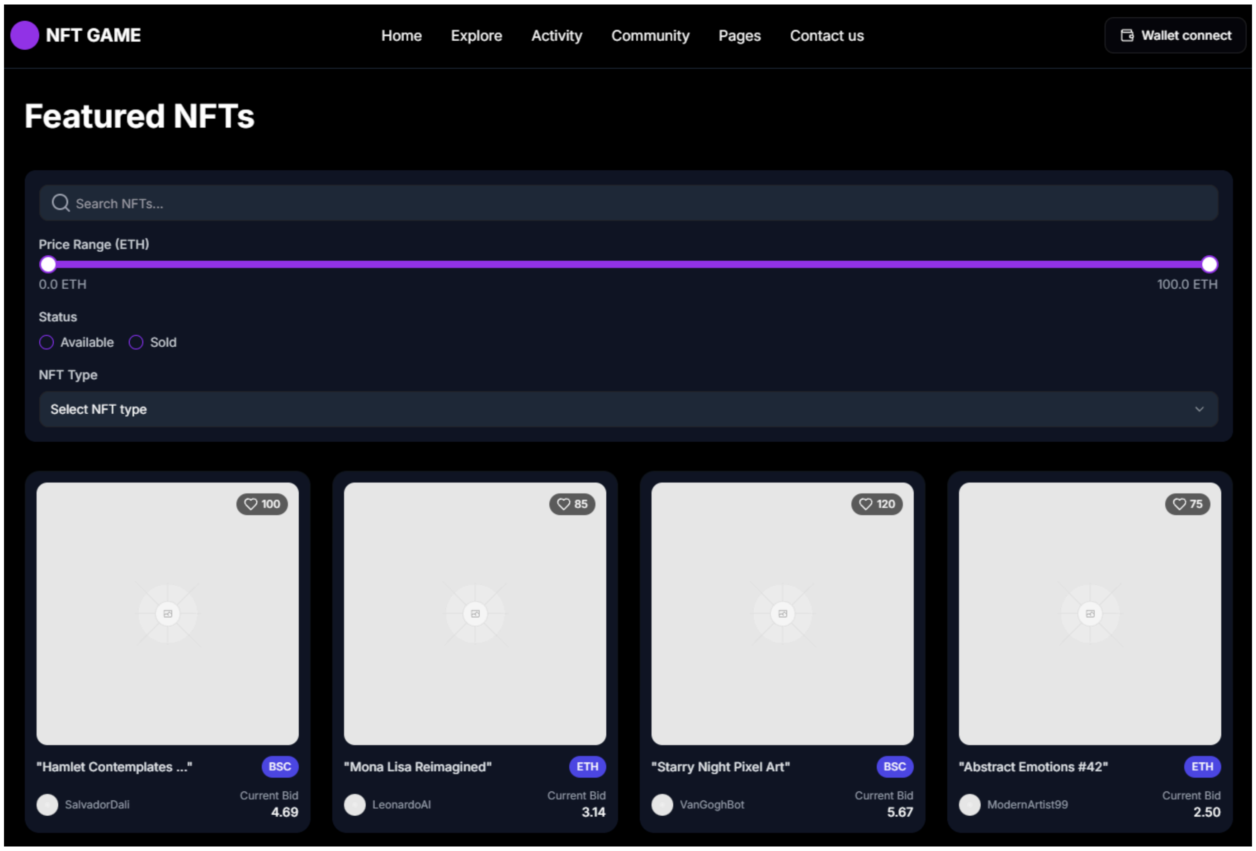
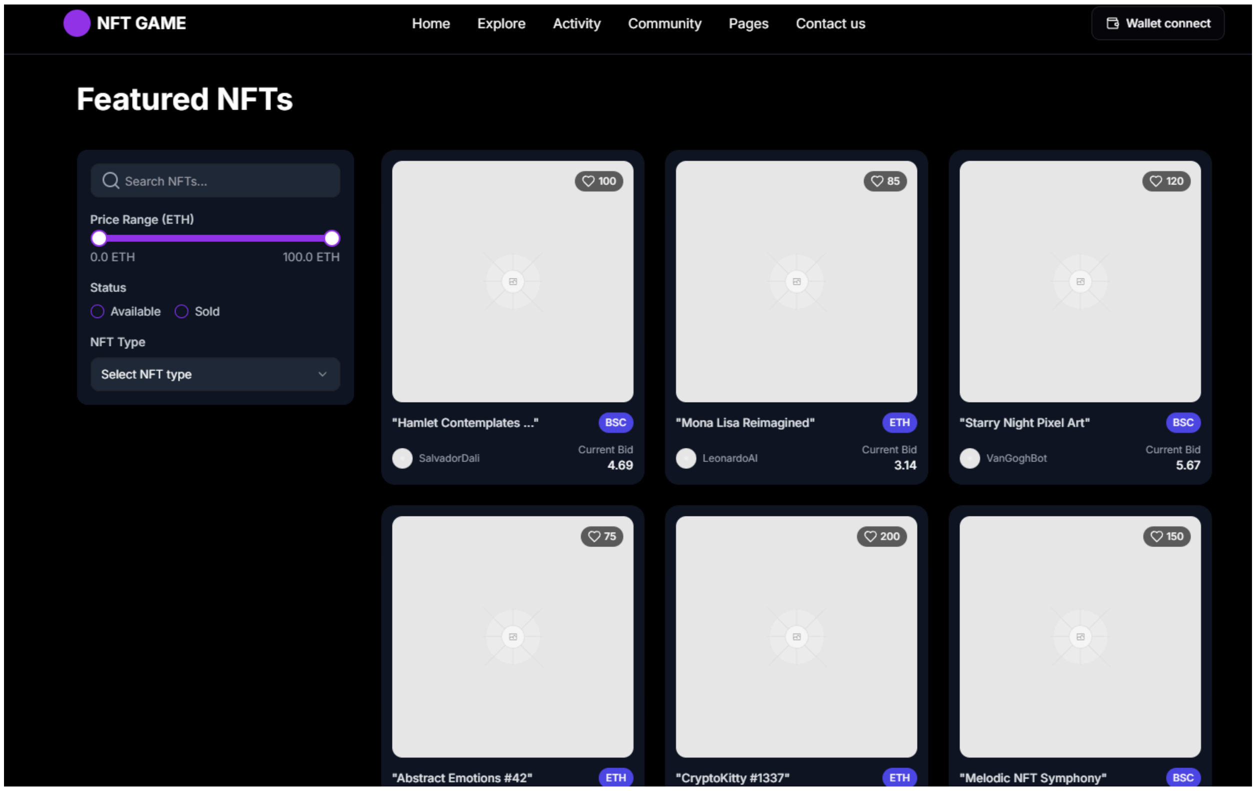
Prompts

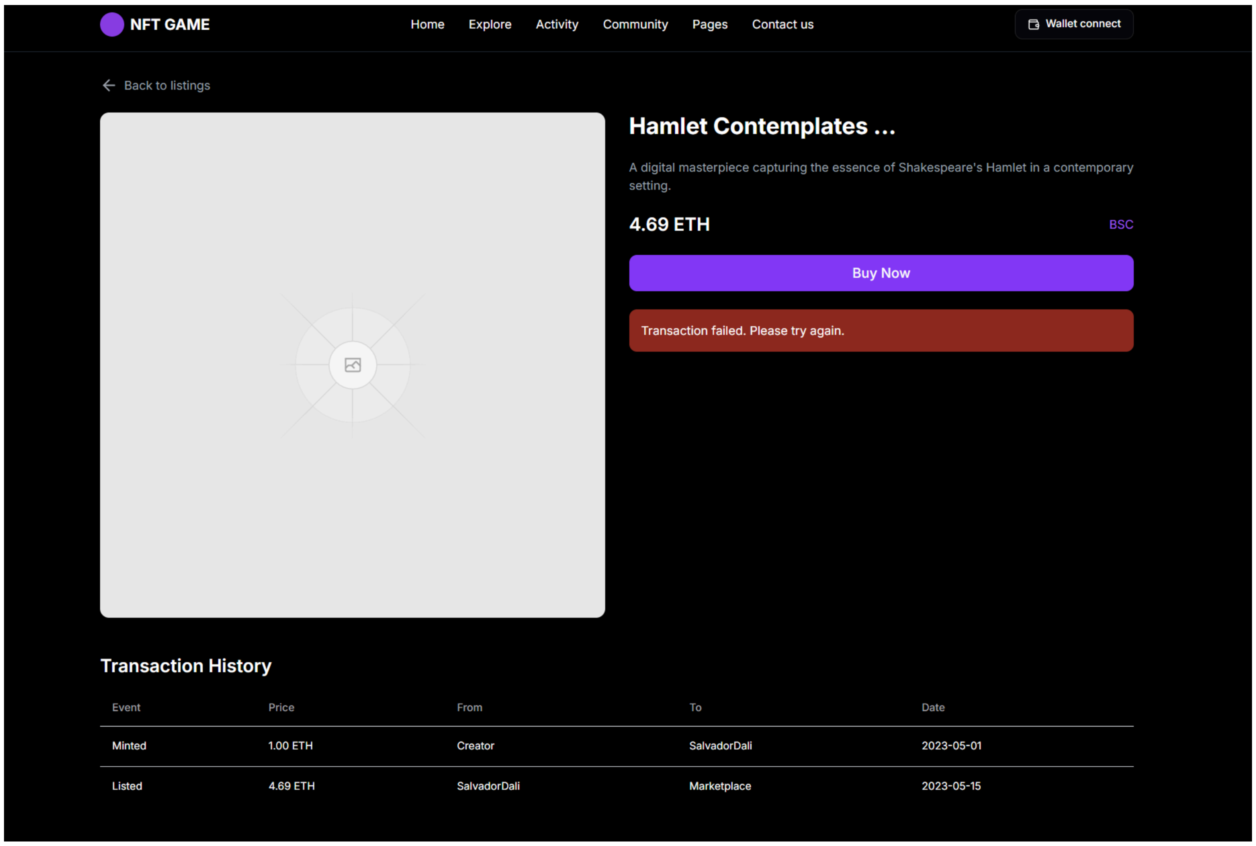
```
Design a modern and responsive NFT marketplace filter component.
- Include a search bar to filter NFTs by name.
- Provide filter options for:
  - Price range with a dual-handle slider (two circular handles to select min and m
  - Status (available/sold).
  - NFT type (dropdown or checkbox selection).
- Ensure a clean, minimal UI with Tailwind CSS.
- Display filter results dynamically with smooth animations.
- Make it mobile-friendly with a compact design.
- Use Next.js and React best practices.
```



```
Update the NFT marketplace UI layout to position the filters and NFT items side b

- Left column (Filters):
  - Search bar to filter NFTs by name.
  - Price range filter with a dual-handle slider (two circular handles for min/max s
  - Status filter with radio buttons ("Available", "Sold").
  - NFT Type filter using a dropdown or checkboxes.

- Right column (NFT Items Grid):
  - Display NFT cards in a 3-column grid (adjustable for responsiveness).
  - Each NFT card should include:
    - NFT image placeholder.
    - NFT name.
    - Blockchain type (ETH/BSC badge).
    - Creator name.
    - Current bid price.
    - Favorite count with a small heart icon.



- Make it fully responsive:
  - On smaller screens, the filters should collapse into a top dropdown or sidebar.
  - NFT grid should adjust to 2-column or 1-column layout on mobile.
```



```
Create an NFT Detail Page for an NFT Marketplace built with Next.js. The page s
 A detailed view of a selected NFT with its image, name, description, price, and s
 A transaction history section listing previous (simulated or API-fetched) transact
 A 'Buy' button that simulates a purchase by transferring tokens from the buyer's
 Wallet integration displaying connected wallet information (address and token ba
Responsive design using Tailwind CSS.
```



Hamlet Contemplates ...

A digital masterpiece capturing the essence of Shakespeare's Hamlet in a contemporary setting.

4.69 ETH

Transaction failed. Please try again.

Transaction History

Event	Price	From	To	Date
Minted	1.00 ETH	Creator	SalvadorDali	2023-05-01
Listed	4.69 ETH	SalvadorDali	Marketplace	2023-05-15

Create storage

```
Write a zustand store using TypeScript for an NFT Marketplace project. The store

- Wallet Integration:
  - walletAddress: a string or null.
  - tokenBalances: an object where each key is a token name and each value is th

- Favorites List:
  - favorites: an array of favorite NFT IDs (type string[]).

- Transaction History:
  - transactionHistory: an array of transactions, where each transaction has the fo
    - id: string
    - buyer: string
    - seller: string
  - amount: number
  - timestamp: number

The update functions in the store must include:
- setWalletAddress(address: string | null): update the wallet address.
- updateTokenBalances(balances: { [token: string]: number }): update token bala
- addFavorite(nftId: string): add an NFT to the favorites list if it is not already pres
- removeFavorite(nftId: string): remove an NFT from the favorites list.
- addTransaction(transaction: Transaction): add a transaction to the transaction h
- resetStore(): reset all user information.

Also, define the corresponding TypeScript interfaces for Transaction, TokenBala
```

Summary

I start by building the basic skeleton of the UI. Once the overall structure is in place, I focus on how the individual components will interact. For this, I use prompts to construct the data store first based on how the data is processed and utilized and only then do I write prompts to develop detailed components in a logical and coherent manner.

During the development process, since some of the newer libraries weren't fully updated, I had to let the AI learn from the new information I provided and then apply that knowledge. In some cases, I handled things manually for example, configuring the wallet connection and Web3 interactions.