
SHEOUT
Online Clothes shop

# MEMBERS

Natthanicha Jewaram

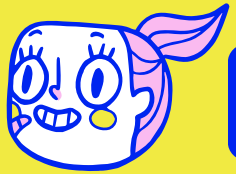
Nawapat Thongpitaks

Pongkit Yingyongtanasarn

# INTRODUCTIONS

## ADMIN

- SHOW HOMEPAGE LIKE USER'S VIEW FOR ADMIN
- CLOTHES'S COLLECTION FOR ADMIN
  1. Add Clothes's Collection
  2. Delete Clothes's Collection
- EDIT DESCRIPTION AND COST FOR ADMIN



## USER

- CLOTHES'S COLLECTION FOR USER
  1. View All Clothes's Collection In Homepage
  2. View Only Clothes's Collection That User Want To Buy
- CLOTH FOR USER
- BASKET FOR USER
  1. View Orders In User's Basket
  2. Add Clothes Into User's Basket
  3. Delete Clothes In User's Basket
  4. Confirm Order In User's Basket

PAST 1

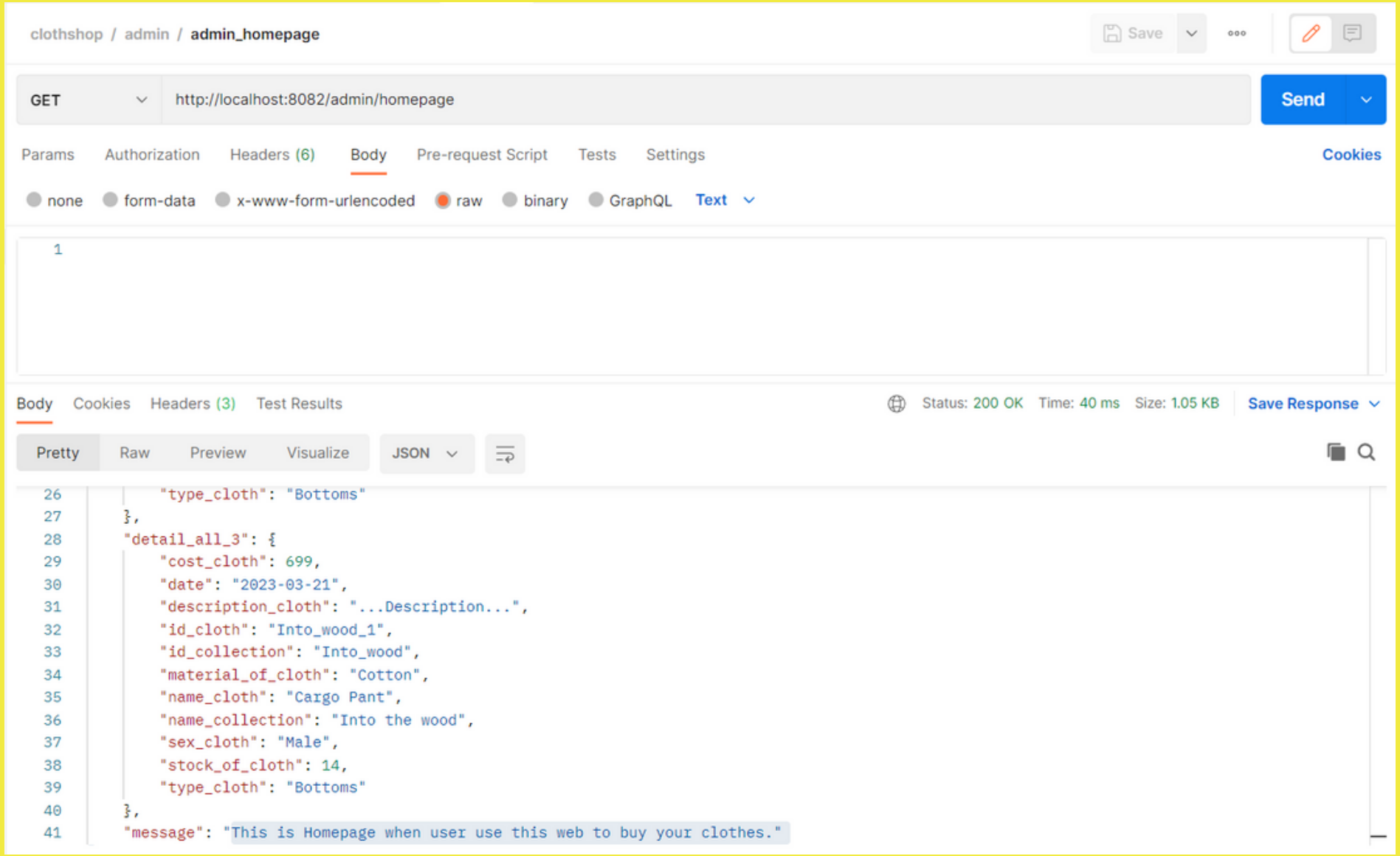
# FOR ADMIN

# SHOW HOMEPAGE LIKE USER'S VIEW FOR ADMIN

```rust
// ****************** Admin สามารถดูหน้า Home page ได้ในมุมมองเดียวกันกับที่ User เห็นตอนเลือก S*********************//
#[get("/admin/homepage")]
async fn show_detail_admin() -> impl Responder {
    info!("Show Homepage Like User's view For Admin");

    let mut _data1: AllDetailClothCollection = AllDetailClothCollection {
        name_cloth: "Snowman Overcoat".to_string(),
        id_cloth: "Cool_win_1".to_string(),
        name_collection: "Be cool in Winter".to_string(),
        id_collection: "Cool_win".to_string(),
        description_cloth: "...Description...".to_string(),
        cost_cloth: 1199,
        type_cloth: "Tops".to_string(),
        material_of_cloth: "Wool".to_string(),
        sex_cloth: "Female".to_string(),
        stock_of_cloth: 42,
        date: "2023-03-20".to_string(),
    };
```

```rust
#[derive(Serialize,Deserialize)]
pub struct AllDetailClothCollection {
    pub name_cloth: String,
    pub id_cloth: String,
    pub name_collection: String,
    pub id_collection: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub type_cloth: String,
    pub material_of_cloth: String,
    pub sex_cloth: String,
    pub stock_of_cloth: i32,
    pub date : String
}
```

```rust
    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        detail_all_1: AllDetailClothCollection,
        detail_all_2: AllDetailClothCollection,
        detail_all_3: AllDetailClothCollection,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        detail_all_1: _data1,
        detail_all_2: _data2,
        detail_all_3: _data3,
        message: "This is Homepage when user use this web to buy your clothes.".to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)
} fn show_detail_admin

```


clothshop / admin / admin_homepage
Save
GET
http://localhost:8082/admin/homepage
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
Text
1
Body
Cookies
Headers (3)
Test Results
Status: 200 OK
Time: 40 ms
Size: 1.05 KB
Save Response
Pretty
Raw
Preview
Visualize
JSON
"type_cloth": "Bottoms"
},
"detail_all_3": {
"cost_cloth": 699,
"date": "2023-03-21",
"description_cloth": "...Description...",
"id_cloth": "Into_wood_1",
"id_collection": "Into_wood",
"material_of_cloth": "Cotton",
"name_cloth": "Cargo Pant",
"name_collection": "Into the wood",
"sex_cloth": "Male",
"stock_of_cloth": 14,
"type_cloth": "Bottoms"
},
"message": "This is Homepage when user use this web to buy your clothes."

# CLOTHES'S COLLECTION FOR ADMIN

- Add Clothes's Collection
- Delete Clothes's Collection

# ADD CLOTHES'S COLLECTION

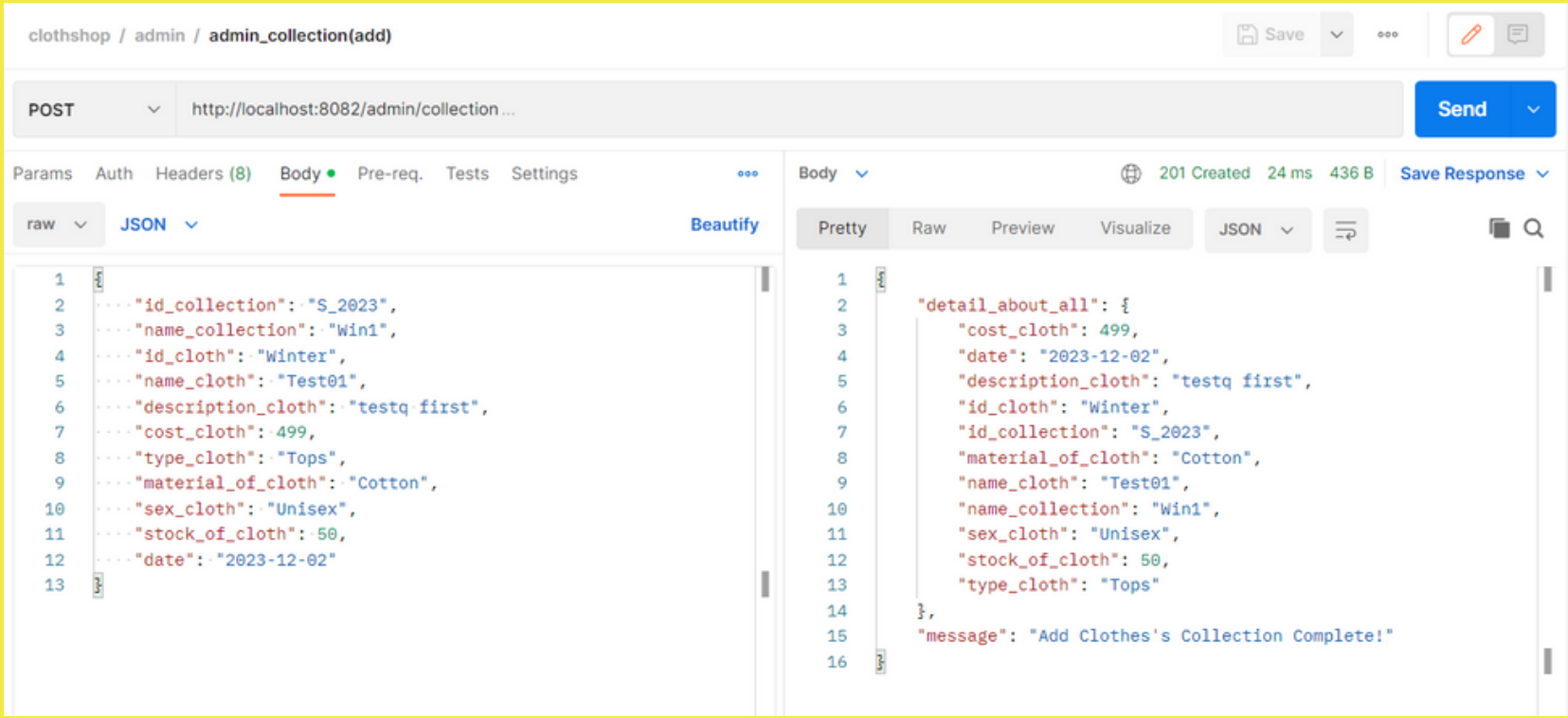
```rust
//****************** Admin สามารถเพิ่ม Clothes's Collection ได้ *********************************//

#[post("/admin/collection")]
async fn add_clothes_collection(input_form: web::Json<AllDetailClothCollection>) -> impl Responder {
    info!("admin add (http:post)");

    let new_collection: AllDetailClothCollection = input_form.into_inner();


    let name_collection_for_add: &str = "";

    let mut _data: AllDetailClothCollection = AllDetailClothCollection {
        id_collection: new_collection.id_collection.to_string(),
        name_collection: new_collection.name_collection.to_string(),
        id_cloth: new_collection.id_cloth.to_string(),
        name_cloth: new_collection.name_cloth.to_string(),
        description_cloth: new_collection.description_cloth.to_string(),
        cost_cloth: new_collection.cost_cloth,
        type_cloth: new_collection.type_cloth.to_string(),
        material_of_cloth: new_collection.material_of_cloth.to_string(),
        sex_cloth: new_collection.sex_cloth.to_string(),
        stock_of_cloth: new_collection.stock_of_cloth,
        date: new_collection.date.to_string(),
    };

```

```rust
#[derive(Serialize,Deserialize)]
pub struct AllDetailClothCollection {
    pub name_cloth: String,
    pub id_cloth: String,
    pub name_collection: String,
    pub id_collection: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub type_cloth: String,
    pub material_of_cloth: String,
    pub sex_cloth: String,
    pub stock_of_cloth: i32,
    pub date : String
}
```

# ADD CLOTHES'S COLLECTION

```
36      //รวมกับ message ของระบบเพื่อแจ้งเตือนการแสดงผล
37
38      #[derive(Serialize, Deserialize)]
39      struct WebResponse {
40          detail_about_all: AllDetailClothCollection,
41          message: String,
42      }
43
44      let web_response: WebResponse = WebResponse {
45          detail_about_all: _data,
46          message: "Add Clothes's Collection Complete!".to_string(),
47      };
48
49      let response: Value = json!(web_response);
50
51      HttpResponse::Created().json(response)
52  } fn add_clothes_collection
53
```

# ADD CLOTHES'S COLLECTION



# DELETE CLOTHES'S COLLECTION

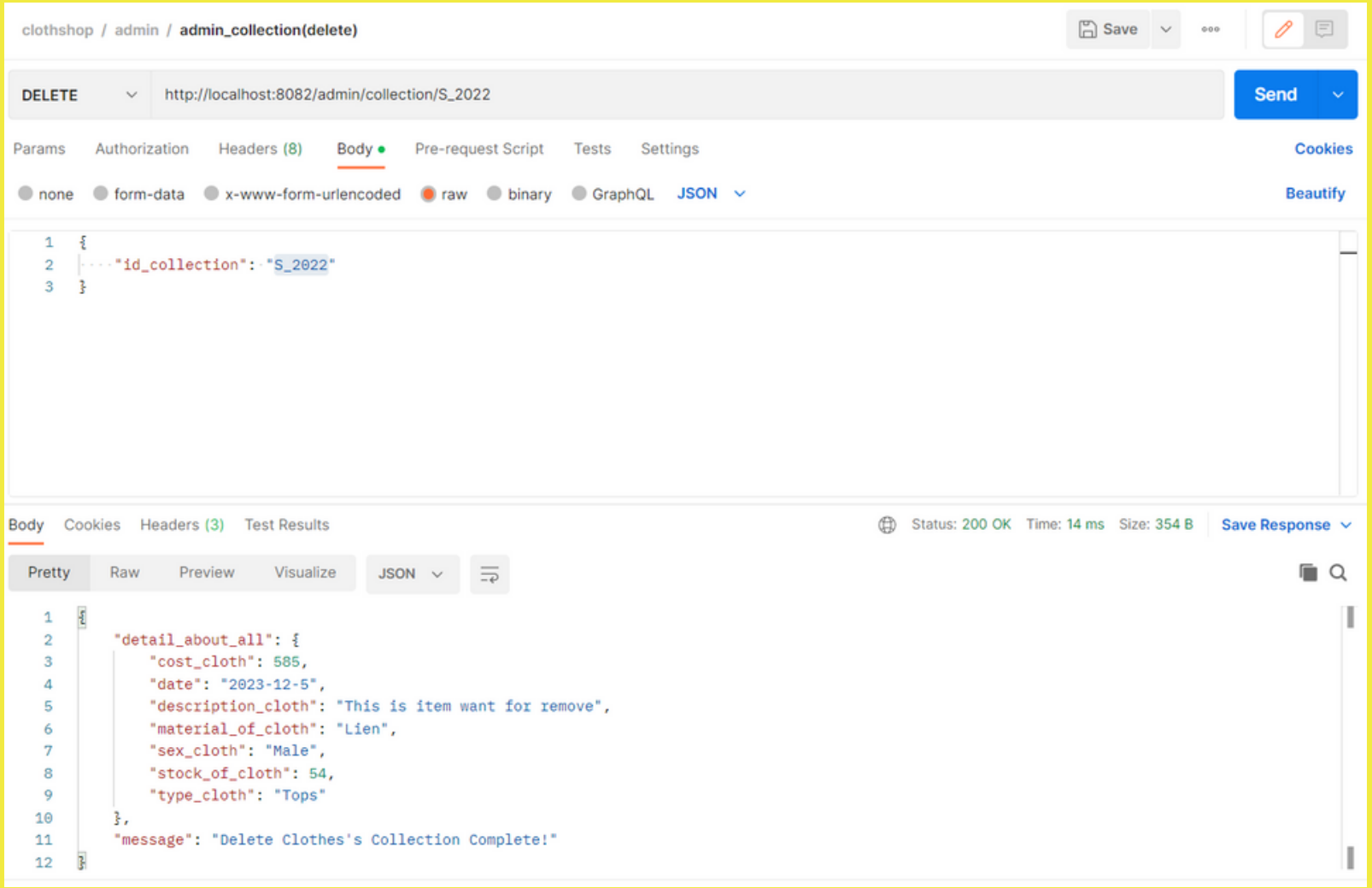
```rust
//****************** Admin สามารถลบ Clothes's Collection ได้ **********************//

#[delete("/admin/collection/{id_collection}")]
async fn delete_clothes_collection(
    id: web::Path<String>,
    input_form: web::Json<EventidCollection>,
) -> impl Responder {
    info!("Delete Clothes's By IDCollection ");

    let id: String = id.to_string();
    let collection_data: EventidCollection = input_form.into_inner();
    // let id:String = id.to_string();
    //สมมติข้อมูลที่ต้องการลบ
    let mut old_collection: BodyClothCollection = BodyClothCollection {
        description_cloth: "This is item want for remove".to_string(),
        cost_cloth: 585,
        type_cloth: "Tops".to_string(),
        material_of_cloth: "Lien".to_string(),
        sex_cloth: "Male".to_string(),
        stock_of_cloth: 54,
        date: "2023-12-5".to_string(),
    };
    let mut message_delete: &str = "";
    //เช็คว่าurlกับbodyตรงกันไหม
    if id.to_string() != collection_data.id_collection.to_string() {
        message_delete = "ID not found or Not Match !!";
        old_collection = BodyClothCollection {
            description_cloth: "No Data".to_string(),
            cost_cloth: 0,
            type_cloth: "No Data".to_string(),
            material_of_cloth: "No Data".to_string(),
            sex_cloth: "No Data".to_string(),
            stock_of_cloth: 0,
            date: "No Data".to_string(),
        };
    } else {
        message_delete = "Delete Clothes's Collection Complete!";
    }

```

```rust
#[derive(Serialize, Deserialize)]

struct EventidCollection {
    id_collection: String,
}
```

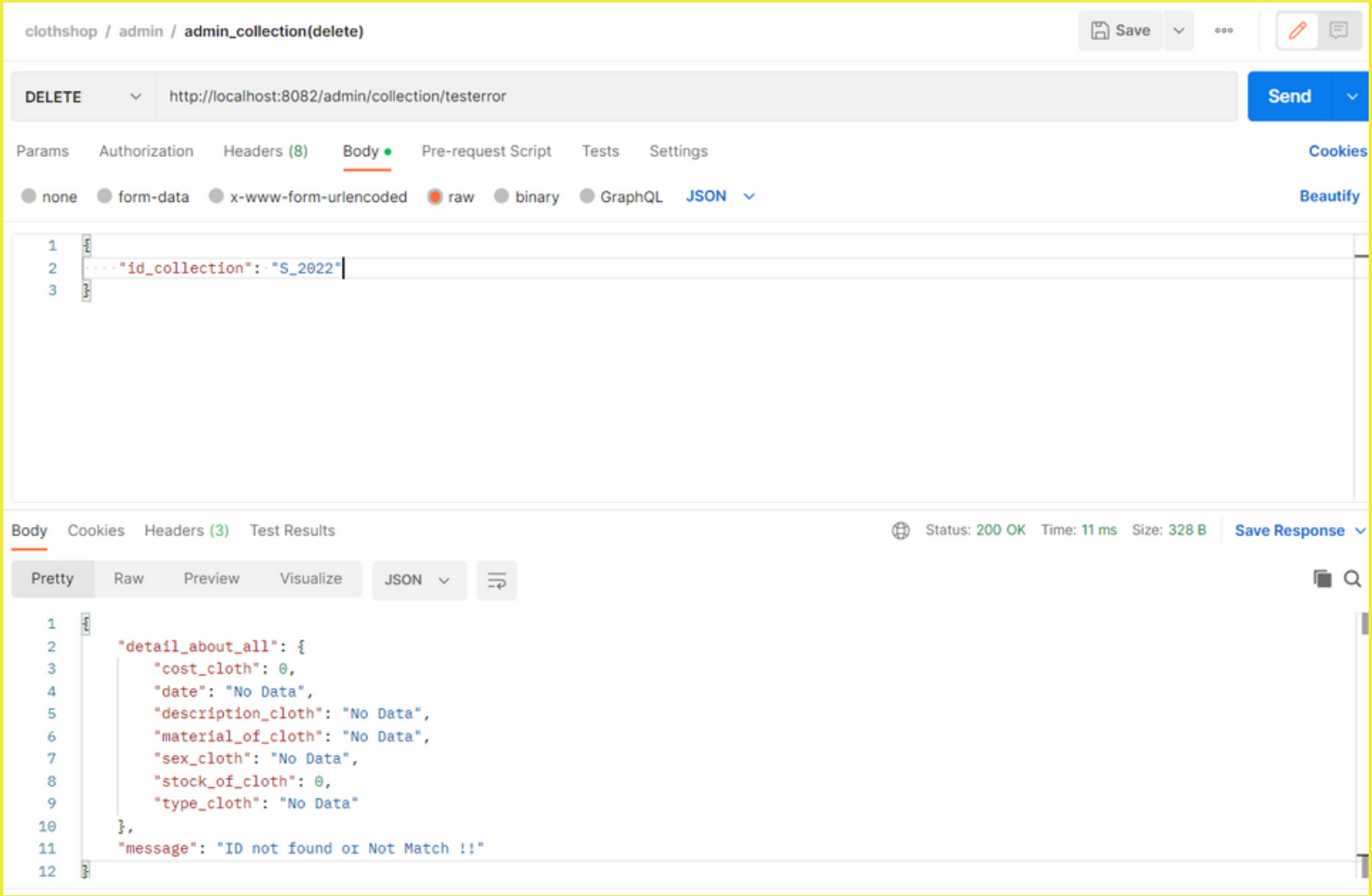
# DELETE CLOTHES'S COLLECTION

```rust
    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        detail_about_all: BodyClothCollection,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        detail_about_all: old_collection,
        message: message_delete.to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)
} fn delete_clothes_collection
```

# DELETE CLOTHES'S COLLECTION



# DELETE CLOTHES'S COLLECTION (ERROR)

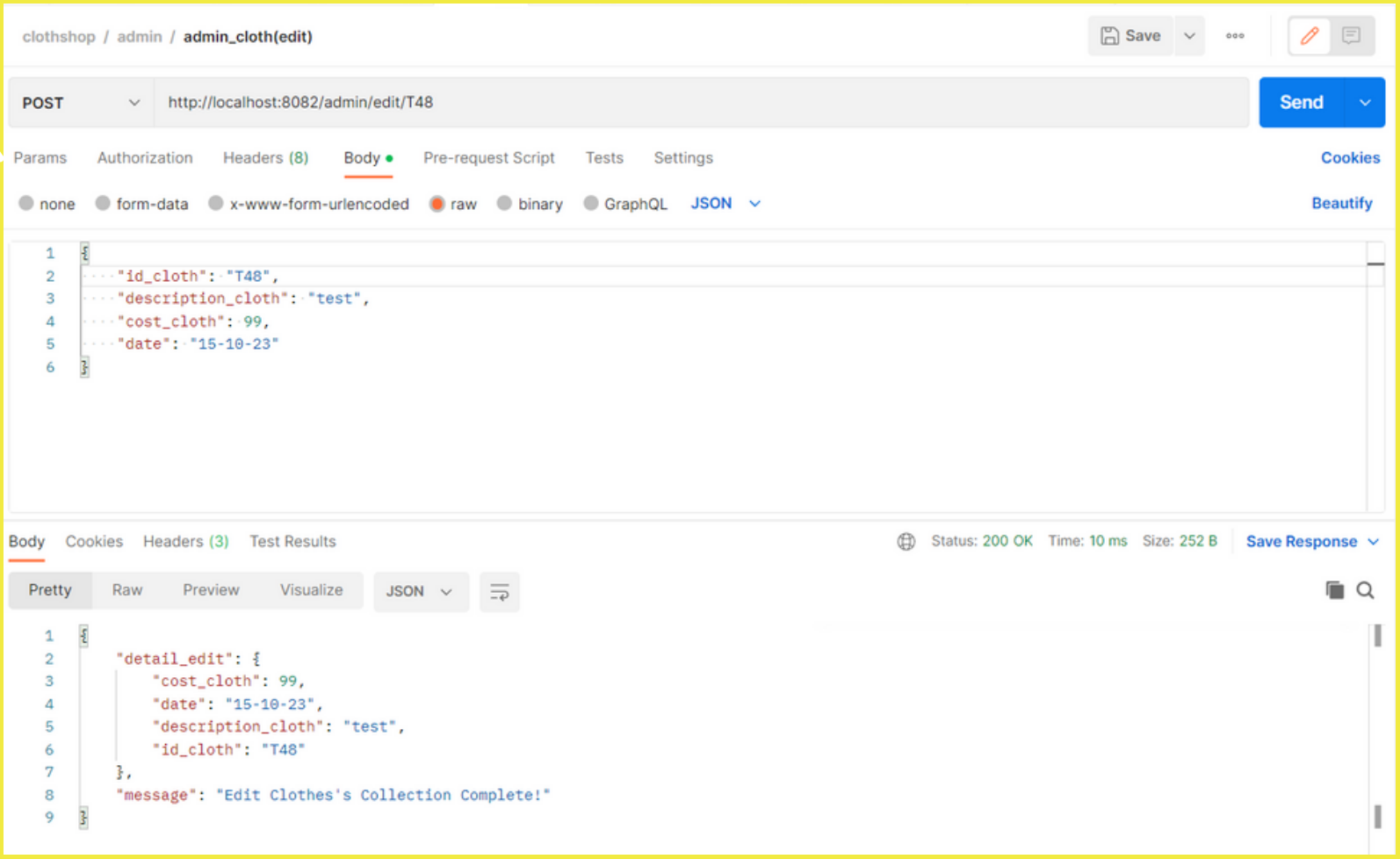


# EDIT DESCRIPTION AND COST FOR ADMIN

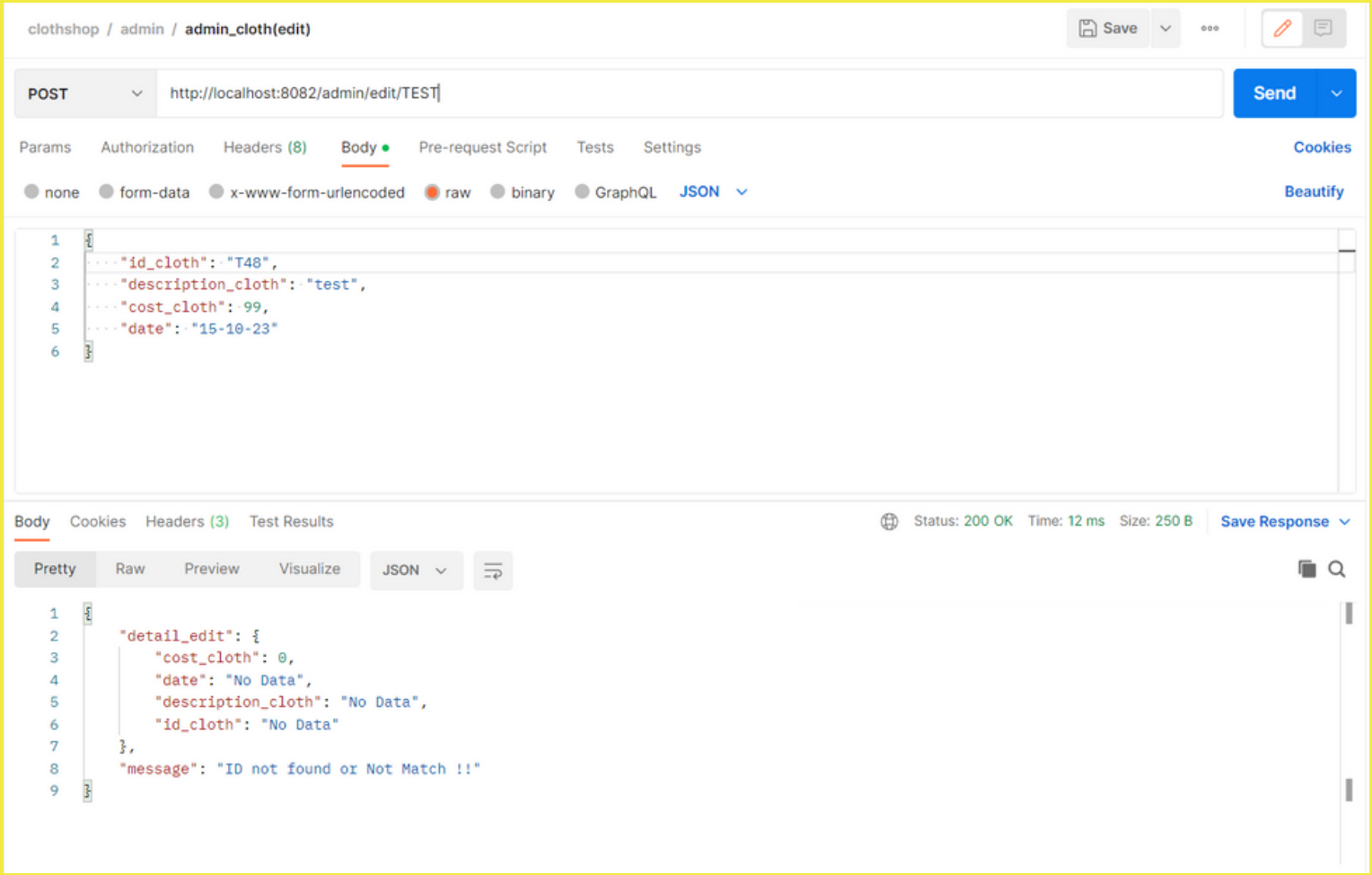
```rust
//********************** Admin สามารถแก้ไขคำอธิบายและราคาของ Clothes's Collection ได้พร้อมลงวันที่การแก้ไขข้อมูลครั้งล่าสุด *****************//
#[post("/admin/edit/{id_cloth}")]
async fn edit_clothes_collection(id:web::Path<String>,input_form:web::Json<ClothEditList>) -> impl Responder {
    info!("Edit Clothes's By id cloth");

    let id:String = id.to_string();
    let id_edit: ClothEditList = input_form.into_inner();

    let mut _edit_detail1: ClothEditList = ClothEditList{
            id_cloth: id_edit.id_cloth.to_string(),
            description_cloth : id_edit.description_cloth.to_string(),
            cost_cloth : id_edit.cost_cloth,
            date : id_edit.date.to_string(),
        };
    let mut message_edit: &str ="Your new correction are complete! Please check your detail about your new correction.";
    //เช็คว่าurlกับbodyตรงกันไหม
    if id.to_string() != id_edit.id_cloth.to_string(){
        message_edit = "ID not found or Not Match !!";
        _edit_detail1 = ClothEditList{
            id_cloth :"No Data".to_string(),
            description_cloth : "No Data".to_string(),
            cost_cloth : 0,
            date : "No Data".to_string(),
        };
    }else{
        message_edit = "Edit Clothes's Collection Complete!";

    }

    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        detail_edit: ClothEditList,
        message: String
    }

    let web_response: WebResponse = WebResponse {
        detail_edit: _edit_detail1,
        message:message_edit.to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)
} fn edit_clothes_collection
```

```rust
#[derive(Serialize,Deserialize)]
2 implementations
pub struct ClothEditList{
    pub id_cloth: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub date : String
}
```

# EDIT CLOTHES'S COLLECTION



# EDIT CLOTHES'S COLLECTION (ERROR)



Part 2

# FOR USER

# CLOTHES'S COLLECTION FOR USER

- View All Clothes's Collection In Homepage
- View Only Clothes's Collection That User Want To Buy

# VIEW ALL CLOTHES'S COLLECTION IN HOMEPAGE

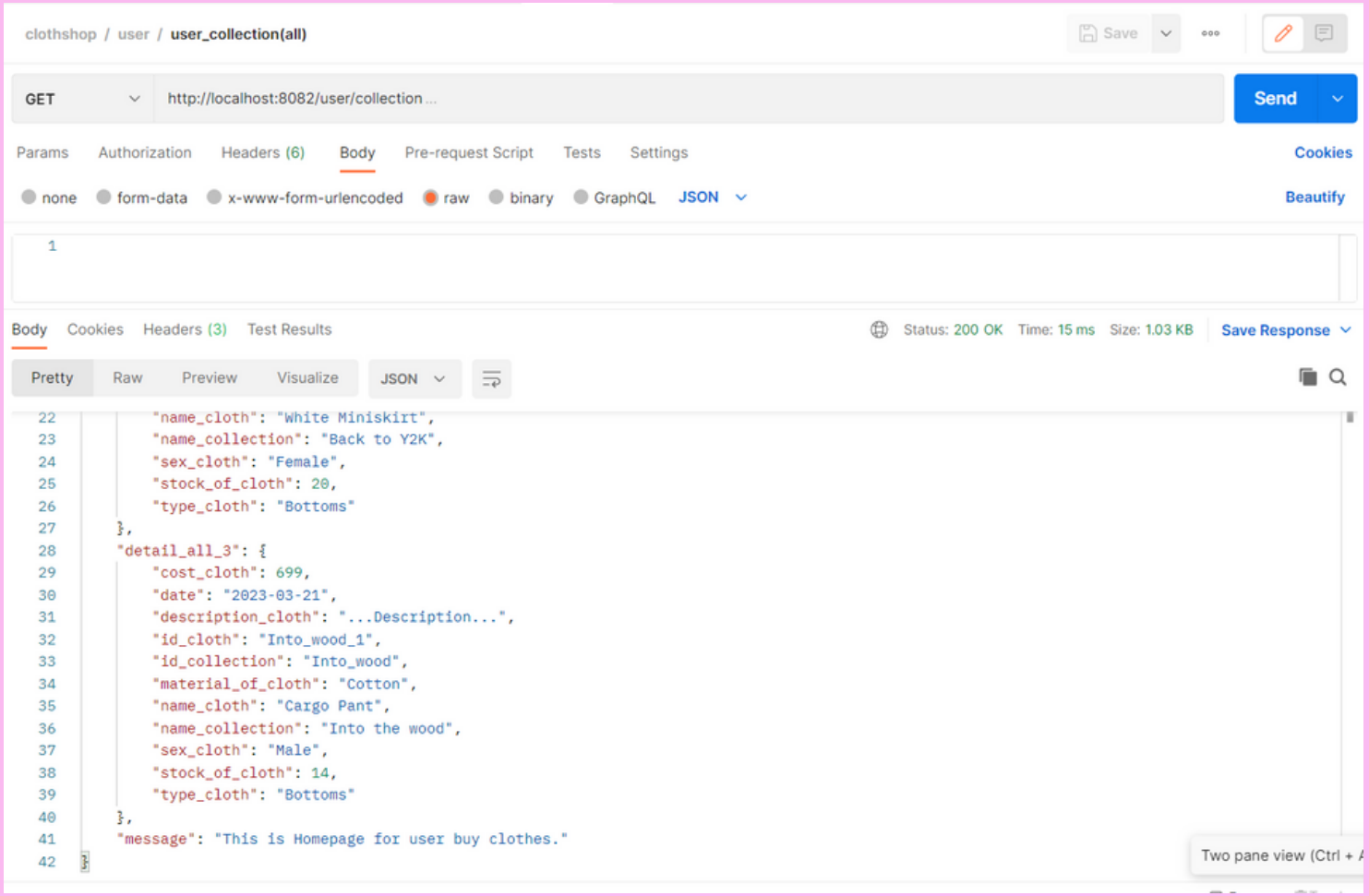
```rust
//*********** User ดู Cothes's Collection ทั้งหมดในหน้า Homepage ********//

#[get("/user/collection")]
async fn view_cloth_all_user() -> impl Responder {
    info!("Show Homepage Like User's view");

    let mut _data1: AllDetailClothCollection = AllDetailClothCollection {
        name_cloth: "Snowman Overcoat".to_string(),
        id_cloth: "Cool_win_1".to_string(),
        name_collection: "Be cool in Winter".to_string(),
        id_collection: "Cool_win".to_string(),
        description_cloth: "...Description...".to_string(),
        cost_cloth: 1199,
        type_cloth: "Tops".to_string(),
        material_of_cloth: "Wool".to_string(),
        sex_cloth: "Female".to_string(),
        stock_of_cloth: 42,
        date: "2023-03-20".to_string(),
    };
```

```rust
#[derive(Serialize,Deserialize)]
pub struct AllDetailClothCollection {
    pub name_cloth: String,
    pub id_cloth: String,
    pub name_collection: String,
    pub id_collection: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub type_cloth: String,
    pub material_of_cloth: String,
    pub sex_cloth: String,
    pub stock_of_cloth: i32,
    pub date : String
}
```

# VIEW ALL CLOTHES'S COLLECTION IN HOMEPAGE

```rust
    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        detail_all_1: AllDetailClothCollection,
        detail_all_2: AllDetailClothCollection,
        detail_all_3: AllDetailClothCollection,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        detail_all_1: _data1,
        detail_all_2: _data2,
        detail_all_3: _data3,
        message: "This is Homepage for user buy clothes.".to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)
} fn view_cloth_all_user
```

# VIEW ALL CLOTHES'S COLLECTION IN HOMEPAGE



# VIEW ONLY CLOTHES'S COLLECTION THAT USER WANT TO BUY

```rust
//***************** User ดูแค่ Clothes's Collection ที่สนใจซื้อเพียง 1 Collection เท่านั้น ******************//

#[get("/user/collection/{id_collection}")]
async fn view_collection_only_user(
    id: web::Path<String>,
    input_form: web::Json<EventidCollection>,
) -> impl Responder {
    info!("Show Only Collection");

    let id: String = id.to_string();
    let collection_id: EventidCollection = input_form.into_inner();
    // let id:String = id.to_string();
    let mut _data1: AllDetailClothCollection = AllDetailClothCollection {
        name_cloth: "Snowman Overcoat".to_string(),
        id_cloth: "Cool_win_1".to_string(),
        name_collection: "Be cool in Winter".to_string(),
        id_collection: collection_id.id_collection.to_string(),
        description_cloth: "...Description...".to_string(),
        cost_cloth: 1199,
        type_cloth: "Tops".to_string(),
        material_of_cloth: "Wool".to_string(),
        sex_cloth: "Female".to_string(),
        stock_of_cloth: 42,
        date: "2023-03-20".to_string(),
    };
```

```rust
#[derive(Serialize, Deserialize)]

struct EventidCollection {
    id_collection: String,
}
```

```rust
#[derive(Serialize,Deserialize)]
pub struct AllDetailClothCollection {
    pub name_cloth: String,
    pub id_cloth: String,
    pub name_collection: String,
    pub id_collection: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub type_cloth: String,
    pub material_of_cloth: String,
    pub sex_cloth: String,
    pub stock_of_cloth: i32,
    pub date : String
}
```

# VIEW ONLY CLOTHES'S COLLECTION THAT USER WANT TO BUY

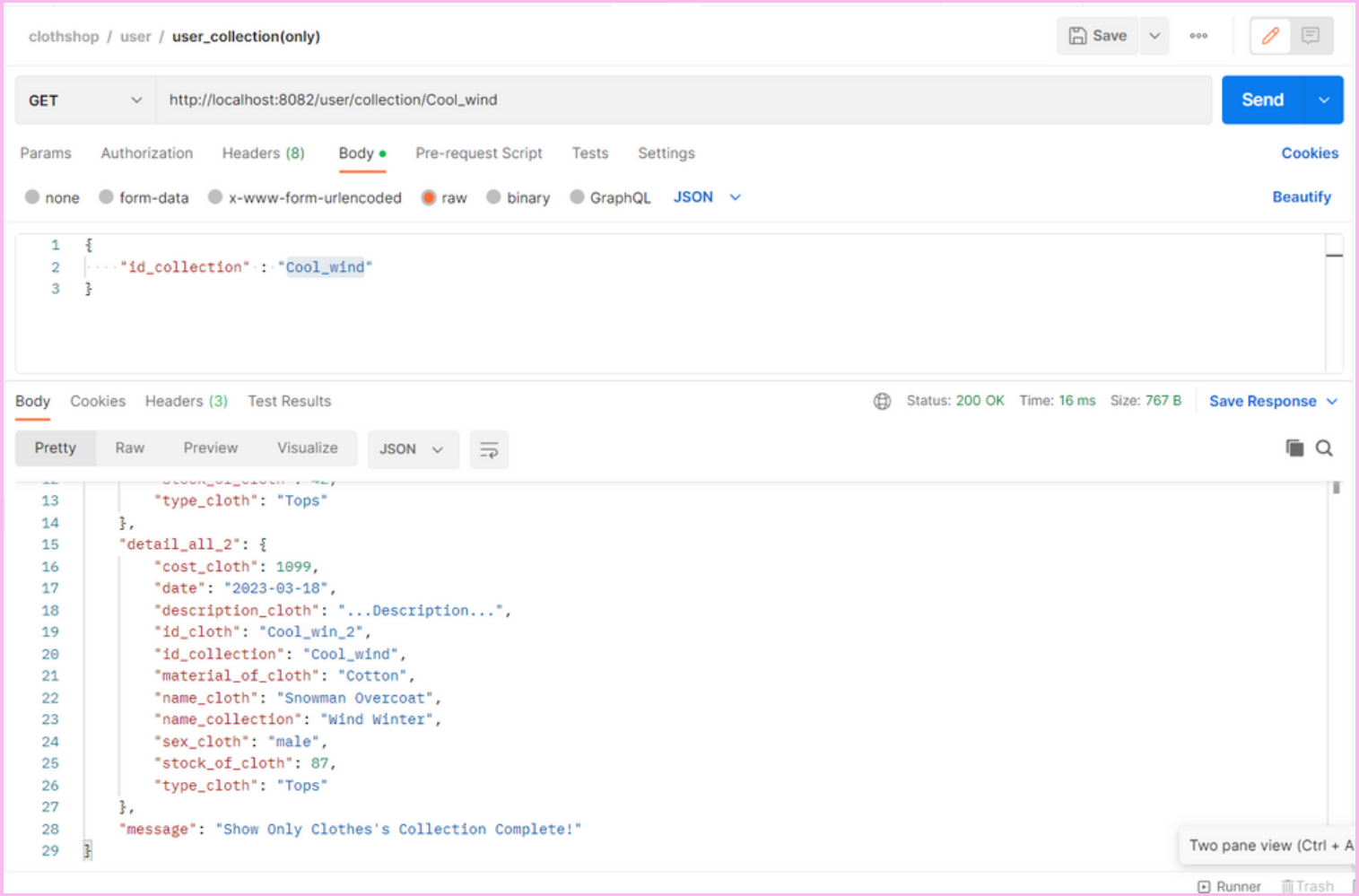
```
118     let mut message: &str = "Show Only Collection";
119     //เช็คว่าid_Collection ว่าurl ตรงกับ bodyไหม
120     if id.to_string() != collection_id.id_collection.to_string() {
121         message = "ID not found or Not Match !!";
122         _data1 = AllDetailClothCollection {
123             name_cloth: "No Data".to_string(),
124             id_cloth: "No Data".to_string(),
125             name_collection: "No Data".to_string(),
126             id_collection: "No Data".to_string(),
127             description_cloth: "...No Data...".to_string(),
128             cost_cloth: 0,
129             type_cloth: "No Data".to_string(),
130             material_of_cloth: "No Data".to_string(),
131             sex_cloth: "No Data".to_string(),
132             stock_of_cloth: 0,
133             date: "No Data".to_string(),
134         };
```

```
#[derive(Serialize,Deserialize)]
pub struct AllDetailClothCollection {
    pub name_cloth: String,
    pub id_cloth: String,
    pub name_collection: String,
    pub id_collection: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub type_cloth: String,
    pub material_of_cloth: String,
    pub sex_cloth: String,
    pub stock_of_cloth: i32,
    pub date : String
}
```

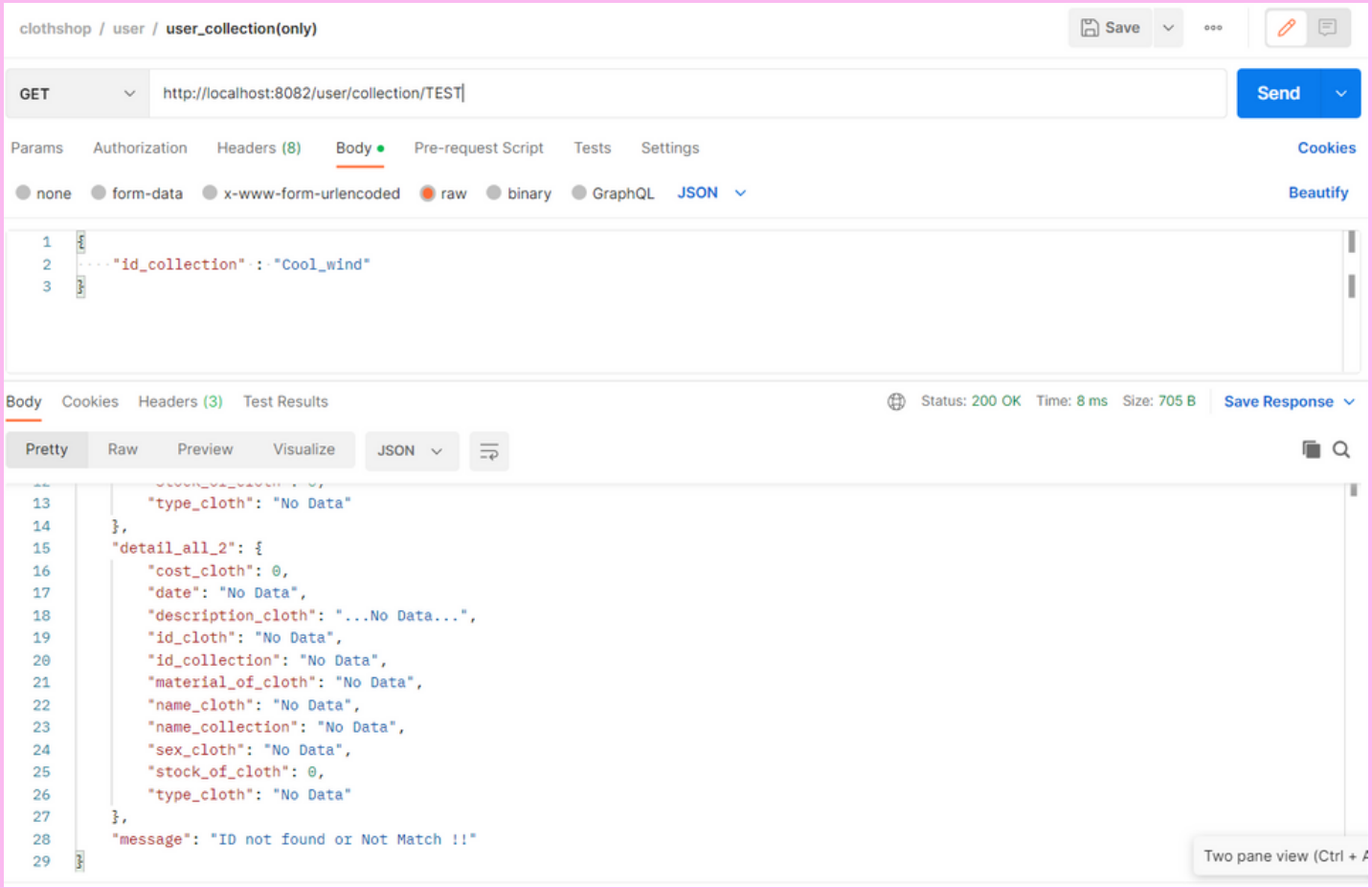
# VIEW ONLY CLOTHES'S COLLECTION THAT USER WANT TO BUY

```
148         } else {
149             message = "Show Only Clothes's Collection Complete!";
150         }
151         #[derive(Serialize, Deserialize)]
152         struct WebResponse {
153             detail_all_1: AllDetailClothCollection,
154             detail_all_2: AllDetailClothCollection,
155             message: String,
156         }
157
158         let web_response: WebResponse = WebResponse {
159             detail_all_1: _data1,
160             detail_all_2: _data2,
161             message: message.to_string(),
162         };
163
164         let response: Value = json!(web_response);
165
166         HttpResponse::Ok().json(response)
167     } fn view_collection_only_user
168
```

# VIEW ONLY CLOTHES'S COLLECTION THAT USER WANT TO BUY



# VIEW ONLY CLOTHES'S COLLECTION THAT USER WANT TO BUY ( ERROR )



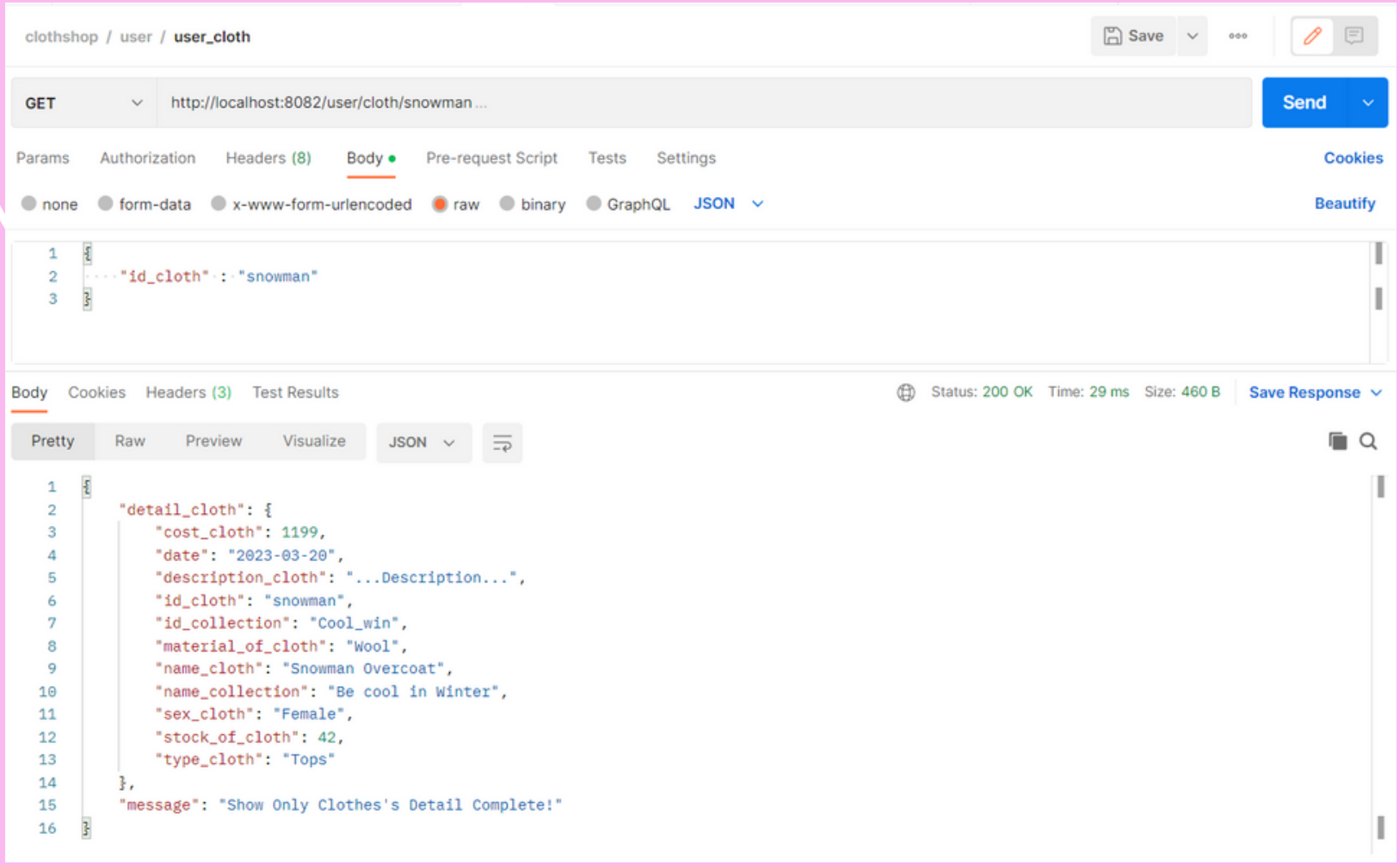
# CLOTH FOR USER

```rust
//************ User ดูแค่ Cloth(เสื้อผ้า) ที่สนใจซื้อเพียง 1 ตัวเท่านั้น **********//

#[get("/user/cloth/{id_cloth}")]
async fn view_cloth_only_user(id: web::Path<String>,input_form: web::Json<EventidCloth>) -> impl Responder {
    info!("Show Only Cloth");

    let id: String = id.to_string();
    let cloth_id: EventidCloth = input_form.into_inner();
    // let id:String = id.to_string();
    let mut _data1: AllDetailClothCollection = AllDetailClothCollection {
        name_cloth: "Snowman Overcoat".to_string(),
        id_cloth: cloth_id.id_cloth.to_string(),
        name_collection: "Be cool in Winter".to_string(),
        id_collection: "Cool_win".to_string(),
        description_cloth: "...Description...".to_string(),
        cost_cloth: 1199,
        type_cloth: "Tops".to_string(),
        material_of_cloth: "Wool".to_string(),
        sex_cloth: "Female".to_string(),
        stock_of_cloth: 42,
        date: "2023-03-20".to_string(),
    };
    let mut message: &str = "Show Only Cloth user want to buy";
    //เช็คว่าid_Collection ว่านurl ตรงกับ bodyไหม
    if id.to_string() != cloth_id.id_cloth.to_string() {
        _data1 = AllDetailClothCollection {
            name_cloth: "No Data".to_string(),
            id_cloth: "No Data".to_string(),
            name_collection: "No Data".to_string(),
            id_collection: "No Data".to_string(),
            description_cloth: "No Data".to_string(),
            cost_cloth: 0,
            type_cloth: "No Data".to_string(),
            material_of_cloth: "No Data".to_string(),
            sex_cloth: "No Data".to_string(),
            stock_of_cloth: 0,
            date: "No Data".to_string(),
        };
        message = "ID not found or Not Match !!";
    } else {
        message = "Show Only Clothes's Detail Complete!";
    }
```

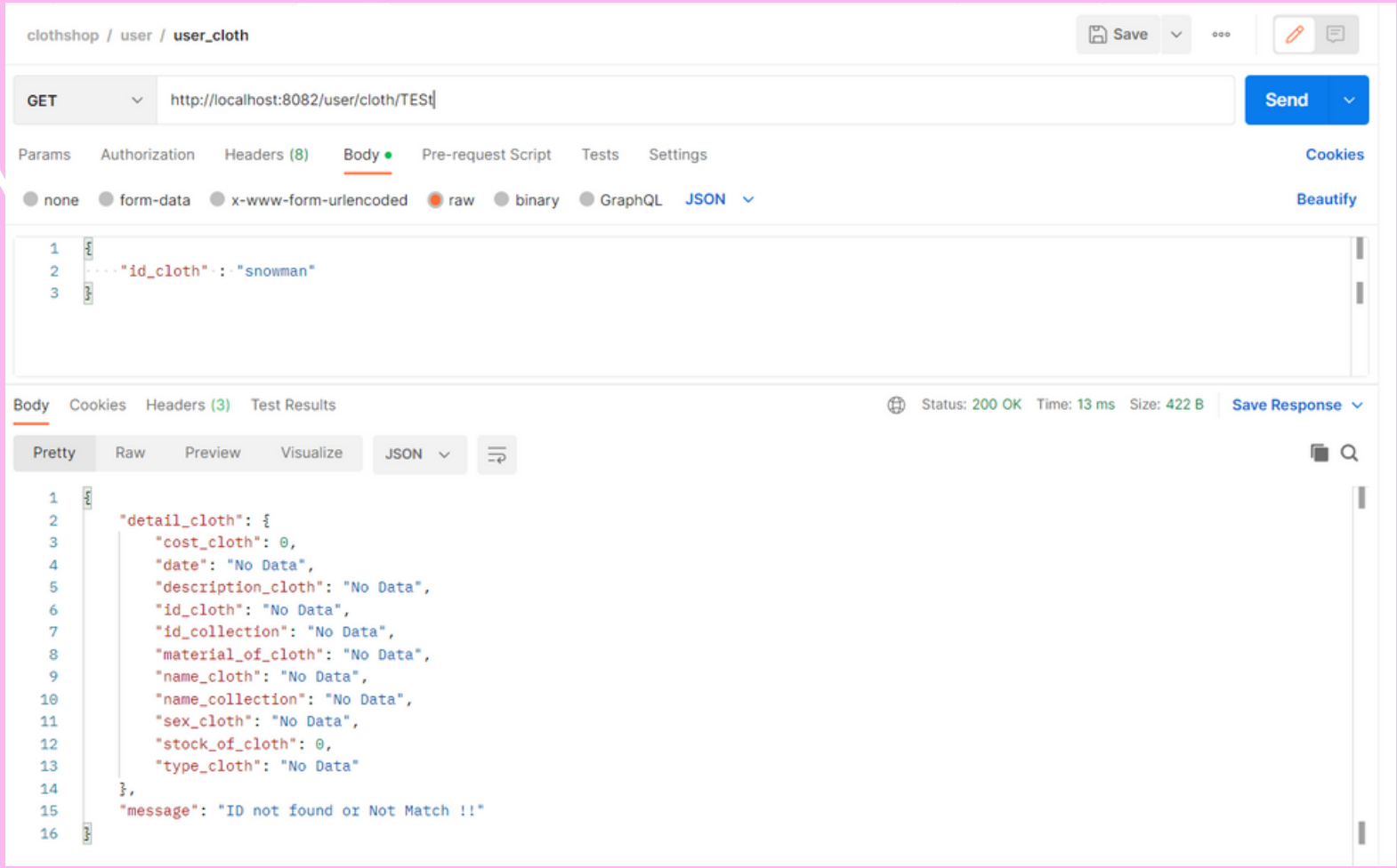
```rust
#[derive(Serialize, Deserialize)]
2 implementations
struct EventidCloth {
    id_cloth: String,
}
```

```rust
#[derive(Serialize,Deserialize)]
pub struct AllDetailClothCollection {
    pub name_cloth: String,
    pub id_cloth: String,
    pub name_collection: String,
    pub id_collection: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub type_cloth: String,
    pub material_of_cloth: String,
    pub sex_cloth: String,
    pub stock_of_cloth: i32,
    pub date : String
}
```

```rust
    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        detail_cloth: AllDetailClothCollection,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        detail_cloth: _data1,
        message: message.to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)
} fn view_cloth_only_user
```


clothshop / user / user_cloth
Save
GET
http://localhost:8082/user/cloth/snowman...
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id_cloth" : "snowman"
}
Body
Cookies
Headers (3)
Test Results
Status: 200 OK
Time: 29 ms
Size: 460 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"detail_cloth": {
"cost_cloth": 1199,
"date": "2023-03-20",
"description_cloth": "...Description...",
"id_cloth": "snowman",
"id_collection": "Cool_win",
"material_of_cloth": "Wool",
"name_cloth": "Snowman Overcoat",
"name_collection": "Be cool in Winter",
"sex_cloth": "Female",
"stock_of_cloth": 42,
"type_cloth": "Tops"
},
"message": "Show Only Clothes's Detail Complete!"
}

# CLOTH FOR USER (ERROR)



# BASKET FOR USER

- View Orders In User's Basket
- Add Clothes Into User's Basket
- Delete Clothes In User's Basket
- Confirm Order In User's Basket

# VIEW ORDERS IN USER'S BASKET

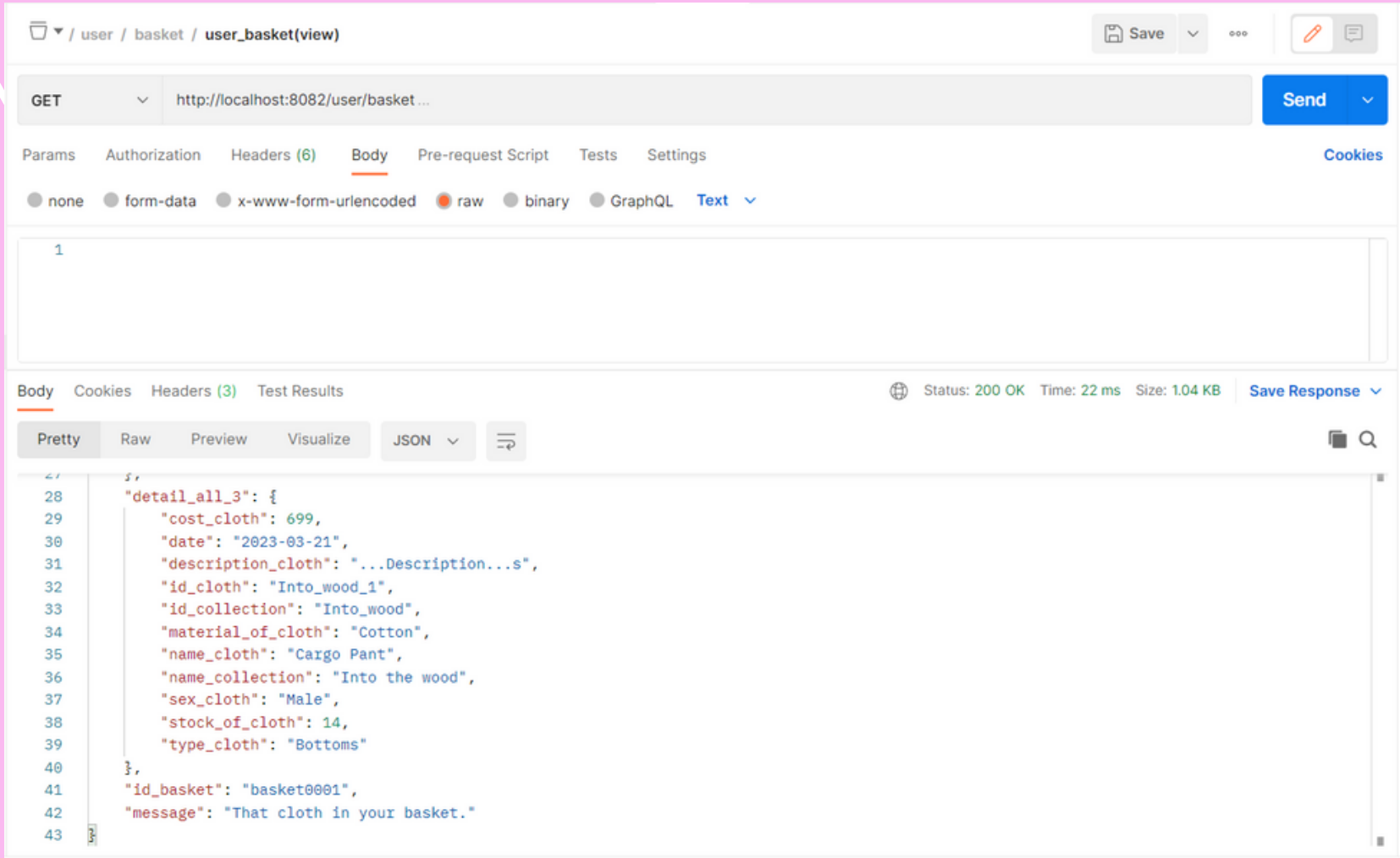
```rust
//**********  User สามารถดูรายการสั่งซื้อสินค้าในตะกร้าได้ **********//

#[get("/user/basket")]
async fn view_order_basket() -> impl Responder {
    info!("Show item in basket");

    let mut _data1: AllDetailClothCollection = AllDetailClothCollection {
        name_cloth: "Snowman Overcoat".to_string(),
        id_cloth: "Cool_win_1".to_string(),
        name_collection: "Be cool in Winter".to_string(),
        id_collection: "Cool_win".to_string(),
        description_cloth: "...Description...".to_string(),
        cost_cloth: 1199,
        type_cloth: "Tops".to_string(),
        material_of_cloth: "Wool".to_string(),
        sex_cloth: "Female".to_string(),
        stock_of_cloth: 42,
        date: "2023-03-20".to_string(),
    };
```

```rust
#[derive(Serialize,Deserialize)]
pub struct AllDetailClothCollection {
    pub name_cloth: String,
    pub id_cloth: String,
    pub name_collection: String,
    pub id_collection: String,
    pub description_cloth: String,
    pub cost_cloth: i32,
    pub type_cloth: String,
    pub material_of_cloth: String,
    pub sex_cloth: String,
    pub stock_of_cloth: i32,
    pub date : String
}
```

# VIEW ORDERS IN USER'S BASKET

```
    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        id_basket: String,
        detail_all_1: AllDetailClothCollection,
        detail_all_2: AllDetailClothCollection,
        detail_all_3: AllDetailClothCollection,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        id_basket: "basket0001".to_string(),
        detail_all_1: _data1,
        detail_all_2: _data2,
        detail_all_3: _data3,
        message: "That cloth in your basket.".to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)
} fn view_order_basket
```

# VIEW ORDERS IN USER'S BASKET

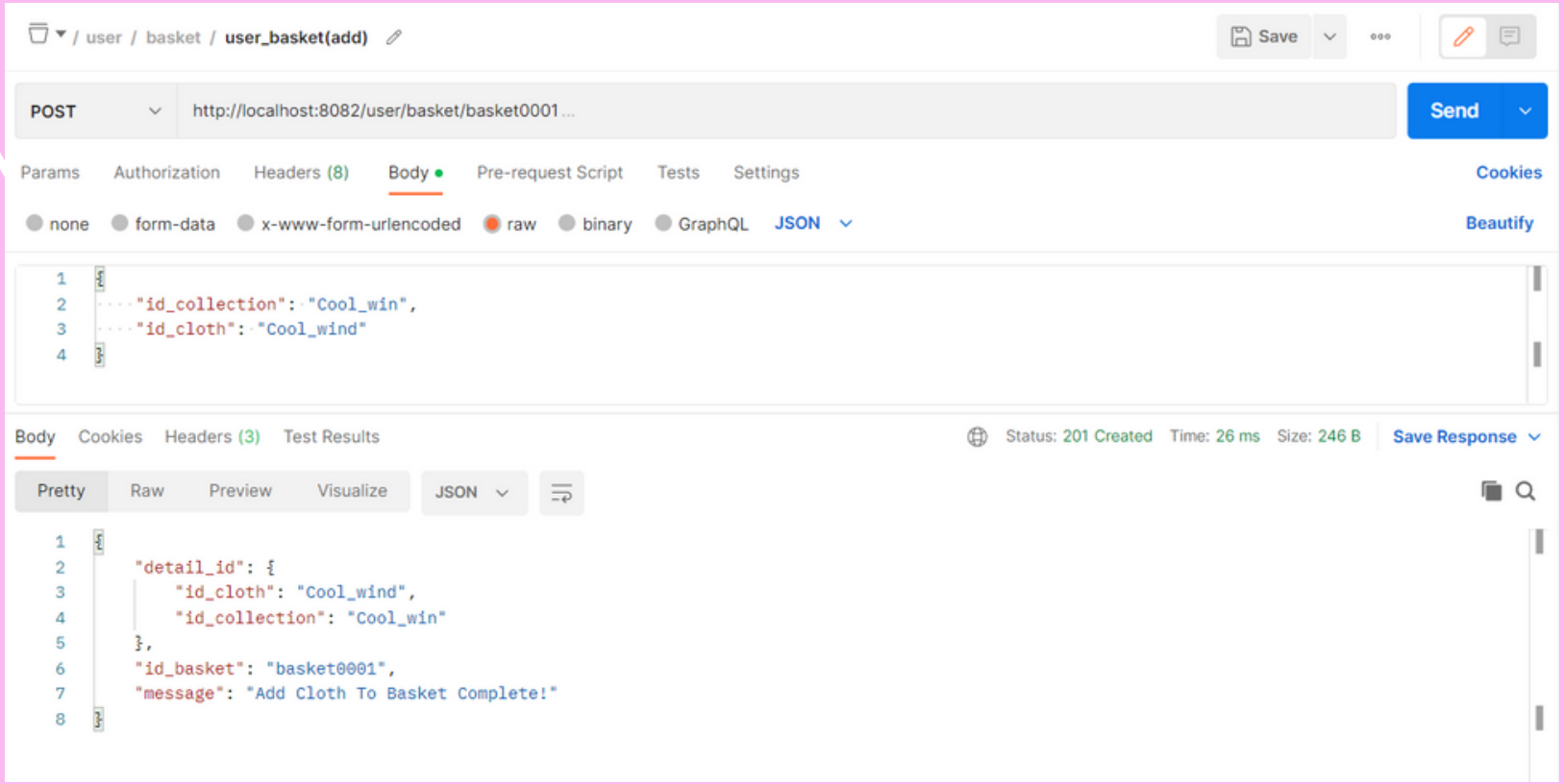


# ADD CLOTHES INTO USER'S BASKET

```rust
// //************* User เพิ่มเสื้อผ้าที่ต้องการซื้อลงในตะกร้า **********************//

#[post("/user/basket/{id_basket}")]
async fn add_order2basket(
    id: web::Path<String>,
    input_form: web::Json<IdClothCollection>,
) -> impl Responder {
    info!("user add Cloth to basket. (http:post)");

    let mut _iddata: IdClothCollection = IdClothCollection {
        id_collection: input_form.id_collection.to_string(),
        id_cloth: input_form.id_cloth.to_string(),
    };

    //สมมติ(Test)[/admin/addition/Test01]

    //รวมกับ message ของระบบเพื่อแจ้งเตือนการแสดงผล

    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        id_basket: String,
        detail_id: IdClothCollection,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        id_basket: id.to_string(),
        detail_id: _iddata,
        message: "Add Cloth To Basket Complete!".to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Created().json(response)
} fn add_order2basket
```

```rust
#[derive(Serialize,Deserialize)]
2 implementations
pub struct IdClothCollection{
    pub id_cloth: String,
    pub id_collection: String,

}
```

# ADD CLOTHES INTO USER'S BASKET

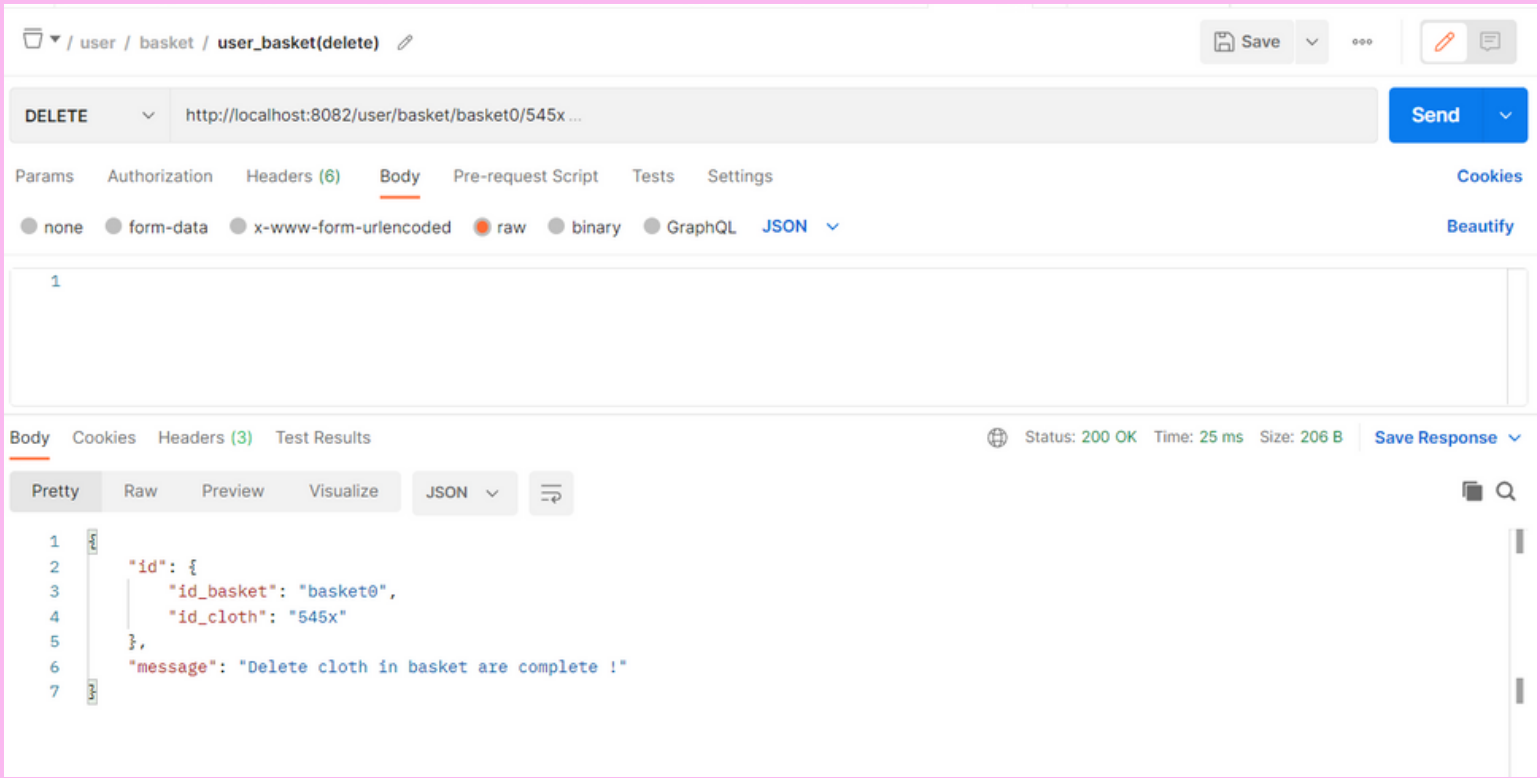


# DELETE CLOTHES IN USER'S BASKET

```rust
//*********** User ลบเสื้อผ้าที่ไม่ต้องการซื้อออกจากตะกร้า **************//

#[delete("/user/basket/{id_basket}/{id_cloth}")]
async fn delete_clothformbasket(
    id: web::Path<Idfordelete>,
) -> impl Responder {
    info!("user delete cloth form basket. (http:delete)");

    let mut _id_delete: Idfordelete = Idfordelete{
        id_cloth : id.id_cloth.to_string(),
        id_basket : id.id_basket.to_string(),
    };

    //สมมติ(Test)[/admin/addition/Test01]

    //รวมกับ message ของระบบเพื่อแจ้งเตือนการแสดงผล

    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        id: Idfordelete,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        id: _id_delete,
        message: "Delete cloth in basket are complete !".to_string(),
    };

    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)
} fn delete_clothformbasket
```

```rust
#[derive(Serialize, Deserialize)]
2 implementations
struct Idfordelete {
    id_basket: String,
    id_cloth:String
}
```

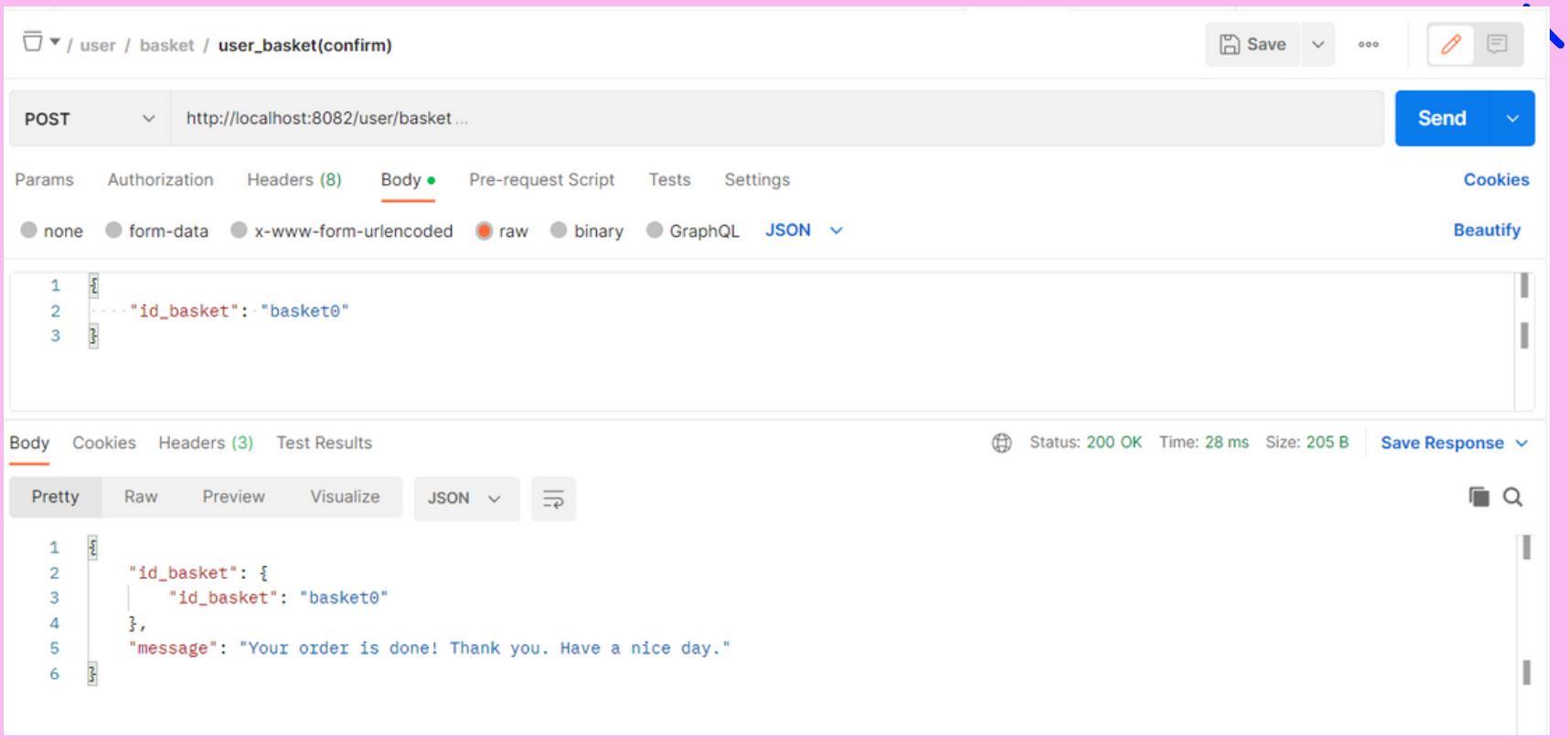
# DELETE CLOTHES IN USER'S BASKET



# CONFIRM ORDER IN USER'S BASKET

```rust
#[derive(Serialize, Deserialize)]
2 implementations
struct Confirmid {
    id_basket: String,
}

//***************** User ตกลงสั่งซื้อสินค้าในตะกร้า S*******************//

#[post("/user/basket")]
async fn confirm_order(input_form:web::Json<Confirmid>) -> impl Responder {
    info!("confirm Clothes's By IDCollection ");

    let id_basket: Confirmid = Confirmid{
        id_basket : input_form.id_basket.to_string()
    };
    #[derive(Serialize, Deserialize)]
    struct WebResponse {
        id_basket: Confirmid,
        message: String,
    }

    let web_response: WebResponse = WebResponse {
        id_basket: id_basket,
        message:"Your order is done! Thank you. Have a nice day.".to_string(),
    };
    let response: Value = json!(web_response);

    HttpResponse::Ok().json(response)

}
```

# CONFIRM ORDER IN USER'S BASKET



# THANK YOU