API Documentation - Krypt Management System (Serverless with Firebase)

This API manages orders with data stored in **Firestore** and integrates **Firebase Authentication** for security. Each function is triggered via HTTP and processes JSON-formatted data.

Database architecture

- The database is designed with the following architecture
- The main document is the orders
- Each child of orders has uuid set as the minted NFT address

Order

- Each order has 4 main children
- Billing info: which contains the address, email, etc
- Created at: date the NFT minted
- Nft metadata: which is the prize and size of the shirt
- State: order state whether active, shipped, or in shipment
- userId: the public key of the user who minted the NFT

Import the query methods

The query methods are stored in src/firebase/query.js

import { saveOrderInfo } from '../../firebase/query';

---

1. Save Order

Method: `POST`
Endpoint: `saveOrderInfo(userId, nftMetadata, billingInfo)`
Description: Creates and stores a new order.
Method Params

```
{
  "userId": "<the user wallet id>",
  "nftMetadata": “{
                nftAddress: <str>,
                Traits: <str>
                Size: <str>
                }",
  “billingInfo”:{
         name:.<str>,
         email: <str>,
         address: <str>,
         city: <str>,
         state: <str>,
         zip: <str>,
}


}
```

Response:

- **201 Created:** Order saved successfully.
- **400 Bad Request:** Invalid data input.

---

2. Get Order by ID

Method: `GET`
Endpoint: `getOrder(orderId)`
Description: Retrieves an order using its ID.
Path Parameter:

- **orderId** (string): Unique order ID to retrieve.

Sample Response:

```
{

    "id": "4fd7R1QJZqX9JcVRqTqBfJxJUAchW745EvHDVTHoyvDR",
    "nftMetadata": {
        "nftAddress":
"4fd7R1QJZqX9JcVRqTqBfJxJUAchW745EvHDVTHoyvDR",
        "size": "Medium",
        "trait": "Gift Box Access"
    },
    "state": "shipped",
    "createdAt": {
        "seconds": 1730229690,
        "nanoseconds": 308000000
    },
    "userId": "9RMbbWjPi9KsG3DLMcC8XVdmmjF9kN61dLw32o2HS6ys",
    "billingInfo": {
        "city": "ikorodu",
        "name": "Samuel Afolabi",
        "zip": "104233",
        "state": "Lagos",
        "email": "info@acornglobal.net",
        "address": "62 Oba Oyefusi Road LA Ikorodu"
    }


}
```

- **200 OK:** Order retrieved.
- **404 Not Found:** Order ID does not exist.

---

3. Get All Orders

Method: `GET`
Endpoint: `fetchOrders()`
Description: Retrieves all saved orders.

Sample Response:

JSON
```
[
{

    "id": "4fd7R1QJZqX9JcVRqTqBfJxJUAchW745EvHDVTHoyvDR",
    "nftMetadata": {
        "nftAddress":
"4fd7R1QJZqX9JcVRqTqBfJxJUAchW745EvHDVTHoyvDR",
        "size": "Medium",
        "trait": "Gift Box Access"
    },
    "state": "shipped",
    "createdAt": {
        "seconds": 1730229690,
        "nanoseconds": 308000000
    },
    "userId": "9RMbbWjPi9KsG3DLMcC8XVdmmjF9kN61dLw32o2HS6ys",
    "billingInfo": {
        "city": "ikorodu",
        "name": "Samuel Afolabi",
        "zip": "104233",
        "state": "Lagos",
        "email": "info@acornglobal.net",
        "address": "62 Oba Oyefusi Road LA Ikorodu"
    }


}
]
```

- **200 OK:** All orders retrieved.

---

3. Update Order State

Method: `POST`
Endpoint: `updateOrderState(orderId, newState)`
Description: Updates order state, whether active, shipped, or shipping

- **200 OK:** order state updated
