**Complete guide for the NFT marketplace**

**Backlog**

Sprint 1: User Authentication, Wallet Integration, and User Profiles

* User Authentication: Allow users to register, log in, and manage their accounts.
* Wallet Connection: Enable users to connect a single wallet to view and manage their NFTs.
* User Profiles: Users can edit profiles (name, image, bio, social links), view other profiles, and follow/unfollow other users.

Sprint 2: NFT Minting, Collections, and Marketplace

* NFT Minting: Allow users to mint new NFTs by uploading media and defining metadata (name, description, price, tags).
* Collections: Users can create collections and either transfer existing NFTs or mint new NFTs directly within collections.
* Marketplace: Display all available NFTs in a marketplace with filtering and sorting options. Allow users to favorite NFTs and manage their favorites.

Sprint 3: Auctions, Bidding, and Notifications

* Auctions and Bidding: Enable creators to list NFTs in auctions with a reserve price and timer. Other users can place bids on auctioned NFTs.
* Notifications: Notify users of important events (Bid, AuctionEnd, Follow) and display notifications within the user interface.

Sprint 4: Additional Features (Dark Mode, Multi-Language Support, and Analytics)

* Dark Mode & Localization: Implement light/dark mode toggle and prepare the app for potential language translation.
* Analytics & Statistics: Track user activity such as wallet connections, NFT creation, and sales for admin analytics.

**Functional Requirements**

1. **User Registration/Login**: Unique email and username required for account creation.
2. **Wallet Management**: Each user can connect only one wallet.
3. **Profile Management**: Editable fields for name, bio, profile picture, and social links.
4. **NFT Minting**: Allow users to mint NFTs with media, metadata, and pricing details.
5. **Collections Management**: Users can create collections, transfer existing NFTs into them, or mint directly within a collection.
6. **Marketplace**: Display NFTs, allow favoriting, and filter by tags, categories, and sale status.
7. **Auctions and Bidding**: Auction feature for NFTs, with tracking of highest bid, reserve price, and bid history.
8. **Notifications**: Notifications for specified types (Bid, AuctionEnd, Follow) and ability to mark as read.
9. **UI Features**: Light/dark mode and support for multi-language UI (if future translation is implemented).

**MongoDB Schema Design**

**UserSchema**

const UserSchema = **new** mongoose.Schema({

    name: { type: *String*, required: true },

    image: *String*,

    bio: *String*,

    socialLinks: [{

      platformName: *String*,

      platformIcon: *String*,

      link: *String*

    }],

    followers: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],

    blockchainAddress: { type: *String*, unique: true, sparse: true },

    email: { type: *String*, unique: true, required: true },

    username: { type: *String*, unique: true, required: true },

    password: { type: *String*, required: true },

    createdNFTs: [{ type: mongoose.Schema.Types.ObjectId, ref: 'NFT' }],

    ownedNFTs: [{ type: mongoose.Schema.Types.ObjectId, ref: 'NFT' }],

    favoriteNFTs: [{ type: mongoose.Schema.Types.ObjectId, ref: 'NFT' }],

    collections: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Collection' }],

    resetPasswordToken: *String*,

    notifications: [{

      type: { type: *String*, enum: ['Bid', 'AuctionEnd', 'Follow'] },

      message: *String*,

      read: { type: *Boolean*, default: false },

      createdAt: { type: *Date*, default: *Date*.now }

    }],

    role: { type: *String*, enum: ['user', 'admin'], default: 'user' },

    walletConnectedAt: *Date*,

    createdAt: { type: *Date*, default: *Date*.now },

    updatedAt: { type: *Date*, default: *Date*.now }

  }, { timestamps: true });

**NFTSchema**

const NFTSchema = **new** mongoose.Schema({

    name: { type: *String*, required: true },

    creator: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },

    description: *String*,

    price: { type: *Number*, required: true },

    highestBid: *Number*,

    auction: {

      active: { type: *Boolean*, default: false },

      timer: *Date*,

      reservePrice: *Number*,

      highestBid: *Number*,

      bidHistory: [{

        amount: *Number*,

        bidder: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },

        timestamp: { type: *Date*, default: *Date*.now }

      }]

    },

    details: {

      etherscanLink: *String*,

      openseaLink: *String*

    },

    tags: [*String*],

    category: *String*,

    mediaUrl: *String*,

    mediaType: { type: *String*, enum: ['image', 'video', 'audio'] },

    transactionHistory: [{

      from: *String*,

      to: *String*,

      timestamp: *Date*,

      transactionId: *String*

    }],

    saleStatus: { type: *String*, enum: ['listed', 'sold', 'onAuction'], default: 'listed' },

    createdAt: { type: *Date*, default: *Date*.now },

    updatedAt: { type: *Date*, default: *Date*.now }

  }, { timestamps: true });

**CollectionSchema**

const CollectionSchema = **new** mongoose.Schema({

    name: { type: *String*, required: true },

    creator: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },

    description: *String*,

    coverImage: *String*,

    nfts: [{ type: mongoose.Schema.Types.ObjectId, ref: 'NFT' }],

    visibility: { type: *String*, enum: ['public', 'private'], default: 'public' },

    createdAt: { type: *Date*, default: *Date*.now },

    updatedAt: { type: *Date*, default: *Date*.now }

  }, { timestamps: true });

**Review for Completeness**

After aligning all elements, here’s a quick checklist:

1. **User Registration/Login** ✅
2. **Single Wallet Connection** ✅
3. **Profile Management** ✅
4. **NFT Minting and Metadata** ✅
5. **Collections (Transfer and Mint within)** ✅
6. **Marketplace with Favorites** ✅
7. **Auction and Bidding Features** ✅
8. **Notifications (Limited to Specified Types)** ✅
9. **UI Features (Dark Mode, Language Support)** ✅