4 gap-3 text-base flex flex-col items-center justify-center sm:text-2xl">
    /* Name */
    <p className="pb-2 font-semibold text-center">{name}</p>

    /* Details Section */
    <div className="flex text-center justify-between w-full px-0 lg:px-5 ">
      <p className="text-xs font-light flex items-center gap-1 text-center justify-center">
        <AlarmClock className='size-5' /> {time[0]} - {time[1]} min
      </p>
      <p className="text-xs font-light flex items-center gap-1 text-center justify-center">
        <Flame className='size-5' /> {kcal} Kcal
      </p>
    </div>

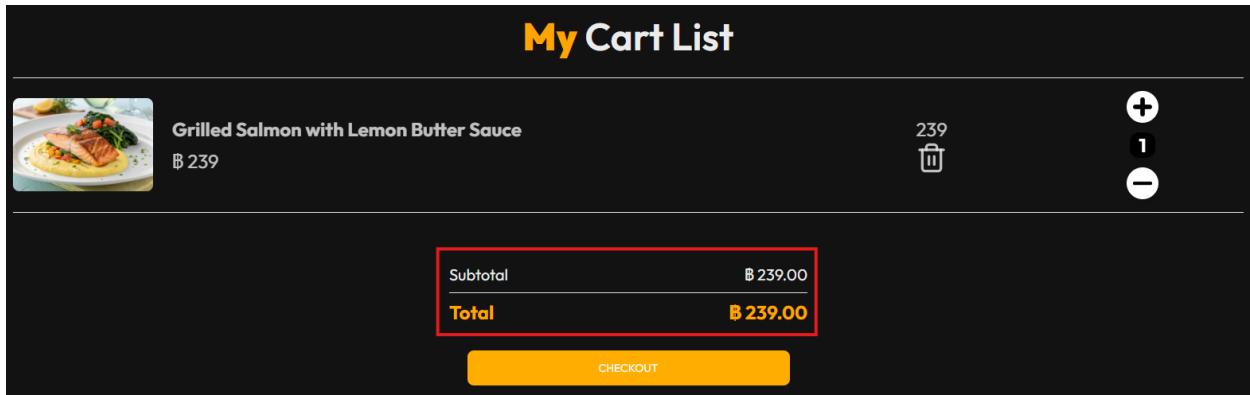
    /* Price */
    <p className="text-2xl sm:text-3xl font-semibold">
      <b>$</b> {price}
    </p>
  </div>
</Link>

```

CartTotal.jsx

คำอธิบาย

ส่วนแสดงค่าใช้จ่ายของรายการอาหารที่ถูกสั่ง ในหน้าตะกร้าสินค้า



โค้ดสำคัญ

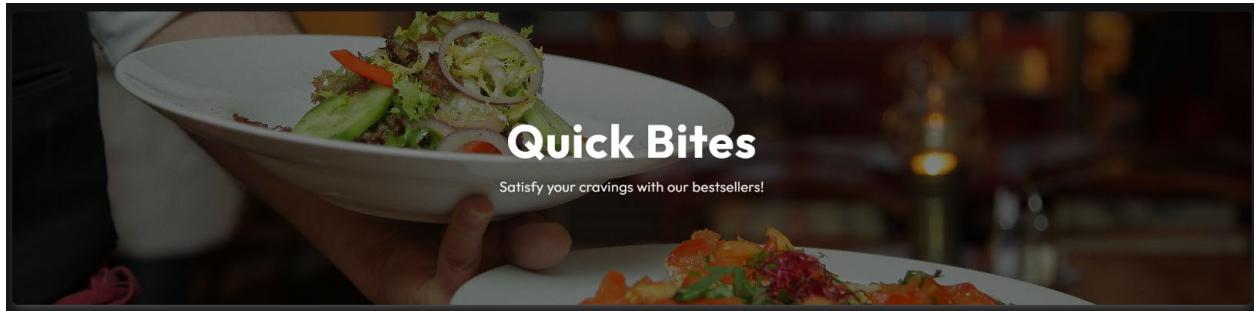
```
const CartTotal = () => {
  const {currency, getCartAmount} = useContext(StoreContext);

  return (
    <div className='w-full'>
      <div className='flex flex-col gap-2 mt-2 text-sm'>
        <div className='flex justify-between text-xl text-Text'>
          <p>Subtotal</p>
          <p>{currency} {getCartAmount()}.00</p>
        </div>
        <hr />
        <div className='flex justify-between text-2xl'>
          <b>Total</b>
          <b>{currency} {getCartAmount() === 0 ? 0 : getCartAmount()}.00</b>
        </div>
      </div>
    </div>
  )
}
```

Hero.jsx

คำอธิบาย

แบบเนอร์หลักของร้านค้า ที่แสดงอยู่ด้านบนของหน้าเว็บ



โค้ดสำคัญ

```
<div className="hidden md:flex relative flex-col sm:flex-row shadow-lg shadow-Text/20 rounded-lg overflow-hidden">
  {/* Image Section */}
  <img
    src={menuItem ? menuItem.menu_main : assets.Home}
    className="w-full h-56 sm:h-96 md:h-[40vh] object-cover"
    alt=""
  />

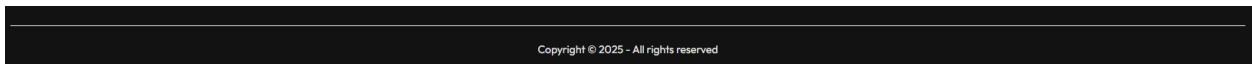
  {/* Overlay Section */}
  <div className="absolute inset-0 bg-black/50 flex items-center justify-center">

    <div className="text-center text-white px-4">
      <h1 className="text-4xl sm:text-5xl lg:text-6xl font-bold tracking-wide drop-shadow-lg">
        Quick Bites
      </h1>
      <p className="mt-4 text-base sm:text-lg font-light tracking-wide drop-shadow-md">
        Satisfy your cravings with our bestsellers!
      </p>
    </div>
  </div>
</div>
```

Footer.jsx

คำอธิบาย

ส่วนแสดงข้อมูลลิขสิทธิ์เกี่ยวกับบริษัท ที่แสดงอยู่ด้านล่างของหน้าเว็บ



โค้ดสำคัญ

```
<footer className='my-10 mt-40 text-sm'>
  <hr />
  <p className='py-5 text-sm text-center text-Text'>Copyright © {year} - All rights reserved</p>
</footer>
```

หน้าเว็บ

Home.jsx

คำอธิบาย

หน้าหลักของ QuickBites Webpage ที่ทำหน้าที่เป็นจุดเริ่มต้นให้ผู้ใช้ได้เลือกคุณรายการอาหาร และสั่งอาหาร ซึ่งมีการนำคอมโพnenท์ที่สำคัญมาประกอบเข้าเป็นโครงสร้างหลักของหน้า

1

Quick Bites
Satisfy your cravings with our bestsellers!

Discover The Best Food

Search for food...

No waiters, no waiting **Just great food!**

MainDish

Creamy Mushroom Risotto	Grilled Salmon with Lemon Butter Sauce	Pan-Seared Pork Chop with Apple Sauce	Roasted Chicken with Herb Stuffing	Seafood Paella
🕒 15 - 20 min 🍗 420 Kcal ฿ 210	🕒 15 - 25 min 🍗 490 Kcal ฿ 239	🕒 15 - 20 min 🍗 450 Kcal ฿ 219	🕒 15 - 25 min 🍗 450 Kcal ฿ 259	🕒 10 - 15 min 🍗 400 Kcal ฿ 289
Spaghetti Carbonara	Beef Steak with Peppercorn Sauce			
🕒 10 - 20 min 🍗 400 Kcal ฿ 199	🕒 15 - 25 min 🍗 530 Kcal ฿ 259			

Copyright © 2025 - All rights reserved **4**

2

3

โค้ดสำคัญ

```
return (
  <div className='mt-0 sm:mt-2 md:mt-3 lg:mt-5'>
    1 <Hero category={category}/>
    2 <ExploreMenu category={category} setCategory={setCategory}/>
    3 <FoodDisplay category={category}/>
    4 <Footer/>
  </div>
)
```

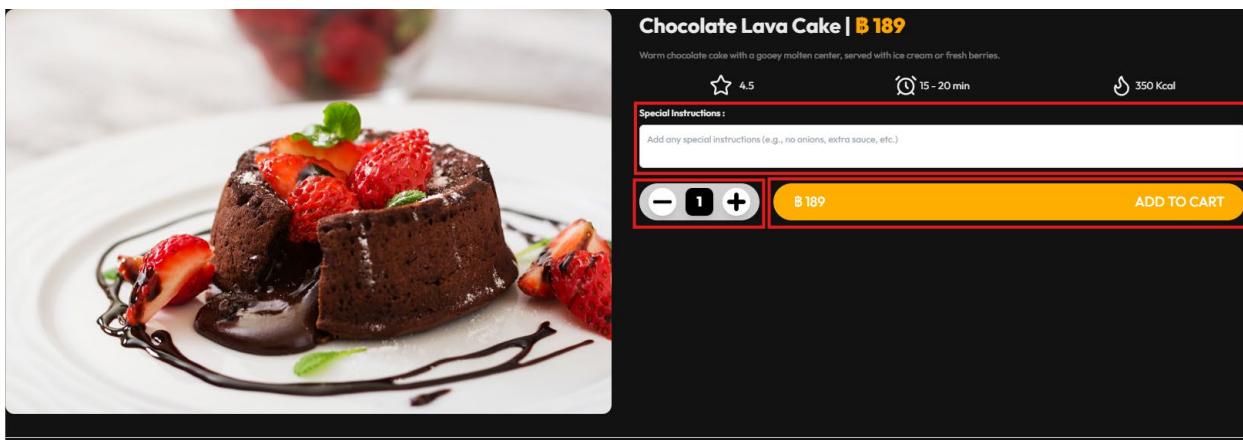
ส่วนเสริม: การแสดงรายการอาหารตามหมวดหมู่ที่เลือก โดยมีค่าเริ่มต้นเป็น "All" เพื่อแสดงรายการอาหารทั้งหมด

```
const [category, setCategory] = useState('All');
```

Food.jsx

คำอธิบาย

หน้าแสดงรายละเอียดของรายการอาหารที่ถูกเลือก โดยจะแสดงปุ่ม เพิ่มจำนวน ลดจำนวน และ เพิ่มสินค้าลงในตะกร้า รวมถึงยังมีพื้นที่ให้ผู้ใช้เขียนความต้องการเพิ่มเติม และการแสดงผลรายการอาหารที่เกี่ยวข้องอื่น ๆ



Chocolate Lava Cake | ₧ 189
 Warm chocolate cake with a gooey molten center, served with ice cream or fresh berries.
 ★ 4.5 15 - 20 min 350 Kcal
 Special Instructions:
 Add any special instructions (e.g., no onions, extra sauce, etc.)
 - 1 + ₧ 189 ADD TO CART

Related Food

 Panna Cotta with Raspberry Coulis ⌚ 10 - 15 min ⚡ 250 Kcal ₩ 169	 Tiramisu Cup ⌚ 10 - 15 min ⚡ 250 Kcal ₩ 159	 Chicken Satay with Peanut Sauce ⌚ 10 - 20 min ⚡ 300 Kcal ₩ 159	 Crispy Spring Rolls with Sweet Chili Sauce ⌚ 10 - 20 min ⚡ 290 Kcal ₩ 119	 Garlic Bread Supreme ⌚ 15 - 20 min ⚡ 250 Kcal ₩ 109
--	---	--	--	---

โค้ดสำคัญ

ส่วนเสริม: การดึงข้อมูลรายการอาหารเพื่อนำมาแสดงผล

```
const { foodId } = useParams();
const { foods_list, setToCart, cartItems } = useContext(StoreContext);
const [itemsCount, setItemsCount] = useState();
const [requirement, setRequirement] = useState();
const [productData, setProductData] = useState(null);
const currentIndex = productData ? foods_list.findIndex((item) => item._id === productData._id) : -1;

const relatedFoods = [];
if (currentIndex !== -1) {
    for (let i = -2; i <= 3; i++) {
        const index = (currentIndex + i + foods_list.length) % foods_list.length;
        if (index !== currentIndex) {
            relatedFoods.push(foods_list[index]);
        }
    }
}

const fetchProductData = () => {
    const product = foods_list.find((item) => item._id === foodId);
    if (product) setProductData(product);
};
```

ส่วนที่ 1: การแสดงรายละเอียดของเมนูอาหาร

```
/* Swiper for Product Images */


<Swiper
    pagination={{dynamicBullets: true, clickable: true}}
    modules={[Pagination]}
    spaceBetween={10}
    slidesPerView={1}
    className="flex items-center justify-center"
  >
    {productData.image.map((item, index) => (
      <SwiperSlide key={index}>
        <img
          src={item}
          className="object-cover"
          alt={`Product Image ${index + 1}`}
        />
      </SwiperSlide>
    )));
  </Swiper>
</div>

/* Product Information */


<h1 className="font-bold text-3xl sm:text-4xl mt-2 text-Text">{productData.name} | <b>฿ {productData.price}</b></h1>
  <p className="mt-5 text-Text/50">{productData.description}</p>

  /* Product Details */
  <div className="flex text-center justify-around w-full mt-5">
    <p className="text-sm sm:text-lg flex items-center gap-1 text-center justify-center text-Text">
      <Star className="size-8 sm:size-10 mr-2" />{productData.rate}
    </p>
    <p className="text-sm sm:text-lg flex items-center gap-1 text-center justify-center text-Text">
      <AlarmClock className="size-8 sm:size-10"/> {productData.time[0]} - {productData.time[1]} min
    </p>
    <p className="text-sm sm:text-lg flex items-center gap-1 text-center justify-center text-Text">
      <Flame className="size-8 sm:size-10" /> {productData.Kcal} Kcal
    </p>
  </div>


```

ส่วนที่ 2: การแสดงฟอร์มกรอกความต้องการของเมนูอาหารเพิ่มเติม

```
<div className="mt-4">
  <label htmlFor="requirements" className="block text-Text font-semibold mb-2">
    Special Instructions :
  </label>
  <textarea
    id="requirements"
    value={requirement}
    onChange={(e) => setRequirement(e.target.value)}
    className="w-full p-3 rounded-lg border border-gray-300 focus:outline-none focus:ring-2 focus:ring-orange-500 resize-none"
    placeholder="Add any special instructions (e.g., no onions, extra sauce, etc.)"
  />
</div>
```

ส่วนที่ 3: การแสดงปุ่มเพิ่มหรือลดจำนวนของเมนูอาหาร

```
<div className="flex items-center gap-2 bg-white/80 px-4 py-2 rounded-full shadow-md z-10 sm:gap-4 sm:px-4 sm:py-2">
  <div className='size-8 md:size-12 bg-white shadow-lg rounded-full p-2'>
    <img
      src={assets.minus}
      alt="Minus Button"
      onClick={() => {
        itemsCount > 0 && setItemsCount((prev) => prev - 1);
      }}
      className="bg w-12 text-2xl cursor-pointer text-red-500 hover:text-red-700"
    />
  </div>
  <div className="flex items-center justify-center w-8 h-8 md:w-12 md:h-12 bg-black shadow-lg rounded-xl">
    <p className="text-lg md:text-2xl text-white font-extrabold">{itemsCount || 0}</p>
  </div>

  <div className='size-8 md:size-12 bg-white shadow-lg rounded-full p-2'>
    <img
      src={assets.add}
      alt="Add Button"
      onClick={() => setItemsCount((prev) => prev + 1)}
      className="w-12 text-2xl cursor-pointer text-green-500 hover:text-green-700"
    />
  </div>
</div>
```

ส่วนที่ 4: การแสดงปุ่มเพิ่มสินค้าลงไปในตะกร้า

```
<Link to="/" className='flex'
  >
  <button
    onClick={() => setToCart(productData._id, itemsCount, requirement)}
    className="flex items-center justify-between w-full bg-Button text-white px-3 sm:px-5 xl:px-7 2xl:px-9 py-3 text-lg sm:text-xl rounded-full hover:bg-orange-500 active:bg-orange-700 transition duration-300"
  >
    <p className="text-md sm:text-2xl md:text-3xl xl:text-2xl font-medium text-white">
      {productData.price * itemsCount || 0}
    </p>
    <p className="text-md sm:text-2xl md:text-3xl xl:text-2xl font-medium text-white">
      ADD TO CART
    </p>
  </button>
</Link>
```

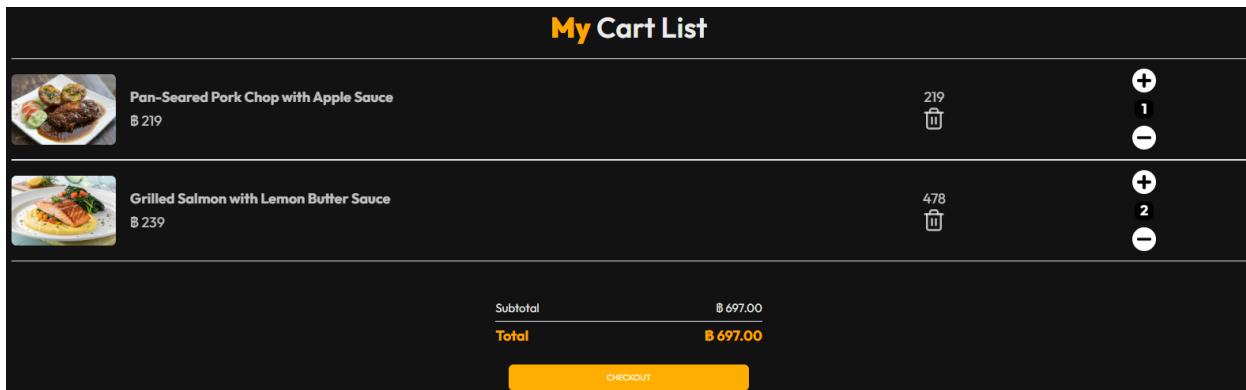
ส่วนที่ 5: การแสดงรายการอาหารที่เกี่ยวข้อง

```
<hr className='my-5 sm:my-10' />
/* Related Food */
<div>
  <h2 className="text-3xl sm:text-4xl font-bold text-Text">Related <b>Food</b></h2>
  <div className="w-full overflow-x-auto">
    <div className="flex gap-4 sm:py-5 justify-start 2xl:justify-center">
      {relatedFoods.map((item, index) => (
        <FoodItem
          id={item._id}
          time={item.time}
          Kcal={item.Kcal}
          name={item.name}
          price={item.price}
          image={item.image[0]}
        />
      ))
    </div>
  </div>
</div>
```

Cart.jsx

คำอธิบาย

หน้าต่างกร้าสินค้าที่ผู้ใช้สามารถตรวจสอบ และแก้ไข รายการอาหารที่เลือกไว้ก่อนที่จะทำการสั่งอาหาร โดยจะแสดง ปุ่มปรับจำนวนอาหาร ปุ่มลบเมนูอาหาร ปุ่มสั่งอาหาร และค่าใช้จ่ายรวม



โค้ดสำคัญ

ส่วนที่ 1: การแสดงรายการอาหารที่เลือกไว้

```
{cartData.map((item, index) => (
  <div key={index}>
    <div data-aos="fade-right" className="flex gap-3 md:gap-6 w-full">
      <img src={item.image[0]} className="w-28 sm:w-44 rounded-lg" alt="" />
      <div className='flex flex-col justify-center'>
        <p className="text-md md:text-2xl font-bold">{item.name}</p>
        <div className="flex items-center text-md md:text-2xl gap-5 mt-2">
          <p>฿ {item.price}</p>
        </div>
      </div>
    </div>
  </div>
)}
```

ส่วนที่ 2: การแสดงปุ่มปรับจำนวนอาหาร

```
<div data-aos="fade-left" className='md:w-[25%] flex flex-col gap-2 items-center justify-center'>
  <div className='flex flex-col gap-3 items-center'>
    <div className='size-6 md:size-10 bg-white shadow-lg rounded-full p-2'>
      <img onClick={() => updateQuantity(item.id, item.quantity + 1)}>
        className="w-10 cursor-pointer" src={assets.add} alt="" />
    </div>

    <div className="flex items-center justify-center size-6 md:size-8 bg-black shadow-lg rounded-md md:rounded-xl">
      <p className="text-lg md:text-2xl text-white font-bold">{item.quantity}</p>
    </div>

    <div className='size-6 md:size-10 bg-white shadow-lg rounded-full p-2'>
      <img src={assets.minus} onClick={() =>
        item.quantity > 1 && updateQuantity(item.id, item.quantity - 1)
      }>
        className="w-10 cursor-pointer" alt="" />
    </div>
  </div>
</div>
```

ส่วนที่ 3: การแสดงปุ่มลบเมนูอาหาร

```
<div data-aos="fade-left" className='md:w-[25%] p-2 flex flex-col items-center'>
  <p className='text-lg md:text-2xl'>{item.quantity*item.price}</p>
  <Trash2 className='size-4 md:size-10 cursor-pointer' onClick={() => removeItem(item.id)}/>
</div>
```

ส่วนที่ 4: การแสดงปุ่มสั่งอาหาร

```
<div className="flex justify-center">
  <div className="w-full sm:w-[450px] mt-10">
    <CartTotal/>
    <div className="w-full flex justify-center text-end">
      <Link onClick={() => {placeOrder();onSubmitHandler();}} to="/orderSummary" className="w-[90%] text-center select-none rounded-lg bg-Button text-white text-sm my-8 px-8 py-3">
        CHECKOUT
      </Link>
    </div>
  </div>
</div>
```

OrderSummary.jsx

คำอธิบาย

หน้าสรุปคำสั่งซื้ออาหาร โดยจะแสดงรายละเอียดของแต่ละคำสั่งซึ่ง ของผู้ใช้แต่ละคนที่จำแนกตามไอดีลูกค้า มีปุ่มชำระเงิน ที่เมื่อกดแล้ว ระบบจะทำการสร้าง QR Code สำหรับชำระเงิน

Order Summary

Order #1: A		฿ 259.00
	Roasted Chicken with Herb Stuffing (x1)	฿ 259.00
Completed		
Additional : No Req		
	Pan-Seared Pork Chop with Apple Sauce (x1)	฿ 219.00
Completed		
Additional : No Req		

Total: ฿478.00

Order #2: A		฿ 318.00
	Salmon Salad with Citrus Dressing (x2)	฿ 318.00
Cooking		
Additional : No Req		

Total: ฿318.00

Order #3: A		฿ 210.00
	Creamy Mushroom Risotto (x1)	฿ 210.00
Completed		
Additional : No Req		
	Creme Brulee (x1)	฿ 159.00
Completed		
Additional : No Req		

13 Order Check Bill ฿ 2688.00

Payment Details



Table : 2

Total Amount	฿ 2688.00
Total Food	13

Confirm Payment

โค้ดสำคัญ

ส่วนที่ 1: การแสดงรายการอาหารที่ถูกสั่งทั้งหมด

```
<div>
  {orderData.map((order, index) => (
    <div key={index} className="mb-4 border-b pb-4 text-Text">
      <h3 className="font-medium text-lg">Order #{index + 1} : {order.userID}</h3>
      <ul className="mt-2">
        {order.products.map((item, idx) => {
          return (
            <li data-aos="fade-up" key={idx} className="flex flex-col sm:flex-row items-start sm:items-center justify-between py-2 md:pr-20">
              <div className="flex items-center justify-center">
                <img src={item.image[0]} alt={item.name} className="w-32 sm:w-44 object-cover mr-4 rounded-lg" />
                <div className="flex flex-col gap-2">
                  <span className="text-md md:text-lg md:w-full lg:text-2xl">{item.name} (x{item.quantity})</span>
                  <span className="text-md md:text-lg lg:text-2xl text-center select-none badge-${order.status.toLowerCase()}">{order.status}</span>
                  <p className="text-md md:text-1xl lg:text-1xl truncate max-w-60 sm:max-w-80 md:max-w-80 lg:max-w-96">
                    <b>Additional</b> : {item.requirement ? item.requirement : 'No Req'}
                  </p>
                </div>
              </div>
            </li>
          );
        ))}
      </ul>
      <div className="mt-2 text-md md:text-2xl font-medium">
        <strong>Total: {currency}</strong>
        {order.products.reduce((total, item) => total + item.totalPrice, 0).toFixed(2)}
      </div>
    </div>
  ))}
</div>
```

ส่วนที่ 2: การแสดงปุ่มชำระเงิน

```
{totalFoodCount > 0 && (
  <div className="fixed bottom-0 left-1/2 transform -translate-x-1/2 w-full p-5 pb-8 md:p-10 bg-orange shadow-lg rounded-lg z-50">
    <div className="flex items-center justify-center">
      <button
        onClick={() => handlePayment(calculateTotalPrice().toFixed(2))}>
        <span>Pay</span>
      </button>
      <p>Total Food: {totalFoodCount}</p>
      <p>Check Bill</p>
      <p>Currency: {calculateTotalPrice().toFixed(2)}</p>
    </div>
  </div>
)}
```

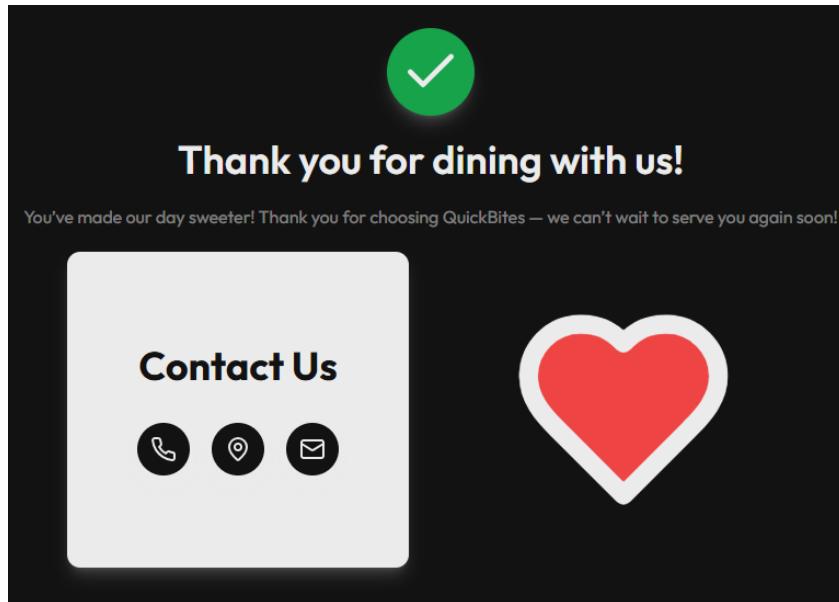
ส่วนที่ 3: การสร้าง QR Code ให้ลูกค้าสแกนเพื่อชำระเงิน

```
{isPaymentOpen && (
  <div className="fixed inset-0 flex items-center justify-center bg-black bg-opacity-60 z-50">
    <div className="bg-white p-6 rounded-lg shadow-lg w-96">
      <div className="flex items-center justify-between mb-5">
        <h1>Payment Details</h1>
        <div onClick={() => setIsPaymentOpen(false)}>
          <span>Close</span>
        </div>
      </div>
      <div className="flex flex-col items-center justify-center">
        <img alt="QR Code" src={imageQrCode} className="w-full" />
        <p>Table : {tableNumber}</p>
      </div>
      <hr />
      <div className="flex justify-between my-2 text-2xl">
        <p>Total Amount</p>
        <b>{calculateTotalPrice().toFixed(2)}</b>
      </div>
      <div className="flex justify-between my-2 text-2xl">
        <p>Total Food</p>
        <b>{totalFoodCount}</b>
      </div>
      <hr />
      <div className="flex justify-center gap-4 mt-6">
        <a href="/thankyou" onClick={() => handleCheckBill()}>
          Confirm Payment
        </a>
      </div>
    </div>
  </div>
)}
```

Thank.jsx

คำอธิบาย

หน้าขอบคุณที่แสดงให้เห็นหลังจากที่ผู้ใช้ทำการสั่งซื้อเสร็จสิ้น โดยจะแสดงข้อความขอบคุณ
พร้อมไอคอน และมีข้อมูลการติดต่อรวมถึงองค์ประกอบภาพอื่น ๆ



โค้ดสำคัญ

```
<div className="flex flex-col gap-5 items-center justify-center min-h-screen bg-BG">
  <div className="w-16 h-16 sm:w-20 sm:h-20 bg-green-600 rounded-full shadow-lg shadow-Text/20 flex items-center justify-center">
    <Check className="size-11 sm:size-14 text-Text" alt="Check" />
  </div>
  <h2 className="text-4xl font-semibold text-center text-Text">Thank you for dining with us!</h2>
  <p className="text-Text/50 w-[70%] md:w-[50%] text-center">
    You've made our day sweeter! Thank you for choosing QuickBites – we can't wait to serve you again soon!
  </p>
  <div className="grid grid-cols-1 md:grid-cols-2 sm:gap-10 justify-center">
    <div className="flex flex-col justify-center items-center gap-8 bg-Text p-16 shadow-lg shadow-Text/20 rounded-xl">
      <strong className="text-4xl text-BG">Contact Us</strong>
      <div className="flex gap-5">
        <div className="w-9 h-9 sm:w-12 sm:h-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center">
          <Phone className="size-4 sm:size-6 text-Text" alt="Phone" />
        </div>
        <div className="w-9 h-9 sm:w-12 sm:h-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center">
          <MapPin className="size-4 sm:size-6 text-Text" alt="MapPin" />
        </div>
        <div className="w-9 h-9 sm:w-12 sm:h-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center">
          <Mail className="size-4 sm:size-6 text-Text" alt="Mail" />
        </div>
      </div>
      <div className="flex flex-col justify-center items-center p-10">
        <Heart className="size-48 sm:size-52 fill-red-500 text-Text" alt="Heart" />
      </div>
    </div>
  </div>
</div>
```

About.jsx

คำอธิบาย

หน้าแสดงข้อมูลเกี่ยวกับประวัติ จุดประสงค์ และวิสัยทัศน์ของ QuickBites

About Us



QuickBites was founded with a simple goal — to make dining easier, faster, and more enjoyable. In a world where time is precious, we believe technology can enhance your restaurant experience without taking away the human touch.

From effortless menu browsing to real-time order tracking, we've designed QuickBites to bring convenience to both diners and restaurant staff. Whether you're a hungry guest or a hardworking team member, we've got your back with smart, seamless tools that just work.

Our Mission

At QuickBites, our mission is to revolutionize the way people dine by blending technology with hospitality — making every meal not just a bite, but an experience.

Why Us

<p>Quality Assurance :</p> <p>We take pride in our commitment to quality assurance, ensuring that our products meet the highest standards of durability, performance, and functionality.</p>	<p>Convenience :</p> <p>We understand the importance of convenience, which is why our products are designed to be user-friendly, easy to use, and accessible to everyone.</p>	<p>Exceptional Customer Service :</p> <p>Our team of dedicated customer service representatives is dedicated to providing exceptional customer service and support, ensuring that you have a smooth and enjoyable shopping experience.</p>
---	--	---

โค้ดสำคัญ

```

<div className="h-full p-6 mt-5 bg-BG border-t">
  <div className="w-full font-bold sm:font-semibold text-2xl sm:text-5xl text-start sm:text-center mb-3 text-Text">
    <h1><b>About</b> Us</h1>
  </div>

  <div className="my-10 flex flex-col md:flex-row gap-16">
    <img className="w-full md:max-w-[490px]" src={assets.home} alt="" />
    <div className="flex flex-col justify-center gap-6 md:w-2/4 text-Text">
      <p>
        QuickBites was founded with a simple goal — to make dining easier, faster, and more enjoyable. In a world where time is precious, we believe technology can enhance your restaurant experience without taking away the human touch.
      </p>
      <p>
        From effortless menu browsing to real-time order tracking, we've designed QuickBites to bring convenience to both diners and restaurant staff. Whether you're a hungry guest or a hardworking team member, we've got your back with smart, seamless tools that just work.
      </p>
      <p><b>Our Mission</b></p>
      <p>At QuickBites, our mission is to revolutionize the way people dine by blending technology with hospitality — making every meal not just a bite, but an experience.</p>
    </div>
  </div>
  <div className="w-full font-bold sm:font-semibold text-2xl sm:text-5xl text-start sm:text-center mb-3 text-Text">
    <h1><b>Why</b> Us</h1>
  </div>
  <div className="flex flex-col md:flex-row text-sm mb-20">
    <div className="border px-10 md:px-16 py-8 sm:py-20 flex flex-col gap-5">
      <b>Quality Assurance :</b>
      <p>We take pride in our commitment to quality assurance, ensuring that our products meet the highest standards of durability, performance, and functionality.</p>
    </div>
    <div className="border px-10 md:px-16 py-8 sm:py-20 flex flex-col gap-5">
      <b>Convenience :</b>
      <p>We understand the importance of convenience, which is why our products are designed to be user-friendly, easy to use, and accessible to everyone.</p>
    </div>
    <div className="border px-10 md:px-16 py-8 sm:py-20 flex flex-col gap-5">
      <b>Exceptional Customer Service :</b>
      <p>Our team of dedicated customer service representatives is dedicated to providing exceptional customer service and support, ensuring that you have a smooth and enjoyable shopping experience.</p>
    </div>
  </div>
  <!-- NewsletterBox -->
</div>

```

Contact.jsx

คำอธิบาย

หน้าแสดงข้อมูลติดต่อของบริษัท QuickBites

Contact Us



Our Store

54709 Willms Station
Suite 350, Washington, USA

Tel : (415) 555-5555
Email : Dummy@forever.com

Careers at Forever

Learn more about our teams and job openings.

[Explore Jobs](#)

โค้ดสำคัญ

```

<div className="h-full p-6 mt-5 bg-BG border-t">
  <div className='w-full font-bold sm:font-semibold text-2xl sm:text-5xl text-start sm:text-center mb-3 text-Text'>
    <h1><b>Contact</b> Us</h1>
  </div>
  <div className='my-10 flex flex-col justify-center md:flex-row gap-10 mb-28'>
    <img className='w-full md:max-w-[480px]' src={assets.Home} alt="" />
    <div className='flex flex-col justify-center items-start gap-6'>
      <p className='font-semibold text-xl text-Text/60'>Our Store</p>
      <p className='text-Text'>54709 Willms Station <br /> Suite 350, Washington, USA</p>
      <p className='text-Text'>Tel : (415) 555-5555 <br />Email : Dummy@forever.com</p>
      <p className='font-semibold text-xl text-Text/60'>Careers at Forever</p>
      <p className='text-Text'>Learn more about our teams and job openings.</p>
      <button className='border border-Text px-8 py-4 text-sm hover:bg-Button hover:text-white transition-all duration-500 text-Text'>Explore Jobs</button>
    </div>
  </div>
</div>

```

Admin Dashboard

แกนหลักของระบบ

App.jsx

คำอธิบาย

- จัดการในส่วนของโทเคน และบทบาทที่จะมอบหมายให้กับพนักงาน ผ่านหน้าล็อกอิน
- การกำหนดโครงสร้างหลัก ได้แก่ แบบนำทาง และແບບด้านข้าง ให้แก่หน้าต่าง ๆ
- การแสดงผลรายละเอียดเฉพาะของหน้าต่าง ๆ
- การจัดเส้นทางระหว่างหน้าต่าง ๆ

โค้ดสำคัญ

ส่วนที่ 1: การมอปโทเคน และบทบาทให้แก่พนักงาน ผ่านการล็อกอินเข้าใช้งาน

```
const [token, setToken] = useState(localStorage.getItem("token") ? localStorage.getItem("token") : "");
const [role, setRole] = useState(localStorage.getItem("role") ? localStorage.getItem("role") : "");
useEffect(() => {
  localStorage.setItem("token", token)
  localStorage.setItem("role", role)
}, [token, role])
```

```
{token === "" ? <Login setToken={setToken} setRole={setRole} />
```

ส่วนที่ 2: การกำหนดโครงสร้างหลัก แบบนำทาง และແບບด้านข้าง

```
/* Navbar */
<Navbar expanded={expanded} setExpanded={setExpanded} setToken={setToken} role={role} />

<div className="flex pt-20">
  /* Sidebar */
  <Sidebar expanded={expanded} setToken={setToken} role={role} />
```

ส่วนที่ 3: การแสดงผลรายละเอียดเฉพาะของหน้าต่าง ๆ รวมถึงการจัดเส้นทางให้แก่หน้าต่าง ๆ ภายในระบบให้สามารถเข้าถึงได้ ในการนี้ที่ลือกอินสำเร็จ

```
/* Main Content */

```

DashboardContext.jsx

คำอธิบาย

ไฟล์นี้เป็นแกนหลักของระบบ Admin Dashboard โดยให้ Context สำหรับการจัดการ state และการสื่อสารกับ API ภายใน DashboardContext จะรวมการดึงข้อมูลและอัปเดตสถานะสำหรับข้อมูลวิเคราะห์ (Analytics), รายการอาหาร, พนักงาน, คำสั่งซื้อ และข้อมูลโภชนาญาติ พร้อมทั้งตั้งค่าเชื่อมต่อแบบ Real-Time ผ่าน Socket เพื่ออัปเดตคำสั่งซื้อและสถานะโภชนาญาติแบบทันที

โค้ดสำคัญ

ส่วนที่ 1: การตั้งค่า State และ Socket Connection

```
const [dateRange, setDateRange] = useState(localStorage.getItem('dateRange') || 'monthly');
const [amountMenu, setAmountMenu] = useState(0);
const [analyticsData, setAnalyticsData] = useState([]);
const [foodList, setFoodList] = useState([]);
const [latestFood, setLatestFood] = useState(null);
const [employeeList, setEmployeeList] = useState([]);
const [orders, setOrders] = useState([]);
const [tables, setTables] = useState([]);
const socket = useState(() => io(backendURL, { transports: ['websocket', 'polling'], withCredentials: true }))[0];
```

ส่วนที่ 2: พัฒนาการดึงข้อมูลประเภทต่าง ๆ

- ข้อมูลยอดขายตามช่วงเวลาที่กำหนด

```
const fetchAnalytics = async () => {
  try {
    const response = await axios.get(backendURL + "/api/analytics/get", {
      params: { dateRange }
    });
    if (response.data.success) {
      setAnalyticsData(response.data.sales);
    }
  } catch (error) {
    console.error(error);
    toast.error("Error updating order status");
  }
};
```

- ข้อมูลรายการอาหาร

```
const fetchFood = async () => {
  try {
    const response = await axios.get(backendURL + '/api/product/list');
    if (response.data.success) {
      setAmountMenu(response.data.product.length);

      const latest = response.data.product[response.data.product.length-1];
      setLatestFood(latest);

      const sortedFood = response.data.product.sort((a, b) => a.category.localeCompare(b.category));
      setFoodList(sortedFood);

    }
  } catch (error) {
    toast.error("Failed to fetch food list");
  }
};
```

- ข้อมูลรายชื่อพนักงาน

```
const fetchEmployee = async () => {
  try {
    const response = await axios.get(backendURL + "/api/employee/listStaff");
    if (response.data.success) {
      setEmployeeList(response.data.staff);
    }
  } catch (error) {
    toast.error("Failed to fetch employee list");
  }
};
```

- ข้อมูลคำสั่งซื้อ

```
const fetchOrders = async () => {
  try {
    const response = await axios.get(backendURL + `/api/order/list`);
    if (response.data.success) {
      toast.success(response.data.message);
      setOrders(response.data.order);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};
```

- ข้อมูลโต๊ะ

```
const fetchTable = async () => {
  try {
    const response = await axios.get(backendURL + '/api/table/list');
    if(response.data.success){
      setTables(response.data.tables)
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};
```

ส่วนที่ 3: การตั้งค่า Real-Time Updates ผ่าน Socket

```
useEffect(() => {
    socket.on("orderUpdated", fetchOrders);
    socket.on("tableUpdated", fetchTable);

    return () => {
        socket.off("orderUpdated", fetchOrders);
        socket.off("tableUpdated", fetchTable);
        socket.disconnect();
    };
}, [socket]);
```

ส่วนที่ 4: การอัปเดตข้อมูลเมื่อช่วงเวลาเปลี่ยน

```
useEffect(() => {
    fetchFood();
    fetchOrders();
    fetchTable();
    fetchEmployee();
    fetchAnalytics();
}, [dateRange]);
```

ส่วนที่ 5: การสรุปข้อมูล

```
const totalOrders = analyticsData.reduce((sum, entry) => sum + entry.orderAmount, 0);
const totalCustomers = analyticsData.reduce((sum, entry) => sum + entry.customerAmount, 0);
const totalIncome = analyticsData.reduce((sum, entry) => sum + entry.totalIncome, 0);
```

ส่วนที่ 6: การประมวลผลข้อมูลยอดขายอาหาร

```
const foodSales = {};

analyticsData.forEach((entry) => {
  entry.OrderFood.forEach((food) => {
    if (!foodSales[food.name]) {
      foodSales[food.name] = {
        quantitySold: food.quantitySold,
        image: food.image[0],
      };
    } else {
      foodSales[food.name].quantitySold += food.quantitySold;
    }
  });
});

const popularFood = Object.entries(foodSales)
  .sort((a, b) => b[1].quantitySold - a[1].quantitySold)
  .slice(0, 5)
  .map(([name, { quantitySold, image }]) => ({ name, quantitySold, image }));
```

ส่วนที่ 7: การส่งค่าที่ถูกประมวลผลกลับไปยังส่วนอื่น ๆ ภายในระบบ

```
const contextValue = {
  amountMenu, setAmountMenu,
  analyticsData, setAnalyticsData, foodList, fetchFood,
  totalOrders, totalCustomers, totalIncome, popularFood, fetchEmployee,
  employeeList, orders, fetchOrders, fetchAnalytics, dateRange, setDateRange,
  tables, setTables, fetchTable, latestFood,
};
```

```
return (
  <DashboardContext.Provider value={contextValue}>
    {props.children}
  </DashboardContext.Provider>
);
```

คอมโพเนนท์

Navbar.jsx

คำอธิบาย

ແຄບນໍາທາງດ້ານบนຂອງເວັບໄຊຕ່າງໆ ທີ່ຈະແສດງ ໂລໂກ້ ປຸ່ມການແຈ້ງເຕືອນເນື່ອມີລູກຄ້າກົດປຸ່ມຂອງຄວາມ
ຫຼວຍເຫຼືອ ແລະປຸ່ມກົດເພື່ອສລັບໂທມດແສງ/ນິດ



ໂຄດສໍາຄັນ

ສ່ວນທີ 1: ການແສດງປຸ່ມແສດງແຄບດ້ານຂ້າງ (ເຂືອນຕ່ອງກັບ Sidebar.jsx)

```
<button
  onClick={() => {
    if (window.innerWidth < 768) { //md
      setVisible(true);
    } else {
      setExpanded((curr) => !curr);
    }
  }}
  className="flex p-2 rounded-lg text-Text hover:bg-Text/30"
>
  {expanded ? <AlignLeft className="size-8" /> : <AlignJustify className="size-8" />}
</button>
```

ສ່ວນທີ 2: ການແສດງໂລໂກ້ ແລະຫຼືອບຣີ່ຫັກ

```
<Link to="/" className="flex items-center gap-4 font-medium text-Text">
  <div className="p-3">
    <Airplay className="size-8 md:size-11" />
  </div>
  <h1 className="hidden sm:block text-Text text-xl font-semibold">QuickBites</h1>
</Link>
```

ส่วนที่ 3: การตรวจสอบบทบาทของพนักงาน ซึ่งจะส่งผลต่อการแสดงผลชื่อ Dashboard (ตำแหน่งแอดมินจะเห็น “Admin Dashboard” ส่วนพนักงานทั่วไปจะเห็น “Employee Dashboard”)

```
{role === "admin" && <h1 className="md:text-2xl font-bold text-Text md:mr-10">Admin Dashboard</h1>
|| <h1 className="md:text-2xl font-bold text-Text md:mr-10">Employee Dashboard</h1>}
```

ส่วนที่ 4: การแสดงปุ่มการแจ้งเตือนเมื่อมีลูกค้ากดปุ่มขอความช่วยเหลือ

```
<div onClick={() => setShowNotifications(!showNotifications)} className="cursor-pointer">
    <div className="size-9 md:size-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center">
        <Bell className="md:size-8 text-Text" />
    </div>
</div>

/* Notifications dropdown */
{showNotifications && (
    <div className="absolute top-16 right-4 bg-Text text-BG rounded-lg shadow-lg p-4 w-64 mt-3 max-h-80 overflow-y-auto">
        <h3 className="font-bold text-lg mb-2">Notifications</h3>
        {tables.map((table) => (
            table.callWaiter && (
                <div key={table.table} className="flex justify-between items-center mb-3 p-4 border-t-4 border-BG rounded-lg shadow-md">
                    <span className="text-sm font-medium">`Table ${table.table}: Call Waiter`</span>
                    <span className="text-sm">Need assistance</span>
                </div>
            )
        )));
    </div>
)}
```

ส่วนที่ 5: การแสดงปุ่มกดเพื่อสลับโหมดแสง/มืด

```
/* Dark Mode */
useEffect(() => {
  const savedTheme = localStorage.getItem('themeMode');
  if (savedTheme === 'dark') {
    setDarkMode(true);
  }
}, []);

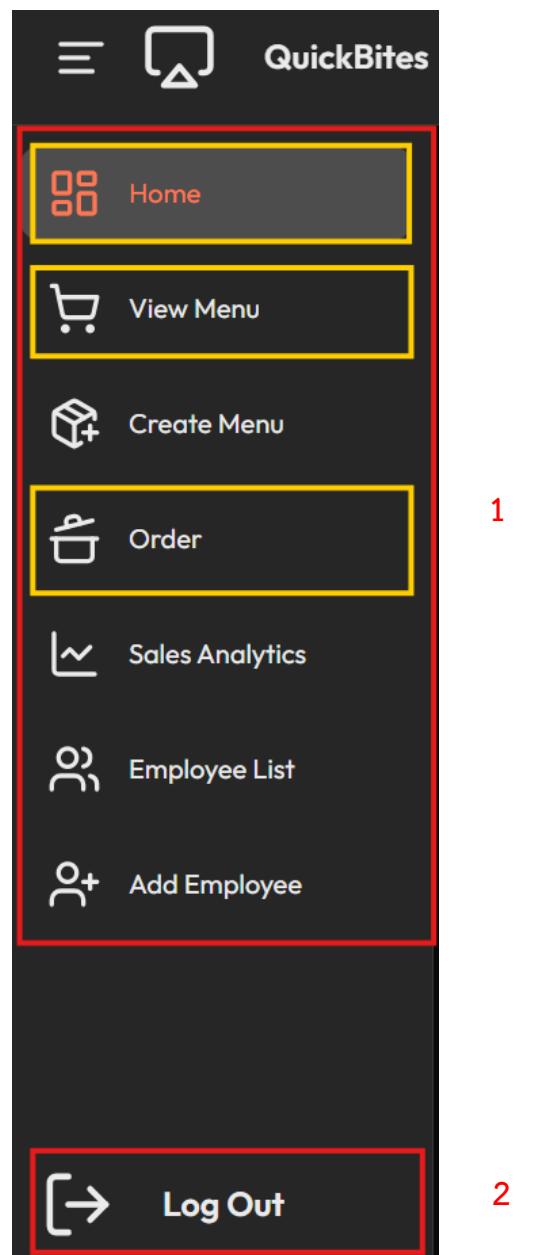
/* Dark Mode */
useEffect(() => {
  localStorage.setItem('themeMode', darkMode ? 'dark' : 'light');
  if (darkMode) {
    document.body.classList.add('dark');
    document.body.classList.remove('light');
  } else {
    document.body.classList.add('light');
    document.body.classList.remove('dark');
  }
}, [darkMode]);
```

```
<div onClick={() => setDarkMode(!darkMode)} className="cursor-pointer">
  <div className="size-9 md:size-12 bg-BG rounded-full shadow-lg shadow-Text/20 flex items-center justify-center">
    {darkMode ? <Sun className="md:size-8 text-Text" /> : <Moon className="md:size-8 text-Text" />}
  </div>
</div>
```

Sidebar.jsx

คำอธิบาย

ແຄບນໍາທາງດ້ານຊ້າງ ຈຶ່ງຈະແສດງປຸ່ມນໍາທາງໄປຢັງໜ້າອື່ນ ແລ້ວ ຈຶ່ງຈະແສດງຜລແຕກຕ່າງກັນໄປຕາມ
ບທບາທຂອງພນັກງານ ຮວມເລື່ອມີປຸ່ມອອກຈາກຮບບ່ອຍຸດ້ານລ່າງ
ໝາຍເຫດ: ແອດມິນສາມາຮດເຂົ້າລຶ່ງໄດ້ທຸກປຸ່ມ (🔴) ແຕ່ພນັກງານທົ່ວໄປສາມາຮດເຂົ້າລຶ່ງໄດ້ເຄີ່ງບ່າງປຸ່ມ (🟡)



ໂຄ້ດສຳຄັນ

ສ່ວນທີ 1: ການແສດງປຸ່ມທຸກປຸ່ມ ທີ່ຈຸກຄັດກຮອງໂດຍຕໍາແໜ່ງຂອງພນັກງານ

```
/* Menu */


{sidemenu
    .filter((item) => item.role === "all" || item.role === role)
    .map((item, index) => {
      const Icon = item.icon;
      return (
        <Link
          key={index}
          to={item.path}
          className={`flex gap-4 pl-4 p-3 items-center rounded-xl transition-colors hover:bg-Text/20 ${activeMenu === item.path ? "text-Highlight bg-Text/20" : ""}`}
        >
          <Icon className="size-8" />
          {expanded && <span>{item.title}</span>}
        </Link>
      );
    )}
  </div>


```

ສ່ວນທີ 2: ການແສດງປຸ່ມອອກຈາກຮບບ

```
/* Log Out Button */


<div className="p-3">
    <LogOut className="size-11" />
  </div>
  {expanded && <h1 className="text-Text text-xl font-semibold">Log Out</h1>}
</div>

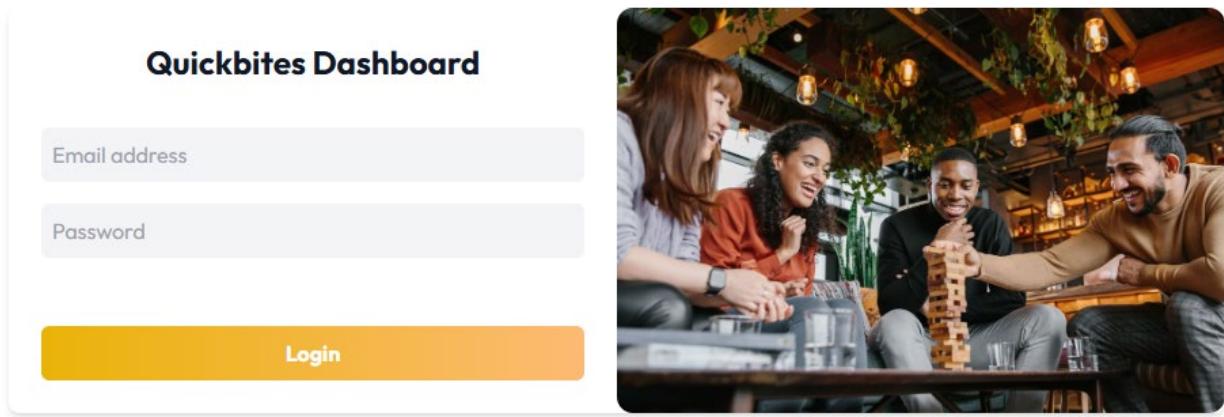

```

หน้าล็อกอิน

Login.jsx

คำอธิบาย

หน้าเข้าสู่ระบบสำหรับ Admin Dashboard โดยจะแสดงฟอร์มให้พนักงานกรอกอีเมลและรหัสผ่าน เมื่อส่งฟอร์ม ระบบจะส่งคำขอไปยัง API เพื่อตรวจสอบข้อมูล หากสำเร็จ ระบบจะมอบโทเคน และบันทึกให้กับพนักงานตามอีเมลที่ใช้ล็อกอิน และนำทางไปยังหน้าหลักของ Dashboard



โค้ดสำคัญ

ส่วนที่ 1: การตรวจสอบความถูกต้องและการส่งคำขอ Login

```

const isValid = async (e) => {
  e.preventDefault();
  const endpoint = "/api/employee/login";
  try {
    const response = await axios.post(backendURL + endpoint, { email, password });
    if (response.data.success) {
      navigate("/");
      setToken(response.data.token);
      setRole(response.data.role);
      toast.success(`Logged in as ${response.data.role === "admin" ? "Admin" : "Employee"}`);
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};

```

ส่วนที่ 2: การแสดงผลฟอร์มล็อกอิน

```

return (
  <div
    className="relative flex flex-row items-center justify-center h-screen"
    style={
      isMobile
      ?
      {
        backgroundImage: `url(${scheduleImg})`,
        backgroundSize: 'cover',
        backgroundPosition: 'center'
      }
      :
      {}
    }
  >
  {/* Mobile overlay */}
  {isMobile && (
    <div className="absolute inset-0 bg-black bg-opacity-50"></div>
  )}
  <div className="grid grid-cols-1 md:grid-cols-2 bg-white rounded-lg shadow-md w-fit max-w-4xl relative">
    {/* Left column: Login form */}
    <div className="p-6 flex flex-col">
      <h2 className="text-2xl font-bold text-gray-900 mb-4 text-center">
        Quickbites Dashboard
      </h2>
      <form className="flex flex-col flex-grow justify-between" onSubmit={isValid}>
        <div>
          <input
            onChange={(e) => setEmail(e.target.value)}
            value={email}
            type="email"
            className="block w-full bg-gray-100 text-gray-900 border-0 rounded-md p-2 mb-4 mt-4"
            focus:bg-gray-200 focus:outline-none focus:ring-1 focus:ring-blue-500"
            placeholder="Email address"
          />
          <input
            onChange={(e) => setPassword(e.target.value)}
            value={password}
            type="password"
            className="block w-full bg-gray-100 text-gray-900 border-0 rounded-md p-2 mb-4"
            focus:bg-gray-200 focus:outline-none focus:ring-1 focus:ring-blue-500"
            placeholder="Password"
          />
        </div>
        <button
          type="submit"
          className="bg-gradient-to-r from-yellow-500 to-orange-300 text-white font-bold py-2 px-4 rounded-md mt-4 transition ease-in-out duration-150"
        >
          Login
        </button>
      </form>
    </div>
  </div>
  {/* Right column (image) - hidden on mobile */}
  <div className="hidden md:block w-full max-w-md">
    <img
      className="rounded-xl"
      src={scheduleImg}
      alt=""
    />
  </div>
</div>
);
};

```

หน้าหลัก

Home.jsx

คำอธิบาย

หน้าแสดงภาพรวมของข้อมูลสำคัญ และกิจกรรมล่าสุด ได้แก่ จำนวนคำสั่งซื้อ จำนวนลูกค้า จำนวนเมนู และรายได้รวม นอกจากนี้ยังมีกราฟแสดงยอดขาย รายการอาหารยอดนิยม และส่วนของคำสั่งซื้อล่าสุด



โค้ดสำคัญ

ส่วนที่ 1: ແກບສຽງจำนวน คำสั่งซื้อ ลูกค้า และรายการอาหาร รวมถึงรายได้รวมของร้านค้า

- กล่องแสดงจำนวนคำสั่งซื้อ

```
<div className="rounded-xl bg-BG flex justify-between px-4 2xl:px-10 py-5">
  <div className="flex flex-col gap-3 lg:gap-7">
    <h2 className="text-lg sm:text-2xl">Orders</h2>
    <p className="text-xl sm:text-2xl 2xl:text-4xl font-bold">{totalOrders}</p>
  </div>
  <div className="flex items-center justify-center">
    <Inbox className="size-14 sm:size-16 2xl:size-24" />
  </div>
</div>
```

- กล่องแสดงจำนวนลูกค้า

```
<div className="rounded-xl bg-BG flex justify-between px-4 2xl:px-7 py-5 ">
  <div className="flex flex-col gap-3 lg:gap-7">
    <h2 className="text-lg sm:text-2xl">Customers</h2>
    <p className="text-xl sm:text-2xl 2xl:text-4xl font-bold">{totalCustomers}</p>
  </div>
  <div className="flex items-center justify-center">
    <ContactRound className="size-14 sm:size-16 2xl:size-24" />
  </div>
</div>
```

- กล่องแสดงจำนวนรายการอาหาร

```
<div className="rounded-xl bg-BG flex justify-between px-4 2xl:px-7 py-5 ">
  <div className="flex flex-col gap-3 lg:gap-7">
    <h2 className="text-lg sm:text-2xl">Menu</h2>
    <p className="text-xl sm:text-2xl 2xl:text-4xl font-bold">{amountMenu}</p>
  </div>
  <div className="flex items-center justify-center">
    <CookingPot className="size-14 sm:size-16 2xl:size-24" />
  </div>
</div>
```

- กล่องแสดงรายได้รวมของร้านค้า

```
<div className="rounded-xl bg-BG flex justify-between px-4 2xl:px-7 py-5">
  <div className="flex flex-col gap-3 lg:gap-7">
    <h2 className="text-lg sm:text-2xl whitespace nowrap">Total Income</h2>
    <p className="text-xl sm:text-2xl 2xl:text-4xl font-bold">฿ {totalIncome.toFixed(2)}</p>
  </div>
  <div className="flex items-center justify-center">
    <Wallet className="size-14 sm:size-16 2xl:size-24" />
  </div>
</div>
```

ส่วนที่ 2: กล่องแสดงกราฟสถิติยอดขายตามช่วงเวลา

```

<div className="min-h-96 xl:col-span-2 rounded-xl bg-BG flex flex-col p-4">
  <h2 className="text-lg sm:text-3xl px-4 2xl:px-10 py-5">Sales Figures</h2>
  <ResponsiveContainer width="100%" height="100%" className={"text-white"}>
    <LineChart data={analyticsData} margin={{ top: 20, right: 40, bottom: 20 }}>
      <defs>
        <linearGradient id="colorIncome" x1="0" y1="0" x2="0" y2="1">
          <stop offset="5%" stopColor="#4f46e5" stopOpacity={0.8} />
          <stop offset="95%" stopColor="#4f46e5" stopOpacity={0} />
        </linearGradient>
      </defs>

      <CartesianGrid strokeDasharray="2 2" strokeOpacity={0.5} />
      <XAxis
        dataKey="date"
        tickFormatter={({date) => format(new Date(date), "d - MMM : HH:mm")}
        tick={{ fontSize: 12, fill: "#888" }}
        tickMargin={10}
        angle={-15}
        dy={5}
      />
      <YAxis tick={{ fontSize: 12, fill: "#888" }} />
      <Tooltip
        contentStyle={{ backgroundColor: "#000", borderRadius: 10 }}
        labelFormatter={({label) => format(new Date(label), "d - MMM : HH:mm")}
      />

      <Line
        type="natural"
        dataKey="totalIncome"
        stroke="#4f46e5"
        strokeWidth={3}
        dot={{ r: 4, fill: "#4f46e5" }}
        activeDot={{ r: 6, fill: "#4f46e5", stroke: "#fff", strokeWidth: 2 }}
      />
    </LineChart>
  </ResponsiveContainer>
</div>

```

ส่วนที่ 3: แบบแสดงรายการอาหารยอดนิยม

```
<div className="lg:col-span-1 rounded-xl bg-BG flex flex-col px-10 py-5">
  <div className="flex justify-between items-center mb-5 gap-2">
    <h1 className="text-lg sm:text-3xl">Popular Food</h1>
    <Link to="/view_menu" className="flex items-center justify-center self-start w-auto h-full p-4 py-2 gap-3 hover:border border-Text bg-Text/20 rounded-2xl whitespace nowrap">
      <p>View all</p>
    </Link>
  </div>
  <ul className="flex flex-col gap-3">
    {popularFood.map((food) => (
      <div key={food.name} className="flex items-center gap-5">
        <img className="w-28 rounded-xl" src={food.image} alt={food.name} />
        <div>
          <p className="font-bold">{food.name}</p>
          <p className="font-thin">Sold : {food.quantitySold}</p>
        </div>
      </div>
    )))
  </ul>
</div>
```

ส่วนที่ 4: แบบแสดงรายการคำสั่งซื้อล่าสุด

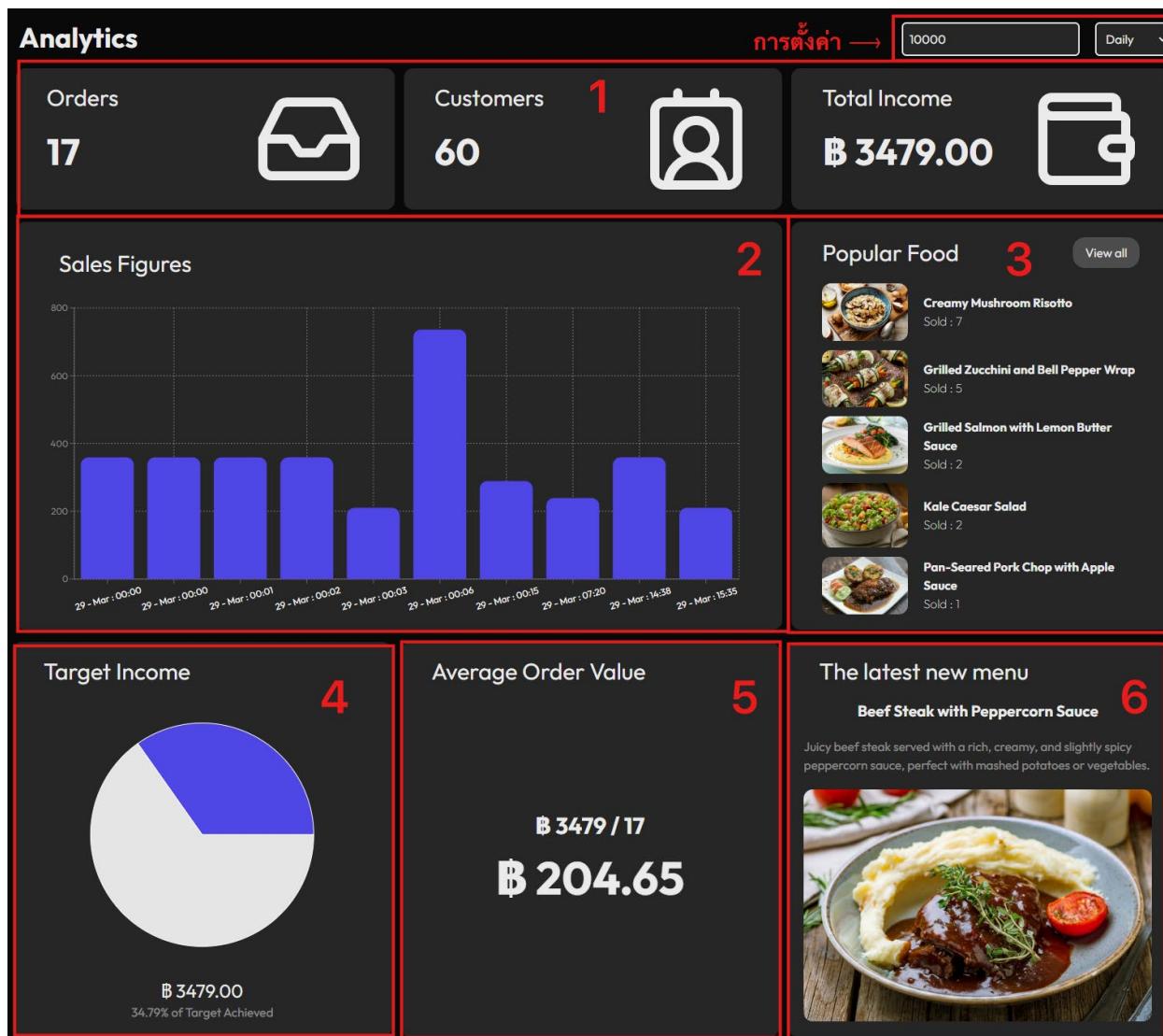
```
<div className="lg:col-span-3 lg:row-span-3 grid lg:grid-cols-3 rounded-xl gap-3">
  <div className="lg:col-span-3 rounded-xl bg-BG flex items-center justify-center">
    <div className="rounded-xl bg-BG flex flex-col justify-between px-4 2xl:px-10 py-5 w-full">
      <div className="flex justify-between items-center mb-5 gap-2">
        <h1 className="text-lg sm:text-3xl">Recent Order Request</h1>
        <Link to="/order" className="flex items-center justify-center self-start w-auto h-full p-4 py-2 gap-3 hover:border border-Text bg-Text/20 rounded-2xl whitespace nowrap">
          <p>View all</p>
        </Link>
      </div>
      <div className="flex flex-col items-center justify-center">
        {orders
          .sort((a, b) => new Date(b.createdAt) - new Date(a.createdAt))
          .slice(0, 5)
          .map((order) => (
            <div key={order._id} className="flex flex-col bg-BG p-4 border-b w-full">
              <div className="flex gap-4 mb-2 items-center">
                <h3 className="text-xl lg:text-3xl font-bold">Table : {order.tableNumber}</h3>
                <p className="text-Text">Order Status</p>
                <p className="text-Text">Time : {format(new Date(order.createdAt), "HH:mm")}</p>
              </div>
              <div className="grid grid-cols-2 md:grid-cols-3 lg:grid-cols-4 xl:grid-cols-5 gap-4">
                {order.products.map((product) => (
                  <div className="shadow-lg shadow-Text/20 text-Text mt-4 rounded-3xl">
                    <div className="relative rounded-t-3xl overflow-hidden">
                      <img src={product.image[0]} alt="" className="w-full transition-all ease-in-out" />
                      <div className="absolute top-2 right-2 w-10 h-10 sm:w-12 sm:h-12 bg-BG rounded-full border shadow-lg flex items-center justify-center">
                        <p className="text-xl font-black text-Text">{product.quantity}</p>
                      </div>
                    </div>
                    <div className="p-2 gap-3 text-base flex flex-col items-center justify-center sm:text-lg">
                      <p className="pb-2 font-semibold text-center">Product Name</p>
                    </div>
                  </div>
                )))
              </div>
            </div>
          )))
        </div>
      </div>
    </div>
  </div>
</div>
```

ระบบวิเคราะห์ข้อมูลยอดขาย

Analytics.jsx

คำอธิบาย

หน้าแสดงข้อมูลทางสถิติเพิ่มเติมจากหน้าหลัก โดยสามารถเลือกดูแบบ รายวัน รายเดือน และรายปี ซึ่งจะแสดงการ์ดสรุป จำนวนคำสั่งซื้อ จำนวนลูกค้า รายได้รวม กราฟแสดงยอดขาย รายการอาหารนิยม เป้าหมายยอดขาย (สามารถตั้งค่าได้) รายได้เฉลี่ยต่อคำสั่งซื้อ และรายการอาหารใหม่ล่าสุด ซึ่งสามารถเข้าถึงได้โดยแอดมินเท่านั้น



โค้ดสำคัญ

ส่วนเสริม: การตรวจสอบว่าพนักงานเป็นแอดมิน

```
if (role !== "admin") {
    return (
        <div className="flex justify-center items-center mt-6">
            <div className="bg-red-100 text-red-700 border border-red-500 rounded-lg px-6 py-4">
                <p className="text-center font-semibold">Access Denied</p>
                <p className="text-center text-sm">Admins only.</p>
            </div>
        </div>
    );
}
```

ส่วนที่ 1: ແບບสรุปจำนวน คำสั่งซื้อ ลูกค้า และรายการอาหาร รวมถึงรายได้รวมของร้านค้า

- กล่องแสดงจำนวนคำสั่งซื้อ

```
<div className="rounded-xl bg-BG flex justify-between px-4 2xl:px-10 py-5 ">
    <div className="flex flex-col gap-3 md:gap-7">
        <h2 className="text-lg md:text-2xl lg:text-3xl">Orders</h2>
        <p className="text-2xl md:text-2xl lg:text-3xl 2xl:text-5xl font-bold">{totalOrders}</p>
    </div>
    <div className="flex items-center justify-center">
        <Inbox className="size-16 sm:size-20 2xl:size-36"/>
    </div>
</div>
```

- กล่องแสดงจำนวนลูกค้า

```
<div className="rounded-xl bg-BG flex justify-between px-4 2xl:px-10 py-5 ">
    <div className="flex flex-col gap-3 md:gap-7">
        <h2 className="text-lg md:text-2xl lg:text-3xl">Customers</h2>
        <p className="text-2xl md:text-2xl lg:text-3xl 2xl:text-5xl font-bold">{totalCustomers}</p>
    </div>
    <div className="flex items-center justify-center">
        <ContactRound className="size-16 sm:size-20 2xl:size-36"/>
    </div>
</div>
```

- กล่องแสดงรายได้รวมของร้านค้า

```
<div className="rounded-xl bg-BG flex justify-between px-4 2xl:px-10 py-5 ">
  <div className="flex flex-col gap-3 md:gap-7">
    <h2 className="text-lg md:text-2xl lg:text-3xl whitespace nowrap">Total Income</h2>
    <p className="text-2xl md:text-2xl lg:text-3xl 2xl:text-5xl font-bold">฿ {totalIncome.toFixed(2)}</p>
  </div>
  <div className="flex items-center justify-center">
    <Wallet className="size-16 sm:size-20 2xl:size-36"/>
  </div>
</div>
```

ส่วนที่ 2: กล่องแสดงกราฟสถิติยอดขายตามช่วงเวลา

```
<div className="min-h-96 xl:col-span-2 rounded-xl bg-BG flex flex-col p-4">
  <h2 className="text-lg sm:text-3xl px-4 2xl:px-10 py-5">Sales Figures</h2>
  <ResponsiveContainer width="100%" height="100%" className="text-white">
    <BarChart data={analyticsData} margin={{ top: 20, right: 40, bottom: 20 }}>
      <CartesianGrid strokeDasharray="2 2" strokeOpacity={0.5} />
      <XAxis
        dataKey="date"
        tickFormatter={(date) => format(new Date(date), "d - MMM : HH:mm")}
        tick={{ fontSize: 12, fill: "#fff" }}
        tickMargin={10}
        angle={-15}
        dy={5}
      />
      <YAxis tick={{ fontSize: 12, fill: "#888" }} />
      <Tooltip
        contentStyle={{ backgroundColor: "#000", borderRadius: 10 }}
        labelFormatter={(label) => format(new Date(label), "d - MMM : HH:mm")}
      />
      <Bar
        dataKey="totalIncome"
        fill="#4f46e5"
        radius={[10, 10, 0, 0]}
      />
    </BarChart>
  </ResponsiveContainer>
</div>
```

ส่วนที่ 3: แบบแสดงรายการอาหารยอดนิยม

```
<div className="md:col-span-1 rounded-xl bg-BG flex flex-col px-10 py-5">
  <div className="flex justify-between items-center mb-5 gap-2">
    <h1 className="text-lg sm:text-3xl">Popular Food</h1>
    <Link to="/view_menu" className="flex items-center justify-center self-start w-auto h-full p-4 py-2 gap-3 hover:border border-Text bg-Text/20 rounded-2xl whitespace nowrap">
      <>>View all</>
    </Link>
  </div>
  <ul className="flex flex-col gap-3">
    {popularFood.map((food) =>
      <div key={food.name} className="flex items-center gap-5">
        <img className="w-28 rounded-xl" src={food.image} alt={food.name} />
        <div>
          <p className="font-bold">{food.name}</p>
          <p className="font-thin">Sold : {food.quantitySold}</p>
        </div>
      </div>
    ))}
  </ul>
</div>
```

ส่วนที่ 4: กล่องแสดงเป้าหมายยอดขาย

```
<div className="lg:col-span-1 rounded-xl bg-BG flex flex-col items-center justify-center p-5">
  <h2 className="text-lg sm:text-3xl self-start ml-5 mb-2">Target Income</h2>
  <ResponsiveContainer width="100%" height='100%'>
    <PieChart>
      <Pie
        data={pieData}
        dataKey="value"
        nameKey="name"
        cx="50%"
        cy="50%"
        outerRadius="80%"
        fill="#4f46e5"
      >
        {pieData.map((entry, index) => (
          <Cell key={`${cell}-${index}`} fill={index === 0 ? "#4f46e5" : "#e5e5e5"} />
        ))}
      </Pie>
    </PieChart>
  </ResponsiveContainer>
  <div className="flex flex-col items-center">
    <p className="text-2xl">฿ {totalIncome.toFixed(2)}</p>
    <p className="text-md text-Text/50">{targetPercentage.toFixed(2)}% of Target Achieved</p>
  </div>
</div>
```

ส่วนเสริม: สามารถตั้งค่าเป้าหมายได้

```
<input
    type="number"
    value={targetIncome}
    onChange={(e) => setTargetIncome(parseFloat(e.target.value)) }
    className="border-2 p-2 rounded-md bg-BG text-Text"
    placeholder="Set Target Income"
    min="0"
/>
```

ส่วนที่ 5: กล่องแสดงรายได้เฉลี่ยต่อคำสั่งซื้อ

```
<div className="lg:col-span-1 rounded-xl bg-BG flex flex-col items-center justify-center p-5">
    <h2 className="text-lg sm:text-3xl self-start ml-5 mb-2">Average Order Value</h2>
    <div className="flex flex-col items-center justify-center h-full sm:gap-5 ">
        <p className="text-xl sm:text-3xl font-bold">฿ {totalIncome} / {totalOrders}</p>
        <p className="text-3xl sm:text-6xl font-bold">฿ {(totalIncome / totalOrders).toFixed(2)}</p>
    </div>
</div>
```

ส่วนที่ 6: กล่องแสดงรายการอาหารใหม่ล่าสุด

```
<div className="lg:col-span-1 rounded-xl bg-BG flex flex-col items-center justify-center p-5">
  <h2 className="text-lg sm:text-3xl self-start ml-5">The latest new menu</h2>
  <div className="flex flex-col items-center justify-center h-full gap-5 mt-5">
    {latestFood ? (
      <div className="flex flex-col items-center justify-center h-full gap-5">
        <div className="text-xl font-bold">{latestFood.name}</div>
        <p className="text-md text-Text/50">{latestFood.description}</p>
        <img
          src={latestFood.image[0]}
          alt={latestFood.name}
          className="rounded-xl"
        />
      </div>
    ) : (
      <p className="text-lg text-Text/50">No menu available</p>
    )}
  </div>
</div>
```

ส่วนเสริม: การแสดงผลข้อมูลทางสถิติแบบ รายวัน รายเดือน และรายปี

```
<select
  value={dateRange}
  onChange={(e) => setDateRange(e.target.value)}
  className="border-2 p-2 rounded-md bg-BG text-Text">
  <option value="daily" >Daily</option>
  <option value="monthly" >Monthly</option>
  <option value="yearly" >Yearly</option>
</select>
```

ระบบจัดการข้อมูลพนักงาน

Employee.jsx

คำอธิบาย

หน้าแสดงรายการข้อมูลของพนักงานทั้งหมดภายในระบบ พร้อมฟังก์ชัน “เพิ่มข้อมูลพนักงาน” “แก้ไขข้อมูลพนักงาน” และ “ลบรายชื่อพนักงาน” ซึ่งสามารถเข้าถึงได้โดยแอดมิน

Employee				Action
Name	Email	Phone	Address	
Athip Yaungkaew	mr.athipyuangkaew@gmail.com	0863428756	187 สำเภาท่องสัง, สหโย, THA, 11000	
Theeratchanon Srithammayos	partycreepian@gmail.com	0863424837	18 Pracha Niwet 3 Soi 13/5, Bangkok, THA, 11000	
Phongphat Bangkha	Phongphat@gmail.com	0987654321	978 นางกอกคลาง, กรุงเทพ, THA, 11700	
Max Kun	max_hunter@hotmail.com	917555367664	67 Nan'an Rd, 广东省, 荔湾区, Germany, 020	
Jordi Nim	fddhdfhdfh@gmail.com	0222221444	Dark continent, Yorknew, USA, 123456	

โค้ดสำคัญ

ส่วนเสริม: การตรวจสอบว่าพนักงานเป็นแอดมิน

```
if (role !== "admin") {
  return (
    <div className="flex justify-center items-center mt-6">
      <div className="bg-red-100 text-red-700 border border-red-500 rounded-lg px-6 py-4">
        <p className="text-center font-semibold">Access Denied</p>
        <p className="text-center text-sm">Admins only.</p>
      </div>
    </div>
  );
}
```

ส่วนเสริม: ดึงข้อมูลพนักงานทุกคนมาจาก DashboardContext.jsx

```
const {employeeList, fetchEmployee} = useContext(DashboardContext);
```

ส่วนที่ 1: การแสดงรายการพนักงานภายในระบบทั้งหมด

```

/* Wrap the content inside a scrollable container */
<div className="overflow-x-auto">
  <table className="w-full table-auto">
    <thead>
      <tr className="uppercase text-left text-Text border-b border-Text/10">
        <th className="py-3 px-4">Name</th>
        <th className="py-3 px-4">Email</th>
        <th className="py-3 px-4">Phone</th>
        <th className="py-3 px-4">Address</th>
        <th className="py-3 px-4">Action</th>
      </tr>
    </thead>

    {/* Employee List */}
    <tbody>
      {employeeList.map((employee, index) => (
        <tr
          className="border-b border-Text/10 text-Text/80"
          key={index}
        >
          {/* Profile Pic & Name */}
          <td className="py-3 px-4 min-w-52">
            <div className="flex items-center gap-3">
              <img
                src={employee.profilePic}
                alt="profile"
                className="w-12 h-12 rounded-2xl"
              />
              <p>
                {employee.firstName} {employee.lastName}
              </p>
            </div>
          </td>
          {/* Email */}
          <td className="py-3 px-4">{employee.email}</td>
          {/* Phone */}
          <td className="py-3 px-4">{employee.phone}</td>
          {/* Address */}
          <td className="py-3 px-4 min-w-60">{employee.address}</td>
        </tr>
      )))
    </tbody>
  </table>
</div>

```

ส่วนที่ 2: การแสดงปุ่มเพิ่มข้อมูลพนักงาน (เชื่อมต่อกับ AddEmployee.jsx)

```
<div className="flex flex-col sm:flex-row gap-4 items-center justify-between w-full">
  <Link
    to="/add_employee"
    className="flex items-center justify-center self-start w-full sm:w-auto h-full p-4 gap-3 text-white hover:bg-green-400 bg-green-600 rounded-2xl">
    <UserPlus />
    <p>Add new</p>
  </Link>
</div>
```

ส่วนที่ 3: การแสดงปุ่มแก้ไขข้อมูลพนักงาน (เชื่อมต่อกับ EditEmployee.jsx)

```
<Link to={`/edit_employee/${employee._id}`}>
  <Pencil className="cursor-pointer" />
</Link>
```

ส่วนที่ 4: การแสดงปุ่มลบรายชื่อพนักงาน

```
const removeEmployee = async (id) => {
  try {
    const response = await axios.post(backendURL + "/api/employee/removeStaff", { id }, { headers: { token } });
    if (response.data.success) {
      toast.success(response.data.message);
      await fetchEmployee();
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};
```

```
<Trash className="cursor-pointer" onClick={() => removeEmployee(employee._id)} />
```

AddEmployee.jsx

คำอธิบาย

หน้าเพิ่มรายชื่อพนักงานใหม่เข้าสู่ระบบ โดยมีฟอร์มสำหรับกรอก ข้อมูลพื้นฐาน ข้อมูลที่อยู่ และ อัปโหลดรูปโปรไฟล์ ของพนักงาน เมื่อกดปุ่มเพิ่มพนักงาน ระบบจะส่งฟอร์มคำขอไปยัง API เพื่อทำการลงทะเบียนพนักงานใหม่ และจะแสดงข้อความแจ้งเตือนว่าสำเร็จหรือไม่ข้อผิดพลาด

The screenshot shows a dark-themed user interface for adding an employee. The form is organized into four main sections:

- Overview (Section 1):** Contains fields for First Name, Last Name, Email, Password, and Phone Number.
- Address Information (Section 2):** Contains dropdowns for Country (Thailand) and Address, and input fields for City and Postal Code.
- Profile Image (Section 3):** Displays a placeholder icon of a person inside a square frame.
- Action (Section 4):** A large yellow button labeled "Add Employee".

โค้ดสำคัญ

ส่วนเสริม: การตรวจสอบว่าพนักงานเป็นแอดมิน

```
if (role !== "admin") {
  return (
    <div className="flex justify-center items-center mt-6">
      <div className="bg-red-100 text-red-700 border border-red-500 rounded-lg px-6 py-4">
        <p className="text-center font-semibold">Access Denied</p>
        <p className="text-center text-sm">Admins only.</p>
      </div>
    </div>
  );
}
```

ส่วนเสริม: กำหนดค่า state สำหรับชนิดข้อมูลที่ต้องกรอกในฟอร์ม

```
const {fetchEmployee} = useContext(DashboardContext);

const [image, setImage] = useState(false);
const [firstname, setFirstname] = useState('');
const [lastname, setLastname] = useState('');
const [email, setEmail] = useState('');
const [password, setPassword] = useState('');
const [phone, setPhone] = useState('');
const [country, setCountry] = useState('THA');
const [address, setAddress] = useState('');
const [city, setCity] = useState('');
const [postalcode, setPostalCode] = useState('');
```

ส่วนที่ 1: การแสดงฟอร์มสำหรับกรอกข้อมูลพื้นฐานของพนักงาน

```
<div className='rounded-lxl bg-gray flex flex-col items-center justify-center p-5'>
  <h1 className='text-2xl font-bold self-start mb-5'>Overview</h1>
  <div className='flex gap-4 w-full mb-4'>
    <div className='w-full'>
      <p>First Name</p>
      <input onChange={(e) => setFirstname(e.target.value)} value={firstname} type="text" className="w-full rounded-lg p-2 bg-Text/20 placeholder-Text/50" placeholder='First Name' required />
    </div>
    <div className='w-full'>
      <p>Last Name</p>
      <input onChange={(e) => setLastname(e.target.value)} value={lastname} type="text" className="w-full rounded-lg p-2 bg-Text/20 placeholder-Text/50" placeholder='Last Name' required />
    </div>
  </div>
  <div className='flex gap-4 w-full mb-4'>
    <div className='w-full mb-2'>
      <p>Email</p>
      <input onChange={(e) => setEmail(e.target.value)} value={email} type="email" className="w-full rounded-lg p-2 bg-Text/20 placeholder-Text/50" placeholder='Email' required />
    </div>
    <div className='w-full'>
      <p>Password</p>
      <input onChange={(e) => setPassword(e.target.value)} value={password} type="password" className="w-full rounded-lg p-2 bg-Text/20 placeholder-Text/50" placeholder='Password' required />
    </div>
  </div>
  <div className='w-full mb-4'>
    <p>Phone Number</p>
    <input onChange={(e) => setPhone(e.target.value)} value={phone} type="text" className="w-full rounded-lg p-2 bg-Text/20 placeholder-Text/50" placeholder='Phone Number' required />
  </div>
</div>
```

ส่วนที่ 2: การแสดงฟอร์มสำหรับกรอกข้อมูลที่อยู่ของพนักงาน

ส่วนที่ 3: การแสดงปุ่มอัปโหลดรูปโปรไฟล์ของพนักงาน

```
<div className="rounded-xl bg-BG flex flex-col items-center justify-center p-5">
  <h1 className="text-2xl font-bold self-start mb-5">Profile Image</h1>
  <label htmlFor="image1" className='w-full h-full'>
    {!image ? (
      // Render your icon if there's no uploaded image
      <SquareUserRound className="w-full h-full cursor-pointer rounded-xl text-Text" />
    ) : (
      // Otherwise, display the uploaded image preview
      <img
        className="w-full rounded-2xl cursor-pointer"
        src={URL.createObjectURL(image)}
        alt="Uploaded"
      />
    )}
    <input
      onChange={(e) => setImage(e.target.files[0])}
      type="file"
      id="image1"
      hidden
    />
  </label>
</div>
```

ส่วนที่ 4: การแสดงปุ่มเพิ่มพนักงานเข้าไปยังระบบ

```
<button
  type="submit"
  className="rounded-xl bg-Button flex flex-col items-center justify-center p-5 text-3xl md:text-4xl xl:text-5xl text-BG active:bg-Button/75 hover:bg-Button/90"
>
  Add Employee
</button>
```

ส่วนเสริม: การตรวจสอบว่าข้อมูลที่กรอกมีความถูกต้อง ถ้าใช่ ระบบจะเพิ่มข้อมูลไปยังฐานข้อมูล แต่ถ้าไม่ ระบบจะทำการแจ้งเตือนว่ามีข้อผิดพลาดเกิดขึ้น

```
const onSubmitHandler = async (e) => {
  e.preventDefault();
  try {
    const formData = new FormData();
    const Fulladdress = `${address}, ${city}, ${country}, ${postalCode}`;

    formData.append('firstname', firstname);
    formData.append('lastname', lastname);
    formData.append('email', email);
    formData.append('password', password);
    formData.append('phone', phone);
    formData.append('address', Fulladdress);

    image && formData.append('profilePic', image);

    const response = await axios.post(backendURL + '/api/employee/registerStaff', formData, { headers: { token } });
    console.log(response)
    if (response.data.success) {
      fetchEmployee();
      toast.success(response.data.message);
      setFirstname('');
      setLastname('');
      setEmail('');
      setPassword('');
      setPhone('');
      setCountry('');
      setAddress('');
      setCity('');
      setPostalCode('');
      setImage(false);
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
}
```

EditEmployee.jsx

คำอธิบาย

หน้าแก้ไขข้อมูลของพนักงานที่มีภายในระบบ โดยจะดึงข้อมูลของพนักงานจากฐานข้อมูล และเติมข้อมูลลงไปในฟอร์ม ซึ่งสามารถแก้ไข ข้อมูลพื้นฐาน ข้อมูลที่อยู่ และ อัปเดตรูปโปรไฟล์ เมื่อกดปุ่มอัปเดตข้อมูลพนักงาน ระบบจะส่งฟอร์มคำขอไปยัง API เพื่อทำการแก้ไขข้อมูลพนักงาน และจะแสดงข้อความแจ้งเตือนว่าสำเร็จหรือมีข้อผิดพลาด

The screenshot shows a dark-themed user interface for editing employee information. The main title is "Edit Employee".

- Overview (Section 1):** Contains fields for First Name (Theeratchanon), Last Name (Sritthamayos), Email (partycreepian@gmail.com), and Phone Number (0863424837). It also includes a placeholder for New Password (Optional) and a note to leave blank to keep current password.
- Address Information (Section 2):** Contains fields for Country (Thailand), Address (18 Pracha Niwet 3 Soi 13/5), City (Bangkok), and Postal Code (11000).
- Profile Picture (Section 3):** Displays a placeholder image of a white mask.
- Update Employee (Section 4):** A large yellow button with the text "Update Employee" and a red number 4.

โค้ดสำคัญ

ส่วนเสริม: การตรวจสอบว่าพนักงานเป็นแอดมิน

```
if (role !== "admin") {
  return (
    <div className="flex justify-center items-center mt-6">
      <div className="bg-red-100 text-red-700 border border-red-500 rounded-lg px-6 py-4">
        <p className="text-center font-semibold">Access Denied</p>
        <p className="text-center text-sm">Admins only.</p>
      </div>
    </div>
  );
}
```

ส่วนเสริม: การดึงข้อมูลพนักงานจากฐานข้อมูล และเติมลงในฟอร์ม

```
const fetchEmployee = async () => {
  try {
    const response = await axios.post(backendURL + '/api/employee/singleStaff', { staffId });

    if (response.data.success) {
      const staff = response.data.staff;
      const addressParts = staff.address ? staff.address.split(' ', ') : ['', '', '', ''];
      setFirstname(staff.firstName || '');
      setLastname(staff.lastName || '');
      setEmail(staff.email || '');
      setPhone(staff.phone || '');
      setCountry(addressParts[2] || 'THA');
      setAddress(addressParts[0] || '');
      setCity(addressParts[1] || '');
      setPostalCode(addressParts[3] || '');
      setProfilePic(staff.profilePic || '');
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};

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

ส่วนที่ 1: การแสดงฟอร์มสำหรับกรอกข้อมูลพื้นฐานของพนักงาน

```
<div className="rounded-xl bg-BG flex flex-col items-center justify-center p-5">
  <h1 className="text-2xl font-bold self-start mb-5">Overview</h1>
  <div className="flex gap-4 w-full mb-4">
    <div className="w-full">
      <p className="mb-2">First Name:</p>
      <input onChange={(e) => setFirstname(e.target.value)} value={firstname} type="text" className="w-full rounded-lg p-2 bg-Text/20" required />
    </div>
    <div className="w-full">
      <p className="mb-2">Last Name:</p>
      <input onChange={(e) => setlastname(e.target.value)} value={lastname} type="text" className="w-full rounded-lg p-2 bg-Text/20" required />
    </div>
  </div>
  <div className="flex gap-4 w-full mb-4">
    <div className="w-full">
      <p className="mb-2">Email:</p>
      <input onChange={(e) => setEmail(e.target.value)} value={email} type="email" className="w-full rounded-lg p-2 bg-Text/20" required />
    </div>
    <div className="w-full">
      <p className="mb-2">New Password (Optional):</p>
      <input onChange={(e) => setPassword(e.target.value)} value={password} type="password" className="w-full rounded-lg p-2 bg-Text/20" placeholder="Leave blank to keep current password" />
    </div>
  </div>
  <div className="w-full mb-4">
    <p className="mb-2">Phone Number:</p>
    <input onChange={(e) => setphone(e.target.value)} value={phone} type="text" className="w-full rounded-lg p-2 bg-Text/20" required />
  </div>
</div>
```

ส่วนที่ 2: การแสดงฟอร์มสำหรับกรอกข้อมูลที่อยู่ของพนักงาน

```
<div className="rounded-xl bg-BG flex flex-col items-center justify-center p-5">
  <h1 className="text-2xl font-bold self-start mb-5">Address Information</h1>
  <div className="w-full mb-4">
    <p className="mb-2">Country:</p>
    <select onChange={(e) => setCountry(e.target.value)} value={country} className="w-full rounded-lg p-2 bg-Text/20">
      <option value="THA">Thailand</option>
      <option value="USA">United States</option>
      <option value="UK">United Kingdom</option>
      <option value="Canada">Canada</option>
      <option value="Australia">Australia</option>
      <option value="Germany">Germany</option>
    </select>
  </div>
  <div className="w-full mb-4">
    <p className="mb-2">Address:</p>
    <input onChange={(e) => setAddress(e.target.value)} value={address} type="text" className="w-full rounded-lg p-2 bg-Text/20" required />
  </div>
  <div className="flex gap-4 w-full mb-4">
    <div className="w-full">
      <p className="mb-2">City:</p>
      <input onChange={(e) => setCity(e.target.value)} value={city} type="text" className="w-full rounded-lg p-2 bg-Text/20" required />
    </div>
    <div className="w-full">
      <p className="mb-2">Postal Code:</p>
      <input onChange={(e) => setPostalCode(e.target.value)} value={postalcode} type="text" className="w-full rounded-lg p-2 bg-Text/20" required />
    </div>
  </div>
</div>
```

ส่วนที่ 3: การแสดงปุ่มอัปโหลดรูปโปรไฟล์ของพนักงาน

```
<div className="rounded-xl bg-BG flex flex-col items-center justify-center p-5">
    <h1 className="text-2xl font-bold self-start mb-5">Profile Picture</h1>
    <label htmlFor="image1">
        <img className="w-full rounded-2xl cursor-pointer" src={image ? URL.createObjectURL(image) : profilePic || assets.upload} alt="" />
        <input onChange={(e) => setImage(e.target.files[0])} type="file" id="image1" hidden />
    </label>
</div>
```

ส่วนที่ 4: การแสดงปุ่มอัปเดตข้อมูลพนักงาน

```
<button type="submit" className="rounded-xl bg-Button p-5 text-3xl text-BG hover:bg-Button/90">Update Employee</button>
```

ส่วนเสริม: การตรวจสอบว่าข้อมูลที่แก้ไขมีความถูกต้อง ถ้าใช่ ระบบจะส่งข้อมูลไปยังฐานข้อมูล แต่ถ้าไม่ ระบบจะทำการแจ้งเตือนว่ามีข้อผิดพลาดเกิดขึ้น

```
const onSubmitHandler = async (e) => {
    e.preventDefault();
    try {
        const formData = new FormData();
        const Fulladdress = `${address}, ${city}, ${country}, ${postalCode}`;

        formData.append('staffId', staffId);
        formData.append('firstName', firstname);
        formData.append('lastName', lastname);
        formData.append('email', email);
        if (password) formData.append('password', password);
        formData.append('phone', phone);
        formData.append('address', Fulladdress);

        image && formData.append('profilePic', image);

        const response = await axios.post(backendURL + '/api/employee/updateStaff', formData, {headers: {token}});

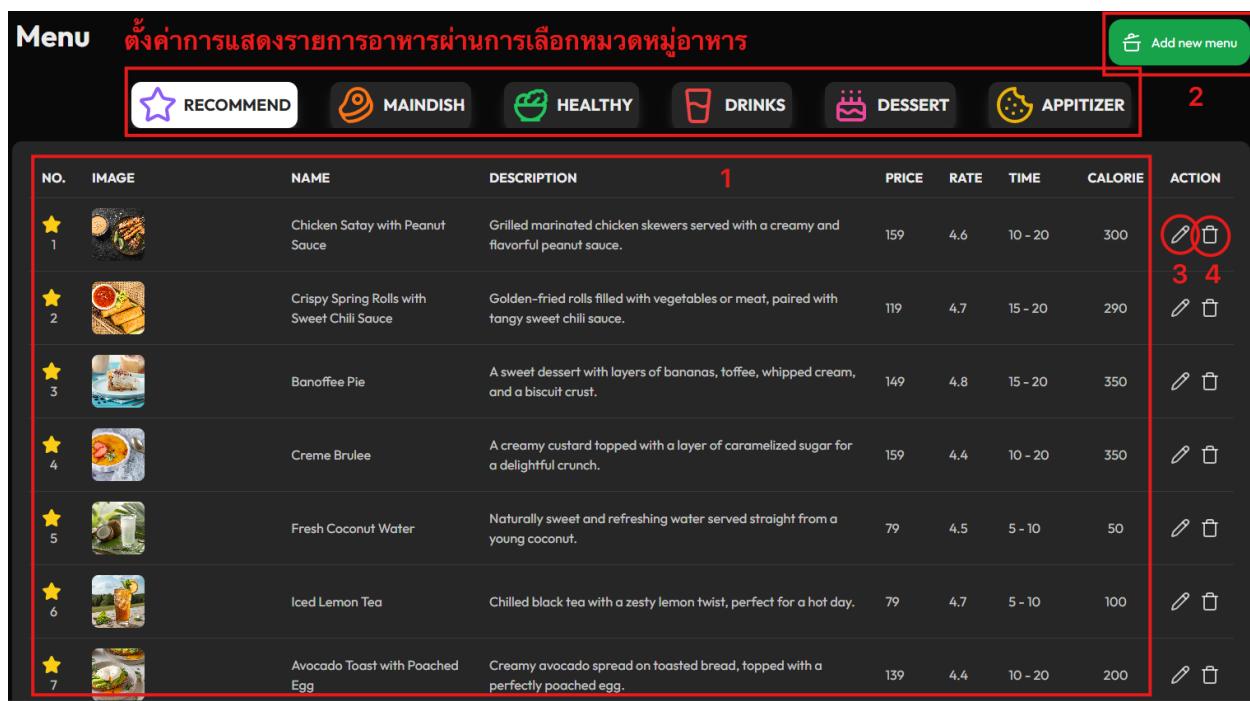
        if (response.data.success) {
            toast.success(response.data.message);
            navigate('/list_employee')
        } else {
            toast.error(response.data.message);
        }
    } catch (error) {
        console.log(error);
        toast.error(error.message);
    }
};
```

ระบบจัดการรายการอาหาร

Menu.jsx

คำอธิบาย

หน้าแสดงรายการอาหารในเมนูทั้งหมดภายในระบบ ซึ่งสามารถเลือกค่าได้ตามการคัดกรองหมวดหมู่อาหาร พร้อมฟังก์ชัน “เพิ่มเมนูอาหาร” “แก้ไขเมนูอาหาร” และ “ลบเมนูอาหาร” ซึ่งสามารถเข้าถึงได้โดยแอดมิน



The screenshot shows a table of food items with the following columns: NO., IMAGE, NAME, DESCRIPTION, 1, PRICE, RATE, TIME, CALORIE, and ACTION. The ACTION column contains edit and delete icons. A red box highlights the first seven rows, and a red circle with numbers 3 and 4 is placed over the edit and delete icons of the second item.

NO.	IMAGE	NAME	DESCRIPTION	1	PRICE	RATE	TIME	CALORIE	ACTION
1		Chicken Satay with Peanut Sauce	Grilled marinated chicken skewers served with a creamy and flavorful peanut sauce.	159	4.6	10 - 20	300		
2		Crispy Spring Rolls with Sweet Chili Sauce	Golden-fried rolls filled with vegetables or meat, paired with tangy sweet chili sauce.	119	4.7	15 - 20	290		
3		Banoffee Pie	A sweet dessert with layers of bananas, toffee, whipped cream, and a biscuit crust.	149	4.8	15 - 20	350		
4		Creme Brulee	A creamy custard topped with a layer of caramelized sugar for a delightful crunch.	159	4.4	10 - 20	350		
5		Fresh Coconut Water	Naturally sweet and refreshing water served straight from a young coconut.	79	4.5	5 - 10	50		
6		Iced Lemon Tea	Chilled black tea with a zesty lemon twist, perfect for a hot day.	79	4.7	5 - 10	100		
7		Avocado Toast with Poached Egg	Creamy avocado spread on toasted bread, topped with a perfectly poached egg.	139	4.4	10 - 20	200		

โค้ดสำคัญ

ส่วนเสริม: การคัดกรองรายการอาหารผ่านการเลือกหมวดหมู่อาหาร

```
const filteredFoodList = category === 'All'
  ? foodList
  : category === 'Recommend'
    ? foodList.filter(food => food.recommend)
    : foodList.filter(food => food.category === category);
```

```
{/* Scrollable Menu */}
<div className="flex w-full overflow-x-auto mt-5">
  <div className="flex gap-6 sm:gap-8 lg:gap-10 mx-auto max-w-screen-xl">
    {menu_list.map((item, index) => (
      <div
        key={index}
        onClick={() => setCategory(prev => prev === item.menu_name ? 'All' : item.menu_name)}
        className={`flex items-center gap-2 bg-BG px-2 py-1 rounded-xl shadow-lg shadow-Text/20 transition-all ease-in-out cursor-pointer ${category === item.menu_name ? 'bg-BG_Black text-BG' : 'text-Text'}`}
      >
      {/* Icon Section */}
      <div className="hover:scale-105 relative w-12 h-12 overflow-hidden rounded-full shrink-0">
        {React.createElement(item.menu_image, { className: `w-full h-full object-center ${item.color}` })}
      </div>
      {/* Text Section */}
      <div className="flex flex-col">
        <p className="text-md sm:lg md:text-xl font-bold uppercase">{item.menu_name}</p>
      </div>
    ))}
  </div>
</div>
```

ส่วนที่ 1: การแสดงรายการอาหารภายในระบบห้องครัว

```

/* Food List */


<div className="overflow-x-auto">
    <table className="w-full table-auto">
      <thead>
        <tr className="uppercase text-left text-Text border-b border-Text/10">
          <th className="py-3 px-4">No.</th>
          <th className="py-3 px-4">Image</th>
          <th className="py-3 px-4">Name</th>
          <th className="py-3 px-4">Description</th>
          <th className="py-3 px-4">Price</th>
          <th className="py-3 px-4">Rate</th>
          <th className="py-3 px-4">Time</th>
          <th className="py-3 px-4">Calorie</th>
          <th className="py-3 px-4">Action</th>
        </tr>
      </thead>

      /* Filtered Food List */
      <tbody>
        {filteredFoodList.map((food, index) => (
          <tr className="border-b border-Text/10 text-Text/80" key={index}>
            {/* ID */}
            <td className="py-3 px-4 text-center">
              {food.recommend ? <Star className="fill-yellow-400 text-yellow-400" /> : ' '}</div>
              <div>{index + 1}</div>
            </td>

            {/* Image */}
            <td className="py-3 px-4 min-w-60 flex">
              {food.image && food.image.length > 0 ? (
                food.image.map((image, imgIndex) => (
                  <img
                    key={imgIndex}
                    src={image}
                    alt={`Food Image ${imgIndex}`}
                    className="size-16 rounded-lg object-cover mr-2"
                  />
                )))
              : (
                <span>No images available</span>
              )}
            </td>

            {/* Name */}
            <td className="py-3 px-4 min-w-52">{food.name}</td>

            {/* Description */}
            <td className="py-3 px-4 min-w-72">{food.description}</td>

            {/* Price */}
            <td className="py-3 px-4">{food.price}</td>

            {/* Rate */}
            <td className="py-3 px-4">{food.rate}</td>

            {/* Time */}
            <td className="py-3 px-4 min-w-24">{food.time[0]} - {food.time[1]}</td>

            {/* Kcal */}
            <td className="py-3 px-4 text-center">{food.Kcal}</td>

            {/* Actions */}
            <td className="py-3 px-4">
              <div className="flex items-center gap-3">
                <Link to={`/edit_menu/${food._id}`}>
                  <Pencil className="cursor-pointer" />
                </Link>
                <Trash onClick={() => removeFood(food._id)} className="cursor-pointer" />
              </div>
            </td>
          </tr>
        ))}
      </tbody>
    </table>

    /* If no food matches the category */
    {filteredFoodList.length === 0 && (
      <p className="text-center text-gray-500 mt-4">No items available in this category.</p>
    )}
  </div>
</div>


```

ส่วนที่ 2: การแสดงปุ่มเพิ่มเมนูอาหาร (เชื่อมต่อกับ CreateMenu.jsx)

```
<Link to="/create" className="flex items-center justify-center self-start w-full sm:w-auto h-full p-4 gap-3 text-white hover:bg-green-400 bg-green-600 rounded-2xl">
  <CookingPot />
  <p>Add new menu</p>
</Link>
```

ส่วนที่ 3: การแสดงปุ่มแก้ไขเมนูอาหาร (เชื่อมต่อกับ EditMenu.jsx)

```
<Link to={`/edit_menu/${food._id}`}>
  <Pencil className="cursor-pointer" />
</Link>
```

ส่วนที่ 4: การแสดงปุ่มลบเมนูอาหาร

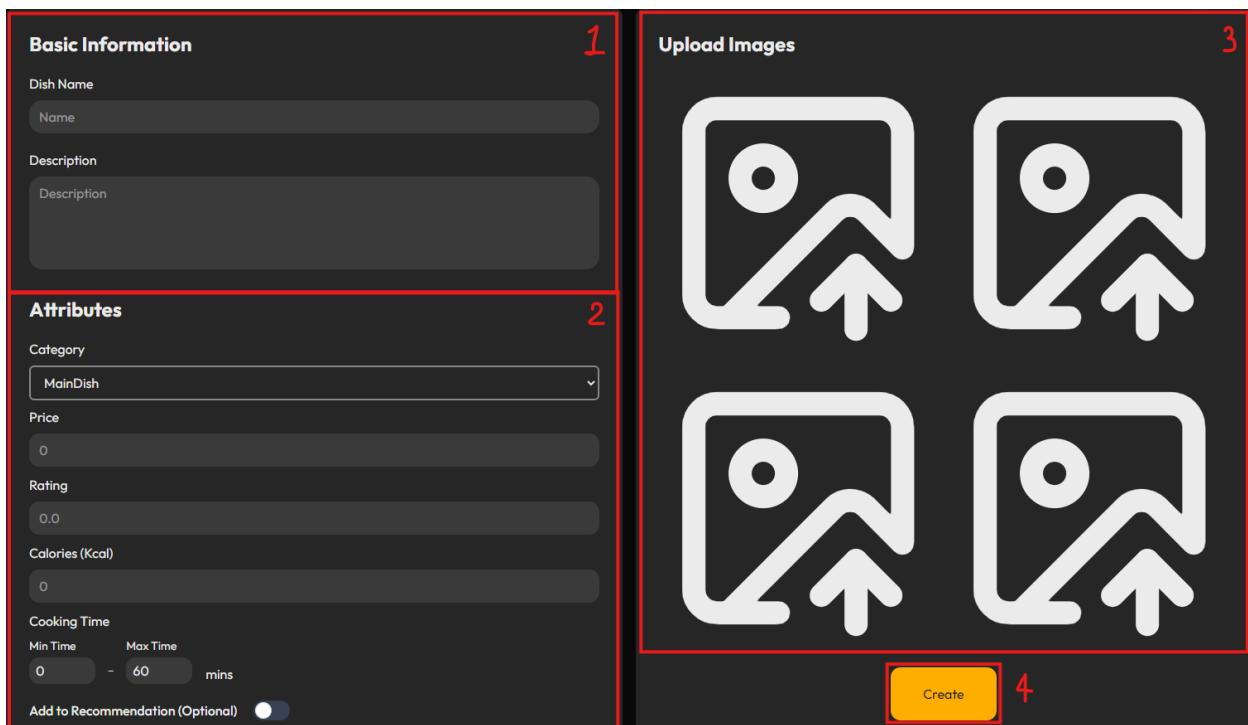
```
const removeFood = async (id) => {
  try {
    const response = await axios.post(backendURL + '/api/product/remove', { id }, { headers: { token } });
    if (response.data.success) {
      toast.success(response.data.message);
      await fetchFood();
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
}
```

```
<Trash onClick={() => removeFood(food._id)} className="cursor-pointer" />
```

CreateMenu.jsx

คำอธิบาย

หน้าเพิ่มรายการอาหารใหม่เข้าสู่ระบบ โดยมีฟอร์มสำหรับกรอก ข้อมูลพื้นฐาน ข้อมูลคุณลักษณะ และ อัปโหลดรูปภาพ ของอาหาร เมื่อกรอกปุ่มเพิ่มเมนู ระบบจะส่งฟอร์มคำขอไปยัง API เพื่อทำการเพิ่มเมนูอาหารใหม่ และจะแสดงข้อความแจ้งเตือนว่าสำเร็จหรือมีข้อผิดพลาด



โค้ดสำคัญ

ส่วนเสริม: การตรวจสอบว่าพนักงานเป็นแอดมิน

```
if (role !== "admin") {
  return (
    <div className="flex justify-center items-center mt-6">
      <div className="bg-red-100 text-red-700 border border-red-500 rounded-lg px-6 py-4">
        <p className="text-center font-semibold">Access Denied</p>
        <p className="text-center text-sm">Admins only.</p>
      </div>
    </div>
  );
}
```

ส่วนเสริม: กำหนดค่า state สำหรับชนิดข้อมูลที่ต้องกรอกในฟอร์ม

```
const {fetchFood} = useContext(DashboardContext);
const [image1, setImage1] = useState(false);
const [image2, setImage2] = useState(false);
const [image3, setImage3] = useState(false);
const [image4, setImage4] = useState(false);

const [name, setName] = useState('');
const [description, setDescription] = useState('');
const [price, setPrice] = useState('');
const [rate, setRate] = useState('');
const [calories, setCalories] = useState('');
const [category, setCategory] = useState('MainDish');
const [bestseller, setBestseller] = useState(false);
const [timeRange, setTimeRange] = useState([0, 60]);
```

ส่วนที่ 1: การแสดงฟอร์มสำหรับกรอกข้อมูลพื้นฐานของอาหาร

```
<h1 className="text-2xl font-bold mb-5 text-Text">Basic Information</h1>
{/* Basic Info Section */}
<div>
  {/* Dish Name */}
  <div className="flex flex-col w-full mb-5">
    <p className="mb-2 text-Text">Dish Name</p>
    <input
      onChange={(e) => setName(e.target.value)}
      value={name}
      className="w-full px-3 py-2 rounded-2xl text-Text bg-Text/10 placeholder-Text/50"
      type="text"
      placeholder="Name"
      required
    />
  </div>

  {/* Dish Description */}
  <div className="flex flex-col w-full mb-7">
    <p className="mb-2 text-Text">Description</p>
    <textarea
      onChange={(e) => setDescription(e.target.value)}
      value={description}
      className="w-full h-28 px-3 py-2 mb-1 rounded-2xl text-Text bg-Text/10 placeholder-Text/50 resize-none"
      placeholder="Description"
      required
    />
  </div>
</div>
```

ส่วนที่ 2: การแสดงฟอร์มสำหรับกรอกข้อมูลคุณลักษณะของอาหาร

```

/* Attributes Section */
<h1 className="text-2xl font-bold mb-5 text-Text">Attributes</h1>
<div className="grid grid-cols-4 gap-1 w-full">
  /* Category */
  <div>
    <p className="flex w-full text-Text mb-2">Category</p>
    <select
      onChange={(e) => setCategory(e.target.value)}
      value={category}
      className="w-full px-3 py-2 placeholder-Text/50 border-2 border-Text/50 p-2 rounded-md bg-Text text-Text"
    >
      <option value="MainDish">MainDish</option>
      <option value="Healthy">Healthy</option>
      <option value="Drinks">Drinks</option>
      <option value="Dessert">Dessert</option>
      <option value="Appitizer">Appitizer</option>
    </select>
  </div>

  /* Price */
  <div>
    <p className="flex-2 w-full text-Text">Price</p>
    <input
      onChange={(e) => setPrice(e.target.value)}
      value={price}
      min={0}
      className="w-full px-3 py-2 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
      type="number"
      placeholder="0"
      step=".01"
    />
  </div>

  /* Rating */
  <div>
    <p className="mb-2 w-full text-Text">Rating</p>
    <input
      onChange={(e) => setRate(e.target.value)}
      value={rate}
      min={0}
      className="w-full px-3 py-2 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
      type="Number"
      placeholder="0.0"
      step={0.1}
    />
  </div>

  /* Calories */
  <div>
    <p className="mb-2 w-full text-Text">Calories (Kcal)</p>
    <input
      onChange={(e) => setCalories(e.target.value)}
      value={calories}
      min={0}
      className="w-full px-3 py-2 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
      type="Number"
      placeholder="0"
    />
  </div>

  /* Cooking Time & Recommendation in one row */
  <div className="grid grid-cols-1 items-center">
    /* Cooking Time */
    <div>
      <p className="mb-2 w-full text-Text">Cooking Time</p>
      <div className="flex gap-4 items-end">
        /* Min Time */
        <div className="flex flex-col">
          <label className="mb-1 text-sm text-Text">Min Time</label>
          <input
            type="number"
            value={timeRange[0]}
            min={0}
            placeholder="0"
            max={timeRange[1]}
            onChange={(e) =>
              setTimeRange([Number(e.target.value), timeRange[1]])
            }
            className="w-20 px-1 py-1 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
          />
        </div>
        <span className="text-Text">~</span>
        /* Max Time */
        <div className="flex flex-col">
          <label className="mb-1 text-sm text-Text">Max Time</label>
          <input
            type="number"
            value={timeRange[1]}
            min={timeRange[0]}
            placeholder="60"
            onChange={(e) =>
              setTimeRange([timeRange[0], Number(e.target.value)])
            }
            className="w-20 px-1 py-1 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
          />
        </div>
        <span className="text-Text">mins</span>
      </div>
    </div>

    /* Recommendation */
    <div className="mt-5">
      <label className="flex items-center italic font-medium">bestseller</label>
      <span className="mr-5 font-medium text-Text">Add to Recommendation (Optional)</span>
      <input
        onChange={() => setBestseller((prev) => !prev)}
        checked={bestseller}
        type="checkbox"
        id="Bestseller"
        className="sr-only peer"
      />
      <div
        className="cursor-pointer relative w-11 h-6 rounded-full peer
        dark:bg-gray-700 peer-checked:after:translate-x-full peer-checked:after:background-color-white
        after:content-['!'] after:absolute after:top-[2px] after:left-[2px]
        after:bg-white after:border-gray-300 after:border after:rounded-full
        after:h-5 after:w-5 after:transition-all peer-checked:bg-green-600
        dark:peer-checked:bg-green-600
      ">
        <div>
          <label>
            <input checked="" type="checkbox" id="Bestseller" />
          </label>
        </div>
      </div>
    </div>
  </div>

```

ส่วนที่ 3: การแสดงปุ่มอัปโหลดครุภาระของอาหาร

```

/* Main content area grows to fill space */
<div className="flex-grow w-full">
  {/* Row 1 of images */}
  <div className="grid grid-cols-2 gap-4 my-5">
    {/* Image 1 */}
    <label htmlFor="image1">
      {image1 ? (
        <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
      ) : (
        <img
          className="size-80 cursor-pointer rounded-xl"
          src={URL.createObjectURL(image1)}
          alt=""
        />
      )}
      <input
        onChange={(e) => setImage1(e.target.files[0])}
        type="file"
        id="image1"
        hidden
      />
    </label>

    {/* Image 2 */}
    <label htmlFor="image2">
      {!image2 ? (
        <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
      ) : (
        <img
          className="size-80 cursor-pointer rounded-xl"
          src={URL.createObjectURL(image2)}
          alt=""
        />
      )}
      <input
        onChange={(e) => setImage2(e.target.files[0])}
        type="file"
        id="image2"
        hidden
      />
    </label>
  </div>

  {/* Row 2 of images */}
  <div className="grid grid-cols-2 gap-4">
    {/* Image 3 */}
    <label htmlFor="image3">
      {image3 ? (
        <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
      ) : (
        <img
          className="size-80 cursor-pointer rounded-xl"
          src={URL.createObjectURL(image3)}
          alt=""
        />
      )}
      <input
        onChange={(e) => setImage3(e.target.files[0])}
        type="file"
        id="image3"
        hidden
      />
    </label>

    {/* Image 4 */}
    <label htmlFor="image4">
      {!image4 ? (
        <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
      ) : (
        <img
          className="size-80 cursor-pointer rounded-xl"
          src={URL.createObjectURL(image4)}
          alt=""
        />
      )}
      <input
        onChange={(e) => setImage4(e.target.files[0])}
        type="file"
        id="image4"
        hidden
      />
    </label>
  </div>
</div>

```

ส่วนที่ 4: การแสดงปุ่มเพิ่มเมนูอาหารเข้าไปยังระบบ

```
/* Button at bottom of right column */


<button
    type="submit"
    className="w-32 md:p-5 py-3 bg-Button text-BG active:bg-Button/75 rounded-xl shadow">
    Create
  </button>
</div>


```

ส่วนเสริม: การตรวจสอบว่าข้อมูลที่แก้ไขมีความถูกต้อง ถ้าใช่ ระบบจะส่งข้อมูลไปยังฐานข้อมูล แต่ถ้าไม่ ระบบจะทำการแจ้งเตือนว่ามีข้อผิดพลาดเกิดขึ้น

```
const onSubmitHandler = async (e) => {
  e.preventDefault();
  try {
    const formData = new FormData();

    formData.append('name', name);
    formData.append('description', description);
    formData.append('price', price);
    formData.append('rate', rate);
    formData.append('Kcal', calories);
    formData.append('category', category);
    formData.append('recommend', bestseller);
    formData.append('time', JSON.stringify(timeRange));

    image1 && formData.append('image1', image1);
    image2 && formData.append('image2', image2);
    image3 && formData.append('image3', image3);
    image4 && formData.append('image4', image4);

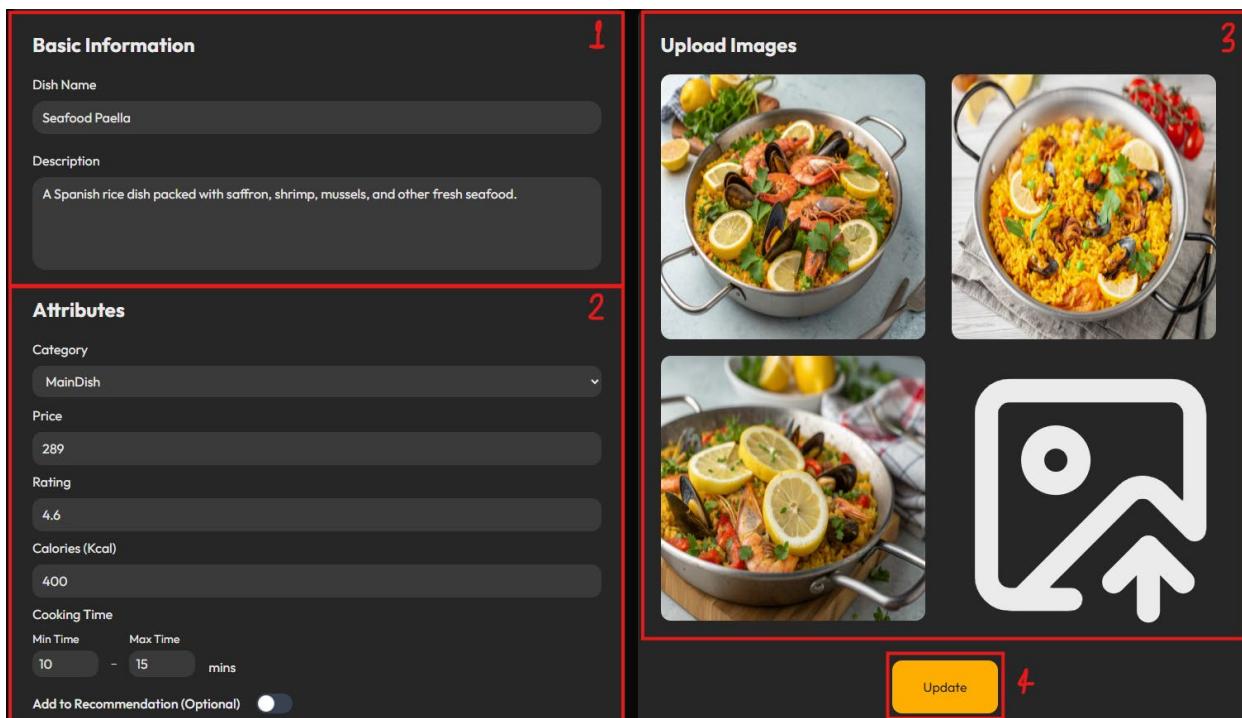
    const response = await axios.post(backendURL + '/api/product/add', formData, { headers: { token } });

    if (response.data.success) {
      toast.success(response.data.message);
      fetchFood();
      setImage1(false);
      setImage2(false);
      setImage3(false);
      setImage4(false);
      setName('');
      setDescription('');
      setPrice('');
      setRate('');
      setCalories('');
      setCategory("MainDish");
      setBestseller(false);
      setTimeRange([0, 60])
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
}
```

EditMenu.jsx

คำอธิบาย

หน้าแก้ไขข้อมูลของการอาหารที่มีภายในระบบ โดยจะดึงข้อมูลของการอาหารจากฐานข้อมูล และเติมข้อมูลลงไบในฟอร์ม ซึ่งสามารถแก้ไข ข้อมูลพื้นฐาน ข้อมูลคุณลักษณะ และอัปเดตรูปภาพอาหาร เมื่อกดปุ่มอัปเดตข้อมูลอาหาร ระบบจะส่งฟอร์มคำขอไปยัง API เพื่อทำการแก้ไขข้อมูลเมนูอาหาร และจะแสดงข้อความแจ้งเตือนว่าสำเร็จหรือมีข้อผิดพลาด



โค้ดสำคัญ

ส่วนเสริม: การตรวจสอบว่าพนักงานเป็นแอดมิน

```
if (role !== "admin") {
  return (
    <div className="flex justify-center items-center mt-6">
      <div className="bg-red-100 text-red-700 border border-red-500 rounded-lg px-6 py-4">
        <p className="text-center font-semibold">Access Denied</p>
        <p className="text-center text-sm">Admins only.</p>
      </div>
    </div>
  );
}
```

ส่วนเสริม: การดึงข้อมูลเมนูอาหารจากฐานข้อมูล และเตรียมลงไปในฟอร์ม

```
const fetchFoodSingle = async () => {
  try {
    const response = await axios.post(backendURL + '/api/product/single', { productId });
    if (response.data.success) {
      const product = response.data.product;

      setName(product.name || '');
      setDescription(product.description || '');
      setPrice(product.price || '');
      setRate(product.rate || '');
      setCalories(product.Kcal || '');
      setCategory(product.category || 'MainDish');
      setBestseller(product.recommend || false);
      setTimeRange(product.time || [0, 60]);

      // Existing DB images
      setFood1(product.image?.[0] || '');
      setFood2(product.image?.[1] || '');
      setFood3(product.image?.[2] || '');
      setFood4(product.image?.[3] || '');
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};

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

ส่วนที่ 1: การแสดงฟอร์มสำหรับกรอกข้อมูลพื้นฐานของเมนูอาหาร

```
<h1 className="text-2xl font-bold mb-5 text-Text">
  Basic Information
</h1>

{/* Basic Info Section */}
<div>
  {/* Dish Name */}
  <div className="flex flex-col w-full mb-5">
    <p className="mb-2 text-Text">Dish Name</p>
    <input
      onChange={(e) => setName(e.target.value)}
      value={name}
      className="w-full px-3 py-2 rounded-2xl text-Text bg-Text/10 placeholder-Text/50"
      type="text"
      placeholder="Name"
      required
    />
  </div>

  {/* Dish Description */}
  <div className="flex flex-col w-full mb-7">
    <p className="mb-2 text-Text">Description</p>
    <textarea
      onChange={(e) => setDescription(e.target.value)}
      value={description}
      className="w-full h-28 px-3 py-2 mb-1 rounded-2xl text-Text bg-Text/10 placeholder-Text/50 resize-none"
      placeholder="Description"
      required
    />
  </div>
</div>
```

ส่วนที่ 2: การแสดงฟอร์มสำหรับกรอกข้อมูลคุณลักษณะของเมนูอาหาร

```

/* Attributes Section */


<div className="text-2xl font-bold mb-5 text-Text">Attributes</div>
    <div className="grid grid-cols-4 gap-2 w-full">
        /* Category */
        <div>
            <p className="flex w-full text-Text mb-2">Category</p>
            <select
                onChange={(e) => setCategory(e.target.value)}
                value={category}
                className="w-full px-3 py-2 rounded-xl text-Text bg-Text/10 placeholder-Text/50"
            >
                <option value="MainDish">MainDish</option>
                <option value="Healthy">Healthy</option>
                <option value="Drinks">Drinks</option>
                <option value="Dessert">Dessert</option>
                <option value="Appitizer">Appitizer</option>
            </select>
        </div>

        /* Price */
        <div>
            <p className="flex mb-2 w-full text-Text">Price</p>
            <input
                onChange={(e) => setPrice(e.target.value)}
                value={price}
                min={0}
                className="w-full px-3 py-2 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
                type="number"
                placeholder="0"
                step="0.01"
            >
        </div>

        /* Rating */
        <div>
            <p className="mb-2 w-full text-Text">Rating:</p>
            <input
                onChange={(e) => setRate(e.target.value)}
                value={rate}
                min={0}
                className="w-full px-3 py-2 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
                type="Number"
                placeholder="0.0"
                step="0.1"
            >
        </div>

        /* Calories */
        <div>
            <p className="mb-2 w-full text-Text">Calories (Kcal)</p>
            <input
                onChange={(e) => setCalories(e.target.value)}
                value={calories}
                min={0}
                className="w-full px-3 py-2 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner"
                type="Number"
                placeholder="0"
            >
        </div>

        /* Cooking Time & Recommendation in one row */
        <div className="grid grid-cols-1 items-center">
            /* Cooking Time */
            <div>
                <p className="mb-2 w-full text-Text">Cooking Time:</p>
                <div className="flex gap-4 items-end">
                    /* Min Time */
                    <div className="flex flex-col">
                        <label className="mb-1 text-sm text-Text">Min Time:</label>
                        <input
                            type="number"
                            value={timeRange[0]}
                            min={0}
                            placeholder="0"
                            max={timeRange[1]}
                            onChange={(e) =>
                                setTimeRange([Number(e.target.value), timeRange[1]])
                            }
                        >
                    </div>
                    <span className="text-Text">-</span>
                    <span className="text-Text">px-2 py-1 rounded-xl text-Text bg-Text/10 placeholder-Text/50 no-spinner</span>
                </div>
            </div>
            /* Max Time */
            <div className="flex flex-col">
                <label className="mb-1 text-sm text-Text">Max Time:</label>
                <input
                    type="number"
                    value={timeRange[1]}
                    min={timeRange[0]}
                    placeholder="0"
                    onChange={(e) =>
                        setTimeRange([timeRange[0], Number(e.target.value)])
                    }
                >
            </div>
            <span className="text-Text">-</span>
            <span className="text-Text">mins</span>
        </div>
    </div>

    /* Recommendation */
    <div className="mt-5">
        <label className="flex items-center" htmlFor="bestseller">
            <span className="mr-5 font-medium text-Text">
                Add to Recommendation (Optional)
            </span>
            <input
                onChange={() => setBestseller((prev) => !prev)}
                checked={bestseller}
                type="checkbox"
                id="bestseller"
                className="sr-only peer"
            />
        <div>
            cursor-pointer relative w-11 h-4 rounded-full peer
            dark:bg-gray-700 peer-checked:after:translate-x-full
            &::before:peer-checked:after:-translate-x-full peer-checked:after:border-white
            after:content [""] after:absolute after:top-[1px] after:left-[2px]
            after:ring-white after:border-gray-300 after:border after:rounded-full
            after:w-5 after:h-5 after:transition-all peer-checked:bg-green-600
            &::after
            &::before
        </div>
    </label>
    </div>
</div>


```

ส่วนที่ 3: การแสดงปุ่มอัปโหลดครุภาระของเมนูอาหาร

```

/* Main content area grows to fill space */
<div className="flex-grow w-full">
  /* Row 1 of images */
  <div className="grid grid-cols-2 gap-4 my-5">
    /* Image 1 */
    <label htmlFor="image1">
      {image1 ? (
        <img
          className="size-80 cursor-pointer rounded-xl"
          src={URL.createObjectURL(image1)}
          alt=""
        />
      ) : food1 ? (
        <img
          className="size-80 cursor-pointer rounded-xl"
          src={food1}
          alt=""
        />
      ) : (
        <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
      )
    </input>
    onChange={(e) => setImage1(e.target.files[0])}
    type="file"
    id="image1"
    hidden
  />
  </label>

  /* Image 2 */
  <label htmlFor="image2">
    {image2 ? (
      <img
        className="size-80 cursor-pointer rounded-xl"
        src={URL.createObjectURL(image2)}
        alt=""
      />
    ) : food2 ? (
      <img
        className="size-80 cursor-pointer rounded-xl"
        src={food2}
        alt=""
      />
    ) : (
      <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
    )
  </input>
  onChange={(e) => setImage2(e.target.files[0])}
  type="file"
  id="image2"
  hidden
  />
  </label>
</div>

/* Row 2 of images */
<div className="grid grid-cols-2 gap-4">
  /* Image 3 */
  <label htmlFor="image3">
    {image3 ? (
      <img
        className="size-80 cursor-pointer rounded-xl"
        src={URL.createObjectURL(image3)}
        alt=""
      />
    ) : food3 ? (
      <img
        className="size-80 cursor-pointer rounded-xl"
        src={food3}
        alt=""
      />
    ) : (
      <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
    )
  </input>
  onChange={(e) => setImage3(e.target.files[0])}
  type="file"
  id="image3"
  hidden
  />
  </label>

  /* Image 4 */
  <label htmlFor="image4">
    {image4 ? (
      <img
        className="size-80 cursor-pointer rounded-xl"
        src={URL.createObjectURL(image4)}
        alt=""
      />
    ) : food4 ? (
      <img
        className="size-80 cursor-pointer rounded-xl"
        src={food4}
        alt=""
      />
    ) : (
      <ImageUp className="w-full h-full cursor-pointer rounded-xl text-Text" />
    )
  </input>
  onChange={(e) => setImage4(e.target.files[0])}
  type="file"
  id="image4"
  hidden
  />
  </label>
</div>
</div>

```

ส่วนที่ 4: การแสดงปุ่มอัปเดตข้อมูลเมนูอาหาร

```
<div className="flex justify-center mt-4">
  <button
    type="submit"
    className="w-32 md:p-5 py-3 bg-Button text-BG active:bg-Button/75 rounded-xl shadow">
    >
    Update
  </button>
</div>
```

ส่วนเสริม: การตรวจสอบว่าข้อมูลที่แก้ไขมีความถูกต้อง ถ้าใช่ ระบบจะส่งข้อมูลไปยังฐานข้อมูล แต่ถ้าไม่ ระบบจะทำการแจ้งเตือนว่ามีข้อผิดพลาดเกิดขึ้น

```
const onSubmitHandler = async (e) => {
  e.preventDefault();
  try {
    const formData = new FormData();

    formData.append('productId', productId);
    formData.append('name', name);
    formData.append('description', description);
    formData.append('price', price);
    formData.append('rate', rate);
    formData.append('Kcal', calories);
    formData.append('category', category);
    formData.append('recommend', bestseller);
    formData.append('time', JSON.stringify(timeRange));

    if (image1) formData.append('image1', image1);
    if (image2) formData.append('image2', image2);
    if (image3) formData.append('image3', image3);
    if (image4) formData.append('image4', image4);

    const response = await axios.post(
      backendURL + '/api/product/update',
      formData,
      { headers: { token } }
    );

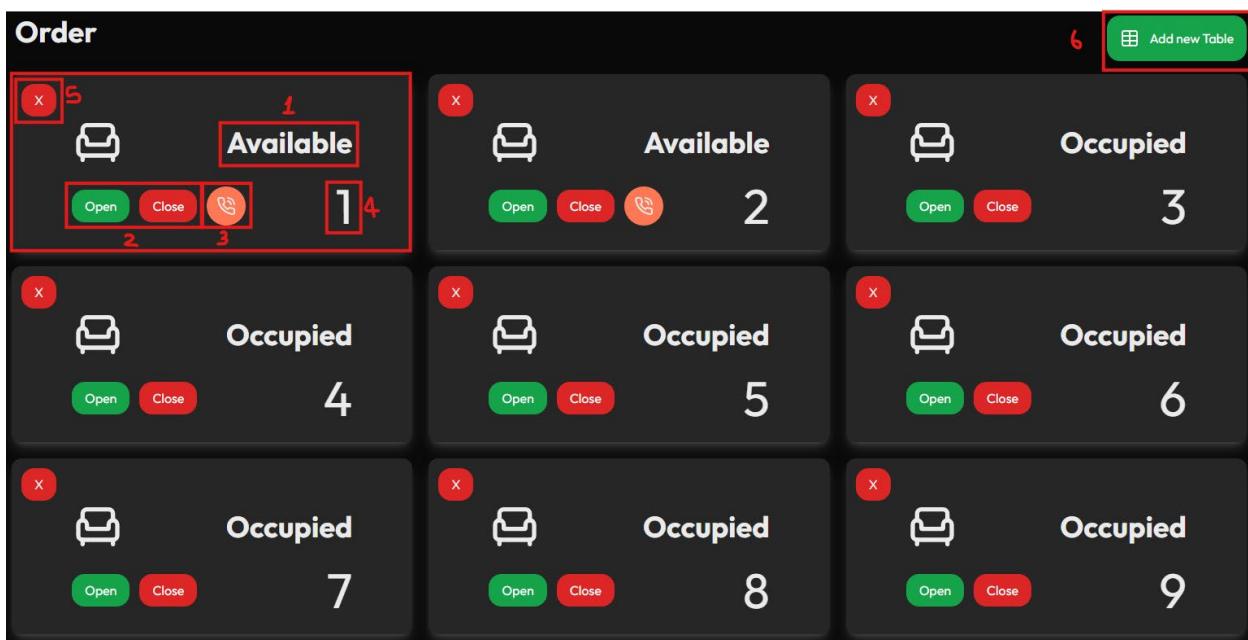
    if (response.data.success) {
      toast.success(response.data.message);
      fetchFood()
      navigate('/view_menu');
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};
```

ระบบจัดการออร์เดอร์

Order.jsx

คำอธิบาย

หน้าจัดการและแสดงข้อมูลของแต่ละโต๊ะภายในระบบ โดยแสดงเป็นแบบกริดที่แต่ละการ์ดแสดงหมายเลขโต๊ะ และสถานะของโต๊ะ รวมถึงมีปุ่มสำหรับการดำเนินการต่าง ๆ ได้แก่ เปิดโต๊ะ ปิดโต๊ะ ตอบรับคำขอความช่วยเหลือจากลูกค้า ลบโต๊ะ และเพิ่มโต๊ะ และยังสามารถดูรายละเอียดคำสั่งซื้อของลูกค้าเมื่อกดไปที่การ์ดของแต่ละโต๊ะได้



โค้ดสำคัญ

ส่วนที่ 1: การแสดงสถานะของโต๊ะ

```
<Link to={`/table/${table.table}`} className="flex w-full justify-between items-center">
  <Armchair className="size-16" />
  <h2 className="text-4xl font-bold">
    {table.available ? "Available" : "Occupied"}
  </h2>
</Link>
```

ส่วนที่ 2: การแสดงปุ่ม เปิด/ปิด การให้บริการของโต๊ะ

```
<div className="flex gap-3">
  <button
    className="flex items-center justify-center self-start w-full sm:w-auto px-4 py-2 gap-3
      text-white hover:bg-green-400 bg-green-600 rounded-2xl cursor-pointer"
    onClick={() => handleOpenTable(table.table)}
  >
    Open
  </button>
  <button
    className="flex items-center justify-center self-start w-full sm:w-auto px-4 py-2 gap-3
      text-white hover:bg-red-400 bg-red-600 rounded-2xl cursor-pointer"
    onClick={() => handleCloseTable(table.table)}
  >
    Close
  </button>
</div>
```

```
const handleOpenTable = async (number) => {
  try {
    const response = await axios.post(backendURL + "/api/table/available", { table: number, available: true, });
    if (response.data.success) {
      toast.success(response.data.message);
      await fetchTable();
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};

const handleCloseTable = async (number) => {
  try {
    const response = await axios.post(backendURL + '/api/table/clear', { table: number })
    if (response.data.success) {
      toast.success(response.data.message);
      await fetchTable();
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};
```

ส่วนที่ 3: การแสดงปุ่มตوبرับคำขอความช่วยเหลือจากลูกค้า

```
{table.callWaiter && (
  <div
    onClick={() => handleAttendTable(table.table)}
    className="p-3 bg-Highlight hover:bg-Highlight/50 rounded-full cursor-pointer flex items-center"
  >
    <PhoneCall />
  </div>
)}
```

```
const handleAttendTable = async (number) => {
  try {
    const response = await axios.post(backendURL + "/api/table/attend", {
      table: number,
    });
    if (response.data.success) {
      toast.success(response.data.message);
      await fetchTable();
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};
```

ส่วนที่ 4: การแสดงหมายเลขโต๊ะ

```
<h1 className="text-6xl">{table.table}</h1>
```

ส่วนที่ 5: การแสดงปุ่มลบໂຕ

```
<button onClick={() => handleDeleteClick(table.table)} className="flex items-center justify-center self-start px-4 py-2 gap-3 text-white hover:bg-red-400 bg-red-600 rounded-2xl cursor-pointer ml-3">
  X
</button>
```

```
const handleDeleteClick = (number) => {
  setTableToDelete(number);
  setDeleteModalOpen(true);
};

const confirmDeleteTable = async () => {
  try {
    const response = await axios.post(backendURL + "/api/table/delete", { tableNumber: tableToDelete, });
    if (response.data.success) {
      toast.success(response.data.message);
      await fetchTable();
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  } finally {
    setDeleteModalOpen(false);
    setTableToDelete(null);
  }
};

const cancelDelete = () => {
  setDeleteModalOpen(false);
  setTableToDelete(null);
};
```

ส่วนที่ 6: การแสดงปุ่มเพิ่มໂຕ

```
<div
  onClick={() => setIsModalOpen(true)}
  className="flex items-center justify-center self-start w-full sm:w-auto h-full p-4 gap-3 text-white hover:bg-green-400 bg-green-600 rounded-2xl cursor-pointer">
  <Table />
  <p>Add new Table</p>
</div>
```

```

const handleAddTable = async () => {
  if (!tableNumber) return;
  try {
    const response = await axios.post(backendURL + "/api/table/add", {
      table: tableNumber,
    });
    if (response.data.success) {
      toast.success(response.data.message);
      setIsModalOpen(false);
      setTableNumber("");
      await fetchTable();
    } else {
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
};

}

```

```

/* Modal for adding new table */
{isModalOpen && (
  <div className="fixed inset-0 bg-black bg-opacity-50 flex justify-center items-center">
    <div className="bg-Text p-6 rounded-lg shadow-lg w-96 text-center">
      <h2 className="text-xl font-bold mb-4 text-BG">Add New Table</h2>
      <input
        type="number"
        placeholder="Enter table number"
        value={tableNumber}
        onChange={(e) => setTableNumber(e.target.value)}
        className="w-full border p-2 rounded mb-4 text-BG"
      />
      <div className="flex justify-between">
        <button
          onClick={() => setIsModalOpen(false)}
          className="px-4 py-2 bg-red-700 rounded"
        >
          Cancel
        </button>
        <button
          onClick={handleAddTable}
          className="px-4 py-2 bg-green-600 text-white rounded"
        >
          Add Table
        </button>
      </div>
    </div>
  </div>
)
}
```

Table.jsx

คำอธิบาย

หน้าแสดงข้อมูลคำสั่งซื้ออย่างละเอียดสำหรับแต่ละโต๊ะ ซึ่งแสดงมีความสามารถอัปเดตสถานะของอาหาร ลบคำสั่งซื้อ

NO.	USER	TOTAL PRICE	TIME	AMOUNT	STATUS	SHOW DETAIL	
			3			5	
		Images	Product Name	Price	Quantity	Requirement	Sum Price
1	A	219	20:26:24	1	Ordering	4	฿219
			Pan-Seared Pork Chop with Apple Sauce	฿219	1	fbbfr	฿219

โค้ดสำคัญ

ส่วนที่ 1: การแสดงหมายเลขโต๊ะ

```
<div className="flex gap-3 md:gap-6">
  <Link
    to="/order"
    className="flex items-center text-Text hover:text-Highlight"
  >
    <SkipBack className="size-8 md:size-10" />
  </Link>
  <h1 className="text-2xl md:text-4xl font-bold ml-1">
    Table {tableNumber}
  </h1>
</div>
```

ส่วนที่ 2: การแสดงยอดรวมค่าใช้จ่ายของโต๊ะ

```
<h1 className="text-2xl md:text-4xl font-bold ml-1">
  Total Price : {totalPrice} ฿
</h1>
```

ส่วนที่ 3: การแสดงรายละเอียดคำสั่งซื้อของโถะ

```

/* Table */


<div className="overflow-x-auto">
    <table className="w-full table-auto">
      <thead>
        <tr className="uppercase text-left text-Text border-b border-Text/10">
          <th className="py-3 px-4">No.</th>
          <th className="py-3 px-4">User</th>
          <th className="py-3 px-4">Total Price</th>
          <th className="py-3 px-4">Time</th>
          <th className="py-3 px-4">Amount</th>
          <th className="py-3 px-4">Status</th>
          <th className="py-3 px-4">Show Detail</th>
        </tr>
      </thead>

      <tbody>
        {orders.map((order, index) => (
          <React.Fragment key={index}>
            <tr className="border-b border-Text/10 text-Text/80">
              <td className="py-3 px-4">{index + 1}</td>
              <td className="py-3 px-4">{order.userID}</td>
              <td className="py-3 px-4 truncate max-w-xs">
                {order.products.reduce((acc, product) => acc + product.totalPrice, 0)}
              </td>
              <td className="py-3 px-4">
                {moment(order.createdAt).format("HH:mm:ss")}
              </td>
              <td className="py-3 px-4">
                {order.products.reduce((acc, product) => acc + product.quantity, 0)}
              </td>
              <td className="py-3 px-4">
                <select
                  className="border rounded-lg p-2 bg-BG"
                  value={order.status}
                  onChange={({e}) => handleStatusChange(index, e.target.value, order._id)}
                >
                  <option value="Ordering">Ordering</option>
                  <option value="Cooking">Cooking</option>
                  <option value="Serving">Serving</option>
                  <option value="Completed">Completed</option>
                </select>
              </td>
              <td className="py-3 px-4">
                <button
                  className="border rounded-lg"
                  onClick={() => toggleRow(index)}
                >
                  {expandedRows[index] ? (
                    <ChevronUp className="size-8 text-red-600" />
                  ) : (
                    <ChevronDown className="size-8 text-green-600" />
                  )}
                </button>
              </td>
            </tr>
            {expandedRows[index] && (
              <tr className="bg-Text/5">
                <td colSpan={7} className="py-3 px-4 text-Text">
                  <table className="table-auto w-full rounded-xl">
                    <thead>
                      <tr className="border-b">
                        <th className="px-4 py-2">Images</th>
                        <th className="px-4 py-2">Product Name</th>
                        <th className="px-4 py-2">Price</th>
                        <th className="px-4 py-2">Quantity</th>
                        <th className="px-4 py-2">Requirement</th>
                        <th className="px-4 py-2">Sum Price</th>
                      </tr>
                    </thead>
                    <tbody>
                      {order.products.map((product, productIndex) => (
                        <tr key={productIndex} className="text-2xl border-b">
                          <td className="px-4 py-2 min-h-44 flex justify-center items-center mt-auto">
                            <img
                              src={product.image[0]}
                              alt={product.name}
                              className="w-20 h-20 object-cover rounded-xl"
                            />
                          </td>
                          <td className="px-4 py-2 border-r">{product.name}</td>
                          <td className="px-4 py-2 text-center border-r">{product.price}</td>
                          <td className="px-4 py-2 text-center border-r">{product.quantity}</td>
                          <td className="px-4 py-2 w-1/3 border-r">{product.requirement || "No special requirement"}</td>
                          <td className="px-4 py-2 border-r">{product.totalPrice}</td>
                        </tr>
                      )))
                    </tbody>
                  </table>
                </td>
              </tr>
            )}
          </React.Fragment>
        )));
      </tbody>
    </table>
  </div>
</div>


```

ส่วนที่ 4: การแสดงปุ่มเปลี่ยนสถานะคำสั่งชื่อของโถะ

```
<td className="py-3 px-4">
  <select
    className="border rounded-lg p-2 bg-BG"
    value={order.status}
    onChange={(e) => handleStatusChange(index, e.target.value, order._id)}
  >
    <option value="Ordering">Ordering</option>
    <option value="Cooking">Cooking</option>
    <option value="Serving">Serving</option>
    <option value="Completed">Completed</option>
  </select>
</td>
```

```
const handleStatusChange = async (index, newStatus, orderId) => {
  try {
    const response = await axios.post(backendURL + `/api/order/update`, {
      id: orderId,
      status: newStatus,
    });

    if (response.data.success) {
      toast.success("Order status updated successfully");
      setOrders((prevOrders) =>
        prevOrders.map((order, i) =>
          i === index ? { ...order, status: newStatus } : order
        )
      );
      fetchOrders();
    } else {
      toast.error("Failed to update status");
    }
  } catch (error) {
    console.error(error);
    toast.error("Error updating order status");
  }
};
```

ส่วนที่ 5: การลบคำสั่งชื่อของโต๊ะ

```
<button
    onClick={() => handleClearOrder()}
    className="flex items-center justify-center self-start w-full sm:w-auto h-full p-4 gap-3 text-white hover:bg-red-400 bg-red-600 rounded-2xl "
>
    <Eraser />
    <p>Clear Order</p>
</button>
```

```
const handleClearOrder = async () =>{
  if (orders.length === 0) {
    toast.info("There are no orders to clear.");
    return;
  }
  const products = orders.reduce((acc, order) => {
    return acc.concat(order.products);
  }, []);
  try {
    const response = await axios.post(backendURL + '/api/analytics/add',{products : products,table : tableNumber})
    await axios.post(backendURL + '/api/table/clear',{table : tableNumber})
    if(response.data.success){
      toast.success(response.data.message);
      setOrders([]);
      setTotalPrice(0);
      await axios.post(backendURL + `/api/order/remove`,{ tableNumber: tableNumber })
      fetchAnalytics();
      fetchOrders();
    }else{
      toast.error(response.data.message);
    }
  } catch (error) {
    console.log(error);
    toast.error(error.message);
  }
}
```

Controller

orderController.js

คำอธิบาย

ไฟล์จัดการกับการดำเนินการทั้งหมดที่เกี่ยวข้องกับคำสั่งซื้อของลูกค้า ประกอบด้วยฟังก์ชันเพิ่มคำสั่งซื้อ ลบคำสั่งซื้อ อัปเดตสถานะคำสั่งซื้อ แสดงรายการคำสั่งซื้อทั้งหมด หรือคำสั่งซื้อตามโต๊ะ เพิ่มคำสั่งซื้อ ลบคำสั่งซื้อ อัปเดตสถานะคำสั่งซื้อ แสดงรายการคำสั่งซื้อทั้งหมด หรือคำสั่งซื้อตามโต๊ะ

โค้ดสำคัญ

ส่วนที่ 1: ฟังก์ชันการเพิ่มคำสั่งซื้อไปในฐานข้อมูล

```
const addOrder = async (req, res) => {
  try {
    const { tableNumber, userID, products } = req.body;
    const productsArray = JSON.parse(products);
    const productsData = productsArray.map(product => {
      const { id, name, price, quantity, requirement, image } = product;
      const totalPrice = Number(price) * Number(quantity);
      return {
        id,
        name,
        price: Number(price),
        quantity: Number(quantity),
        totalPrice,
        requirement,
        image,
      };
    });
    const orderData = {
      tableNumber,
      userID,
      products: productsData,
      status: 'Ordering',
      createdAt: Date.now(),
    };

    const order = new orderModel(orderData);
    await order.save();

    res.json({ success: true, message: "Order Added" });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 2: พัฒนาการลบคำสั่งซื้อออกจากฐานข้อมูล

```
const removeOrder = async (req, res) => {
  try {
    const { tableNumber } = req.body;
    const result = await orderModel.deleteMany({ tableNumber });

    if (result.deletedCount > 0) {
      res.json({ success: true, message: `${result.deletedCount} Orders Removed` });
    } else {
      res.json({ success: false, message: "No orders found for this table number" });
    }
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 3: พัฒนาการอัปเดตสถานะคำสั่งซื้อ

```
const updateOrder = async (req, res) => {
  try {
    const { id, status } = req.body;
    await orderModel.findByIdAndUpdate(id, { status });
    res.json({ success: true, message: "Order Status Updated" });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 4: พังก์ชันการแสดงรายการคำสั่งซื้อทั้งหมดในฐานข้อมูล

```
const listOrders = async (req, res) => {
  try {
    const order = await orderModel.find({});
    res.json({ success: true, order });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 5: พังก์ชันการแสดงรายการคำสั่งซื้อตามหมายเลขโต๊ะ

```
const listOrdersByTable = async (req, res) => {
  const { tableNumber } = req.params;
  try {
    const orders = await orderModel.find({ tableNumber: tableNumber });

    if (orders.length === 0) {
      return res.json({ success: false, message: `No orders found for ${tableNumber}` });
    }

    const totalFoodCount = orders.reduce((count, order) => {
      order.products.forEach(item => {
        count += item.quantity; // Accumulate the quantity
      });
      return count;
    }, 0);

    res.json({ success: true, orders, totalFoodCount });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

productController.js

คำอธิบาย

ไฟล์จัดกับการดำเนินการทั้งหมดที่เกี่ยวข้องกับรายการอาหารในเมนู ประกอบด้วยฟังก์ชันเพิ่มเมนูอาหาร ลบเมนูอาหาร อัปเดตเมนูอาหาร และการแสดงรายการอาหารทั้งหมด

โค้ดสำคัญ

ส่วนที่ 1: ฟังก์ชันการเพิ่มเมนูอาหารไปในฐานข้อมูล

```
const addProduct = async (req,res) => {
    try {
        const {name, description, price, rate, time, Kcal, category, recommend} = req.body;

        const image1 = req.files.image1 && req.files.image1[0]
        const image2 = req.files.image2 && req.files.image2[0]
        const image3 = req.files.image3 && req.files.image3[0]
        const image4 = req.files.image4 && req.files.image4[0]

        const images = [image1,image2,image3,image4].filter((item) => item !== undefined)

        let imageUrl = await Promise.all(
            images.map(async (item) => {
                let result = await cloudinary.uploader.upload(item.path,{resource_type:'image'});
                return result.secure_url
            })
        )

        const productData = {
            name,
            description,
            price : Number(price),
            image: imageUrl,
            rate,
            time: JSON.parse(time),
            Kcal,
            category,
            date:Date.now(),
            recommend: recommend === "true" ? true : false
        }

        console.log(productData);

        const product = new productModel(productData)
        await product.save()

        res.json({success:true,message:"Product Added"});
    } catch (error) {
        res.json({success:false,message:error.message})
    }
}
```

ส่วนที่ 2: พัฒนาการลบเมนูอาหารออกจากฐานข้อมูล

```
const removeProduct = async (req, res) => {
    try {
        await productModel.findByIdAndDelete(req.body.id)
        res.json({ success: true, message: "Product Removed" })
    } catch (error) {
        res.json({ success: false, message: error.message })
    }
}
```

ส่วนที่ 3: พัฒนาการอัปเดตข้อมูลเมนูอาหาร

```
const updateProduct = async (req, res) => {
    try {
        const { productId, name, description, price, rate, time, Kcal, category, recommend } = req.body;

        const image1 = req.files && req.files.image1 && req.files.image1[0];
        const image2 = req.files && req.files.image2 && req.files.image2[0];
        const image3 = req.files && req.files.image3 && req.files.image3[0];
        const image4 = req.files && req.files.image4 && req.files.image4[0];

        const images = [image1, image2, image3, image4].filter((item) => item !== undefined);

        let imageUrl = [];
        if (images.length > 0) {
            imageUrl = await Promise.all(
                images.map(async (item) => {
                    let result = await cloudinary.uploader.upload(item.path, { resource_type: 'image' });
                    return result.secure_url;
                })
            );
        }

        const updatedData = {
            name,
            description,
            price: Number(price),
            rate,
            time: JSON.parse(time),
            Kcal,
            category,
            recommend: recommend === "true" ? true : false,
            image: imageUrl.length > 0 ? imageUrl : undefined
        };

        const updatedProduct = await productModel.findByIdAndUpdate(productId, updatedData, { new: true });

        if (!updatedProduct) {
            return res.json({ success: false, message: "Product not found" });
        }

        res.json({ success: true, message: "Product Updated", updatedProduct });
    } catch (error) {
        res.json({ success: false, message: error.message });
    }
};
```

ส่วนที่ 4: พัฒนาการแสดงรายการอาหารทั้งหมดในฐานข้อมูล

```
const listProducts = async (req,res) => {
    try {
        const product = await productModel.find({});
        res.json({success:true,product})
    } catch (error) {
        res.json({success:false,message:error.message})
    }
}
```

tableController.js

คำอธิบาย

ไฟล์จัดการกับการดำเนินการทั้งหมดที่เกี่ยวข้องกับข้อมูลโต๊ะ ประกอบด้วยฟังก์ชันการเพิ่ม ลบ แก้ไข อัปเดตสถานะความพร้อมใช้งานของโต๊ะ ล้างข้อมูลคำสั่งซึ่งออกจากโต๊ะ และแสดงรายการโต๊ะทั้งหมด และการเรียกพนักงาน

โค้ดสำคัญ

ส่วนที่ 1: ฟังก์ชันการเพิ่มโต๊ะไปในฐานข้อมูล

```
const addTable = async (req, res) => {
  try {
    const { table } = req.body;

    if (!table || isNaN(table)) {
      return res.json({success: false, message: "Invalid table number."});
    }

    let existingTable = await tableModel.findOne({ table });
    if (existingTable) {
      return res.json({success: false, message: "Table already exists."});
    }

    const newTable = new tableModel({ table, users: [], available: false, callWaiter: false });
    await newTable.save();

    res.json({success: true, message: `Table ${table} has been created.`});
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 2: พัฒนาการลบโต๊ะออกจากฐานข้อมูล

```
const deleteTable = async (req, res) => {
  try {
    const { tableNumber } = req.body;
    const table = await tableModel.findOne({ table: tableNumber });

    if (!table) {
      return res.json({ success: false, message: `Table with number ${tableNumber} not found.` });
    }

    await tableModel.deleteOne({ table: tableNumber });

    return res.json({ success: true, message: `Table ${tableNumber} has been deleted successfully.` });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 3: พัฒนาอัปเดตสถานะความพร้อมใช้งานของโต๊ะ

```
const availableTable = async (req, res) => {
  try {
    const { table, available } = req.body;
    let tableData = await tableModel.findOne({ table });

    if (!tableData) {
      return res.json({ success: false, message: "Table not found." });
    }

    tableData.available = available;
    await tableData.save();

    res.json({ success: true, message: `Table ${table} is now ${available ? "open" : "closed"}.` });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 4: พัฒนาล้างข้อมูลคำสั่งซื้อออกจากโต๊ะ

```
const clearTable = async (req, res) => {
  try {
    const { table } = req.body;
    let tableData = await tableModel.findOne({ table });

    if (!tableData) {
      return res.json({ success: false, message: "Table not found." });
    }

    tableData.users = [];
    tableData.available = false;
    tableData.callWaiter = false;
    await tableData.save();

    res.json({ success: true, message: `Table ${table} has been cleared.` });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 5: พัฒนาการแสดงรายการโต๊ะทั้งหมดในฐานข้อมูล

```
const listTables = async (req, res) => {
  try {
    let tables = await tableModel.find({});
    res.json({ success: true, tables });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};
```

ส่วนที่ 6: พัฒนาการเรียกพนักงานให้มาบริการตามโต๊ะ

```
const callWaiter = async (req, res) => {
  try {
    const { table } = req.body;
    let tableData = await tableModel.findOne({ table });

    if (!tableData) {
      return res.json({ success: false, message: `Table ${table} does not exist.` });
    }

    tableData.callWaiter = true;
    await tableData.save();

    res.json({ success: true, message: `A waiter has been notified for table ${table}.` });
  } catch (error) {
    console.log(error);
    res.json({ success: false, message: error.message });
  }
};
```

staffController.js

คำอธิบาย

ไฟล์จัดการกับการดำเนินการทั้งหมดที่เกี่ยวข้องกับพนักงานในระบบ ประกอบด้วยฟังก์ชัน การเข้าสู่ระบบ การลงทะเบียนพนักงาน การอัปเดตข้อมูลพนักงาน การลบข้อมูลพนักงาน และการแสดงรายชื่อพนักงาน

โค้ดสำคัญ

ส่วนที่ 1: ฟังก์ชันการเข้าสู่ระบบ

```
const login = async (req, res) => {
  try {
    const { email, password } = req.body;

    if (email === process.env.ADMIN_EMAIL) {
      if (password === process.env.ADMIN_PASSWORD) {
        const token = jwt.sign(email + password, process.env.JWT_SECRET);
        return res.json({ success: true, token, role: "admin" });
      } else {
        return res.json({ success: false, message: "Invalid admin credentials" });
      }
    }

    const staff = await staffModel.findOne({ email });
    if (!staff) {
      return res.json({ success: false, message: "Staff doesn't exist" });
    }

    const isMatch = await bcrypt.compare(password, staff.password);
    if (isMatch) {
      const token = createToken(staff._id);
      return res.json({ success: true, token, role: "staff", id: staff._id });
    } else {
      return res.json({ success: false, message: "Invalid staff credentials" });
    }
  } catch (error) {
    return res.json({ success: false, message: error.message });
  }
};
```

សំណើលេខ 2: ផងក់ចំណាត់ការលងទៅបើយនពន្យការងារខ្សោតរាជាណម្មូត

```

const registerStaff = async (req, res) => {
    try {
        const { firstName, lastName, email, password, phone, address } = req.body;
        const profilePic = req.files.profilePic && req.files.profilePic[0];

        const exists = await staffModel.findOne({ email });
        if (exists) return res.json({ success: false, message: "Staff already exists" });
        if (!validator.isEmail(email)) return res.json({ success: false, message: "Please enter a valid email" });
        const phoneExists = await staffModel.findOne({ phone });
        if (phoneExists) return res.json({ success: false, message: "Phone number already registered" });
        if (!/^[\d{10,15}$/.test(phone)) return res.json({ success: false, message: "Invalid phone number format (must be 10-15 digits)" });
        if (password.length < 8) return res.json({ success: false, message: "Please enter a strong password (min 8 characters)" });

        let profilePicUrl = "";
        if (profilePic !== undefined) {
            try {
                let result = await cloudinary.uploader.upload(profilePic.path, { resource_type: "image" });
                profilePicUrl = result.secure_url;
            } catch (error) {
                console.error("Cloudinary Upload Error:", error);
                return res.json({ success: false, message: "Error uploading profile picture" });
            }
        }

        const salt = await bcrypt.genSalt(10);
        const hashedPassword = await bcrypt.hash(password, salt);

        const newStaff = {
            firstName,
            lastName,
            email,
            password: hashedPassword,
            phone,
            profilePic: profilePicUrl || "",
            address: address || ""
        };
        const staff = new staffModel(newStaff);
        await staff.save();

        res.json({ success: true, message: "Staff registered successfully" });
    } catch (error) {
        res.json({ success: false, message: "Server error, please try again." });
    }
};

```

ส่วนที่ 3: พัฒนาการอัปเดตข้อมูลพนักงาน

```

const updateStaff = async (req, res) => {
  try {
    const {staffId, firstName, lastName, email, password, phone, address} = req.body;
    const profilePic = req.files.profilePic && req.files.profilePic[0];

    const staff = await staffModel.findById(staffId);
    if (!staff) return res.json({ success: false, message: "Staff not found" });

    if (email && email !== staff.email) {
      const emailExists = await staffModel.findOne({ email });
      if (emailExists) return res.json({ success: false, message: "Email already registered" });
      if (!validator.isEmail(email)) return res.json({ success: false, message: "Invalid email format" });
    }

    if (phone && phone !== staff.phone) {
      const phoneExists = await staffModel.findOne({ phone });
      if (phoneExists) return res.json({ success: false, message: "Phone number already registered" });
      if (!/\^d{10,15}\$/.test(phone)) return res.json({ success: false, message: "Invalid phone number format (must be 10-15 digits)" });
    }

    let hashedPassword = staff.password;
    if (password) {
      if (password.length < 8) return res.json({ success: false, message: "Password must be at least 8 characters" });
      const salt = await bcrypt.genSalt(10);
      hashedPassword = await bcrypt.hash(password, salt);
    }

    let profilePicUrl = staff.profilePic;
    if (profilePic) {
      try {
        const result = await cloudinary.uploader.upload(profilePic.path, { resource_type: "image" });
        profilePicUrl = result.secure_url;
      } catch (error) {
        console.error("Cloudinary Upload Error:", error);
        return res.json({ success: false, message: "Error uploading profile picture" });
      }
    }
  }

  const updatedStaff = {
    firstName: firstName || staff.firstName,
    lastName: lastName || staff.lastName,
    email: email || staff.email,
    phone: phone || staff.phone,
    password: hashedPassword,
    profilePic: profilePicUrl,
    address: address || staff.address
  };

  const staffUpdate = await staffModel.findByIdAndUpdate(staffId, updatedStaff, { new: true });
  if (!staffUpdate) return res.json({ success: false, message: "Error updating staff" });

  res.json({ success: true, message: "Staff updated successfully", updatedStaff });
} catch (error) {
  res.json({ success: false, message: "Server error, please try again." });
}
};


```

ส่วนที่ 4: พัฒนาการลบข้อมูลพนักงานออกจากฐานข้อมูล

```
const removeStaff = async (req, res) => {
  try {
    await staffModel.findByIdAndDelete(req.body.id)
    res.json({success:true,message:"Staff removed successfully"})
  } catch (error) {
    console.log(error);
    res.json({success:false,message:error.message})
  }
}
```

ส่วนที่ 5: พัฒนาการแสดงรายชื่อพนักงานทั้งหมดในฐานข้อมูล

```
// function for list staff
const listStaff = async (req, res) => {
  try{
    const staff = await staffModel.find({});
    res.json({success:true,staff})
  }catch(error){
    console.log(error);
    res.json({success:false,message:error.message})
  }
}
```

analyticsController.js

คำอธิบาย

ไฟล์จัดการกับการดำเนินการทั้งหมดที่เกี่ยวข้องกับการวิเคราะห์และรายงานยอดขาย

โค้ดสำคัญ

ส่วนที่ 1: พิ่งก์ชันประมวลผลข้อมูลธุรกรรม โดยดึงรายละเอียดสินค้าจากคำสั่งซื้อ จากนั้นคำนวณรายได้รวม จำนวนคำสั่งซื้อ และอื่น ๆ และบันทึกข้อมูลยอดขายใหม่ลงในฐานข้อมูล

```

const addTransaction = async (req, res) => {
  try {
    const { products, table } = req.body;

    let totalIncome = 0, orderAmount = 0;
    const FoodData = products.map(product => {
      const { id, name, quantity, totalPrice, image } = product;
      totalIncome += totalPrice;
      orderAmount++;
      return {
        image,
        id,
        name,
        quantitySold: Number(quantity),
      };
    });

    let tableData = await tableModel.findOne({ table });

    const customerAmount = tableData ? tableData.users.length : 1;

    const SalesData = {
      date: new Date().toISOString(),
      totalIncome,
      OrderFood: FoodData,
      orderAmount,
      customerAmount,
    };

    console.log(SalesData);

    const sales = new analyticsModel(SalesData);
    await sales.save();

    res.json({ success: true, message: "Clear Order & Save Data" });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};

```

ส่วนที่ 2: พัฒนาดึงข้อมูลยอดขายตามช่วงเวลาที่ระบุ (รายวัน รายเดือน หรือรายปี)

```

const getAnalyticsData = async (req, res) => {
  try {
    const { dateRange } = req.query;

    let filter = {};

    // Create a date filter based on the dateRange
    if (dateRange === "daily") {
      const now = new Date();

      const offset = 7 * 60;
      const localStartOfDay = new Date(now.getFullYear(), now.getMonth(), now.getDate(), 0, 0, 0);
      const localEndOfDay = new Date(now.getFullYear(), now.getMonth(), now.getDate(), 23, 59, 59, 999);

      const startOfDay = new Date(localStartOfDay.getTime() - offset * 60 * 1000).toISOString();
      const endOfDay = new Date(localEndOfDay.getTime() - offset * 60 * 1000).toISOString();

      filter.date = { $gte: startOfDay, $lt: endOfDay };
    } else if (dateRange === "monthly") {
      const firstDayOfMonth = new Date();
      firstDayOfMonth.setDate(1);

      const lastDayOfMonth = new Date(firstDayOfMonth);
      lastDayOfMonth.setMonth(lastDayOfMonth.getMonth() + 1);
      lastDayOfMonth.setDate(0);

      filter.date = { $gte: firstDayOfMonth.toISOString(), $lt: lastDayOfMonth.toISOString() };
    } else if (dateRange === "yearly") {
      const firstDayOfYear = new Date(new Date().getFullYear(), 0, 1);
      const lastDayOfYear = new Date(new Date().getFullYear() + 1, 0, 0);

      filter.date = { $gte: firstDayOfYear.toISOString(), $lt: lastDayOfYear.toISOString() };
    }

    const analyticsData = await analyticsModel.find(filter).sort({ date: 1 });

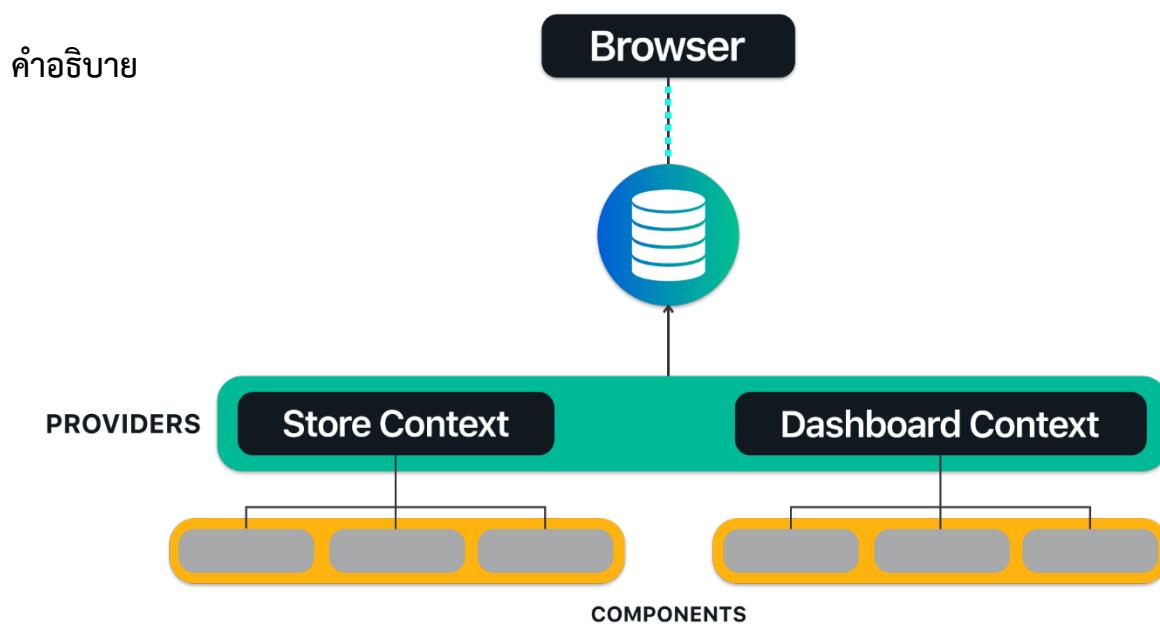
    if (!analyticsData.length) {
      return res.json({ success: false, message: "No data found" });
    }

    res.json({ success: true, sales: analyticsData });
  } catch (error) {
    res.json({ success: false, message: error.message });
  }
};

```

Design Pattern

Provider



Provider Pattern ถูกนำมาใช้โดยใช้ React Context API ได้แก่ไฟล์ **StoreContext.jsx** และ **DashboardContext.jsx** ซึ่งเป็นการรวมข้อมูลและฟังก์ชันสำคัญไว้ในที่เดียว เพื่อจัดการและส่ง state ไปให้คอมโพเนนท์อื่น ๆ ของระบบนำไปใช้งานได้ โดยที่

1. **StoreContext.jsx**

จัดการ state ในส่วนของร้านค้า เช่น หมายเลขโทรศัพท์ ไอดีลูกค้า รายการอาหาร ตะกร้า และคำสั่งซื้อ

2. **DashboardContext.jsx**

จัดการ state ในส่วนของผู้ดูแลระบบ เช่น ข้อมูลวิเคราะห์ รายการพนักงาน คำสั่งซื้อ และตัว

ประโยชน์

- คอมโพเนนท์เข้าถึงข้อมูลได้โดยตรงจาก Context
- บำรุงรักษาระบบได้ง่าย เนื่องจาก โค้ดการจัดการ state ถูกรวมไว้ในที่เดียว

Observer

คำอธิบาย

Observer Pattern ถูกนำมาใช้ผ่าน Socket.IO เพื่อจัดการกับการอัปเดตแบบ Real-Time โดยคอมโพเนนท์หน้าที่เป็น Observer ซึ่งจะอยู่ติดตามกับเหตุการณ์เฉพาะต่าง ๆ ที่ถูกส่งมาจาก backend เมื่อเหตุการณ์เกิดขึ้น ทำให้คอมโพเนนท์ที่สมัครสมาชิกทั้งหมดได้รับข้อมูล จากนั้นจึงปรับปรุง state ของตนเองโดยอัตโนมัติ ทำให้ UI มีการตอบสนองโดยที่ไม่ต้องให้ผู้ใช้งานกดรีเฟรช

ตัวอย่าง

ในส่วนของ Admin Dashboard จะมี DashboardContext.jsx ทำหน้าที่เป็น Observer อยู่ติดตามเหตุการณ์เกี่ยวกับ ข้อมูลคำสั่งซื้อ และข้อมูลโต๊ะ ทำให้คอมโพเนนท์ที่ติดต่อกับ Context นี้ นำข้อมูลไปอัปเดตตนเองแบบ Real-Time

```
useEffect(() => {
  socket.on("orderUpdated", fetchOrders);
  socket.on("tableUpdated", fetchTable);

  return () => {
    socket.off("orderUpdated", fetchOrders);
    socket.off("tableUpdated", fetchTable);
    socket.disconnect();
  };
}, [socket]);
```