Vectre: Sprint 1 Documentation

Table of Contents

Login Flow

1. **(Frontend)** User presses “Connect with Metamask” button @ `*FRONTEND_BASE_URL/login*`
    a. Sends wallet address from Metamask in request body to `*BACKEND_BASE_URL/users/login/nonce*`
2. **(Backend)** `POST */users/login/nonce*` returns user.nonce from User specified by `*walletAddress*`
3. **(Frontend)** Received nonce from backend. Prompts user to sign message containing nonce with Metamask
    a. Sends wallet address signed message from Metamask in request to `*BACKEND_BASE_URL/users/login*`
4. **(Backend)** POST */users/login* verifies signed message with Metamask
    a. Generates JWT (JSON Web Token) for User specified by `*walletAddress*`
    b. Updates *user.nonce* with a (new) randomly generated nonce
    c. Returns JWT token in an HttpOnly cookie (token expires after 7 days)
5. **(Frontend)** Stores JWT cookie
    a. Cookie is sent with all requests to authorize User access to restricted endpoints

Neo4j Setup Queries

User

```
CREATE CONSTRAINT user_properties_walletAddress IF NOT EXISTS
FOR (user:User)
REQUIRE user.walletAddress IS NOT NULL;
CREATE CONSTRAINT user_properties_walletAddress_unique IF NOT EXISTS
FOR (user:User)
REQUIRE user.walletAddress IS UNIQUE;

CREATE CONSTRAINT user_properties_username IF NOT EXISTS
FOR (user:User)
REQUIRE user.username IS NOT NULL;
CREATE CONSTRAINT user_properties_username_unique IF NOT EXISTS
FOR (user:User)
REQUIRE user.username IS UNIQUE;

CREATE CONSTRAINT user_properties_name IF NOT EXISTS
FOR (user:User)
REQUIRE user.name IS NOT NULL;

CREATE CONSTRAINT user_properties_nonce IF NOT EXISTS
FOR (user:User)
REQUIRE user.nonce IS NOT NULL;
```

Post

```
CREATE CONSTRAINT post_properties_postID IF NOT EXISTS
FOR (post:Post)
REQUIRE post.postID IS NOT NULL;

CREATE CONSTRAINT post_properties_postID_unique IF NOT EXISTS
FOR (post:Post)
REQUIRE post.postID IS UNIQUE;

CREATE CONSTRAINT post_properties_timestamp IF NOT EXISTS
FOR (post:Post)
REQUIRE post.timestamp IS NOT NULL;

CREATE CONSTRAINT post_properties_text IF NOT EXISTS
FOR (post:Post)
REQUIRE post.text IS NOT NULL;

CREATE CONSTRAINT post_properties_edited IF NOT EXISTS
FOR (post:Post)
REQUIRE post.edited IS NOT NULL;
```

Node Properties

User

Property	Constraints
id	Default
walletAddress	NOT NULL, UNIQUE
username	NOT NULL, UNIQUE
name	NOT NULL
nonce	NOT NULL
bio	
dashboard	

Post

Property	Constraints
postID	NOT NULL, UNIQUE
author	
text	NOT NULL
imageURL	
edited	NOT NULL
timestamp	NOT NULL

Endpoints

Users

GET /users	
Description	Returns all User nodes
Authentication/ Authorization	None
Request Body	None

Response Body	{
	success: true,
	users: [user1, user2, …]
	}

GET /users/{walletAddress}	
Description	Returns User node with specified walletAddress
Authentication/ Authorization	None
Request Body	None

Response Body	{
	success: true,
	user: queried_user
	}

GET /users/{walletAddress}/posts	
Description	Retrieves all posts made by user with specific wallet address
Authentication/ Authorization	None
Request Body	None

Response Body	{
	success: true,
	posts: [post1, post2, …]
	}
	
	Post object:
	{
	author: wallet address of poster
	text: text contents of post

	imageURL: image contents of post, can be empty string
	edited: boolean, true iff post was updated
	timestamp: UTC timestamp
	}
	
	{
	success: false,
	message: specific error message regarding issue
	}

POST /users/register	
Description	Creates User from request body data
Authentication/ Authorization	None

Request Body	{
	walletAddress: “0x93asdf89uy930304ad”,
	username: “example_username”,
	name: “Example User”,
	bio: “”
	}

Response Body	{
	success: true,
	user: new_user
	}

POST /users/login/nonce	
Description	Returns nonce from User specified in request body
Authentication/ Authorization	None

Request Body	{
	walletAddress: “0x93asdf89uy930304ad”
	}

Response Body	{
	success: true,
	nonce: “938483”
	}

POST /users/login	
Description	Validates signed nonce with Metamask and returns specified User's JWT in a cookie
Authentication/ Authorization	None

Request Body	{
	walletAddress: "0x93asdf89uy930304ad",
	signedNonce: "34809a8dfajdkf"
	}

Response Body	{
	success: true,
	authorizationToken:
	"eyJhbGciOiJIUzI1NiJ9.eyJuYW1lIjoiSm9lIENvZGVyIn0.5dlp7GmziL2QS06sZgK4mtaqv0_xX4oFUuTDh1zHK4U"
	}

Cookies	- "token": "eyJhbGciOiJIUzI1NiJ9.eyJuYW1lIjoiSm9lIENvZGVyIn0.5dlp7GmziL2QS06sZgK4mtaqv0_xX4oFUuTDh1zHK4U" - HttpOnly - Expires in 7 days

GET /users/loggedInUser	
Description	Returns User node logged in with cookie JWT
Authentication/ Authorization	Logged in
Request Body	None

Response Body	{
	success: true,
	user: loggedInUser
	}

GET /users/{walletAddress}/dashboard	
Description	Returns dashboard field of User node with specified walletAddress
Authentication/ Authorization	None
Request Body	None
Response Body	"dashboard": [{ "image_url": "https://uploads-ssl.webflow.com/a19_imagesloaded.jpg", "collection_name": "Doodles", "asset_contract": "0x23e700f9b556651f5c9bead14bd5c63200178b13" "token_id": "3690", }, { "image_url": "https://uploads-ssl.webflow.com/a19_imagesloaded.jpg", "collection_name": "Doodles", "asset_contract": "0x23e700f9b556651f5c9bead14bd5c63200178b13" "token_id": "3692", }, ….]

POST /users/updateDashboard	
Description	Updates dashboard field of User node with specified walletAddress
Authentication/ Authorization	Logged in

Request Body	{
	walletAddress: “0x93asdf89uy930304ad”,
	dashboard: “ [
	{
	"image_url": "https://uploads-ssl.webflow.com/a19_imagesloaded.jpg",
	"collection_name": "Doodles",
	"asset_contract": "0x23e700f9b556651f5c9bead14bd5c63200178b13"
	"token_id": "3690",
	},
	{
	"image_url": "https://uploads-ssl.webflow.com/a19_imagesloaded.jpg",
	"collection_name": "Doodles",
	"asset_contract": "0x23e700f9b556651f5c9bead14bd5c63200178b13"
	"token_id": "3692",
	},
	….
	]
	}

Response Body	{
	"success": true,
	"user": {
	"id": "3",
	"walletAddress":
	"0xD6Fdc7C527844c62e85a76c03e2F2142c82AeDBf",
	"username": "horsekingu",
	"name": "horseking",
	"bio": "coolest kid ever",
	"dashboard": "help update plslslslspslpslpsl"
	}
	}

PUT /users/{walletAddress}/update	
Description	Updates profile properties of User node with specified walletAddress
Authentication/ Authorization	Logged in & walletAddress must match

Request Body	{
	username: “updatedUsername”,
	name: “Updated User”,
	bio: “Cheese pizza is my favourite”
	}

Response Body	{
	success: true,
	Message: “Edit success.”
	}

DELETE /users/{walletAddress}/delete	
Description	Delete User node with specified walletAddress
Authentication/ Authorization	Logged in & walletAddress must match
Request Body	None

Response Body	{
	success: true,
	message: “Deleted User”
	}

GET /users/{walletAddress}/nft	
Description	Returns dashboard field of User node with specified walletAddress
Authentication/ Authorization	None
Request Body	None

Response Body	{
	"success": true,
	"nft": [
	{
	"id": 54200800,
	"num_sales": 0,
	"background_color": null,
	"image_url": "https://lh3.googleusercontent.com/HKXYcDwOb3mitLFHXVfbC-2b1_WaXy3DLpl2vqpCCxAo23fpXQ_1mdXJ8_xUf1N3eMzmU-bS7lC-yIn0ZJYuANtPn7m1v0ZkS0uIqQ",
	"image_preview_url": "https://lh3.googleusercontent.com/HKXYcDwOb3mitLFHXVfbC-2b1_WaXy3DLpl2vqpCCxAo23fpXQ_1mdXJ8_xUf1N3eMzmU-bS7lC-yIn0ZJYuANtPn7m1v0ZkS0uIqQ=s250",
	"image_thumbnail_url": "https://lh3.googleusercontent.com/HKXYcDwOb3mitLFHXVfbC-2b1_WaXy3DLpl2vqpCCxAo23fpXQ_1mdXJ8_xUf1N3eMzmU-bS7lC-yIn0ZJYuANtPn7m1v0ZkS0uIqQ=s128",
	"image_original_url": "ipfs://QmdjeTcbCb7s4BMYTSv8tdA31hrxhfNXX9uB4qf3mxtbJQ/100.png",
	"animation_url": null,
	"animation_original_url": null,
	"name": null,
	"description": "Kibo's collection FINAL",
	"external_link": null,
	"asset_contract": {
	"address": "0xfc89fabb00a3cbfaf8344697c358ff8d52787b2d",
	"asset_contract_type": "non-fungible",
	"created_date": "2022-04-04T15:29:43.936315",
	"name": "Kibo's collection #5",
	"nft_version": "3.0",
	"opensea_version": null,
	"owner": 7205983,
	"schema_name": "ERC721",

```
            "symbol": "Kibo5",
            "total_supply": "0",
            "description": null,
            "external_link": null,
            "image_url": null,
            "default_to_fiat": false,
            "dev_buyer_fee_basis_points": 0,
            "dev_seller_fee_basis_points": 0,
            "only_proxied_transfers": false,
            "opensea_buyer_fee_basis_points": 0,
            "opensea_seller_fee_basis_points": 250,
            "buyer_fee_basis_points": 0,
            "seller_fee_basis_points": 250,
            "payout_address": null
        },
        "permalink":
"https://testnets.opensea.io/assets/rinkeby/0xfc89fabb00a3cbfaf8344697c
358ff8d52787b2d/100",
        "collection": {
            "banner_image_url": null,
            "chat_url": null,
            "created_date": "2022-04-04T15:31:35.118836",
            "default_to_fiat": false,
            "description": null,
            "dev_buyer_fee_basis_points": "0",
            "dev_seller_fee_basis_points": "0",
            "discord_url": null,
            "display_data": {
                "card_display_style": "contain",
                "images": []
            },
            "external_url": null,
            "featured": false,
            "featured_image_url": null,
            "hidden": false,
            "safelist_request_status": "not_requested",
            "image_url": null,
            "is_subject_to_whitelist": false,
            "large_image_url": null,
            "medium_username": null,
            "name": "Kibos collection 5",
            "only_proxied_transfers": false,
            "opensea_buyer_fee_basis_points": "0",
```

```
            "opensea_seller_fee_basis_points": "250",
            "payout_address": null,
            "require_email": false,
            "short_description": null,
            "slug": "kibos-collection-5",
            "telegram_url": null,
            "twitter_username": null,
            "instagram_username": null,
            "wiki_url": null,
            "is_nsfw": false
         },
         "decimals": 0,
         "token_metadata":
"https://opensea.mypinata.cloud/ipfs/QmeHjvYL548iP2skX9u954sqamG2i
jSZ2wnHkF5nYRWwfK/100.json",
         "is_nsfw": false,
         "owner": {
            "user": {
               "username": null
            },
            "profile_img_url":
"https://storage.googleapis.com/opensea-static/opensea-profile/8.png",
            "address": "0x749c89f7f6054c8cd4a982c93e3e05e996bd5c19",
            "config": ""
         },
         "sell_orders": null,
         "seaport_sell_orders": null,
         "creator": {
            "user": {
               "username": null
            },
            "profile_img_url":
"https://storage.googleapis.com/opensea-static/opensea-profile/8.png",
            "address": "0x749c89f7f6054c8cd4a982c93e3e05e996bd5c19",
            "config": ""
         },
         "traits": [],
         "last_sale": null,
         "top_bid": null,
         "listing_date": null,
         "is_presale": false,
         "transfer_fee_payment_token": null,
         "transfer_fee": null,
```

	"token_id": "100" }, "message": "Successfully retrieved NFTs for user with wallet address 0x749C89F7F6054C8CD4a982c93E3E05e996BD5C19" }

Posts

POST /posts/create	
Description	Create a new post for User with specified walletAddress
Authentication/ Authorization	Logged in

Request Body	{
	author: “wallet address of poster”,
	text: “text contents of the post”,
	imageURL: “image contents of the post, can be empty”,
	edited: boolean value (true iff this post was updated),
	timestamp: “UTC timestamp”
	}

Response Body	{
	success: true,
	message: “User created successfully”
	}
	
	{
	success: false,
	message: “Error creating user”
	}

POST /posts/{postID}/update	
Description	Updates an already existing post object with the new information
Authentication/ Authorization	Logged in & walletAddress must match

Request Body	{
	author: “wallet address of poster”,
	text: “text contents of the post”,
	imageURL: “image contents of the post, can be empty”,
	edited: boolean value (true iff this post was updated),
	timestamp: “UTC timestamp”
	}

Response Body	{
	success: true,
	message: “Post updated successfully”
	}
	
	{
	success: false,
	message: error message regarding specific issue
	}