PayByMint Candidate Assessment (5/14/25)

Candidate: **Ferdinand Vincent Tabora**

1. Metamask logo issue

There was a problem with the Metamask logo appearance on the wallet connection dialog box.

The asset (logo) was pointing this url:
https://github.com/MetaMask/brand-resources/raw/master/SVG/metamask-fox.svg

It is no longer valid and thus, the image logo appears with a broken link. Metamask provides another link where logos can be found.

I have changed it to point to a new link created by Metamask to point to their brand logo.

https://images.ctfassets.net/clixtyxoaeas/4rnpEzy1ATWRKVBOLxZ1Fm/a74dc1eed36d23d7ea6030383a4d5163/MetaMask-icon-fox.svg

This was specified in the Metamask Github link, that points to a new url for brand assets (logos):

https://metamask.io/assets

The Metamask link to their brand assets allows developers to download a logo pack which comes with zipped SVG format files. These can be used on the website instead of using the asset logo link from Metamask's own website.

I made changes to the file:

/src/store/actions/user.js

Here is the modified code:

```
display: {
        logo:
"https://images.ctfassets.net/clixtyxoaeas/4rnpEzy1ATWRKVBOLxZ1Fm/a74dc1eed36d23d7ea6030383a4d5163/MetaMask-icon-fox.svg",
        name: "MetaMask",
        description: "Connect with MetaMask in your browser",
      }
```

Another option is to copy one of the logos to an assets folder in the project folder and use the path as the link. That way, if the link is changed it will not directly affect the application's wallet connection dialog.

2. Previous arrow button overlaps the wallet connection dialog box

The wallet connection dialog box is a modal that must remain on top or in front of other page elements. Using the z-index CSS property, helps control the stacking order of elements on a webpage.

The fix for this problem is to give a lower z-index value to the button so that it renders below other elements on the page. We apply a z-index hierarchy that ensures buttons stay below modals. This requires absolute positioning and image handling for proper rendering.

The component affected was:

EffectCardSwiper (/src/components/EffectCardSwiper/EffectCardSwiper.js)

This is under the container div class:

<div className="relative w-[80vw] md:w-fit" style={{ zIndex: 1 }}>

The style property style={{ zIndex: 1 } was added.

The image element for arrow-prev (the button overlapping with the wallet connection dialog) was modified:

```
<Image
        src="/assets/icons/arrow-prev.svg"
        alt="arrow-prev"
        width={32}
        height={24}
        style={{ position: "relative" }}
      />
```

The line style={{ position: "relative" }} was added.

Likewise the image element for arrow-next was also modified.

```
<Image
        src="/assets/icons/arrow-next.svg"
        alt="arrow-next"
        width={32}
        height={24}
        style={{ position: "relative" }}
      />
```

The line style={{ position: "relative" }} was also added like in arrow-prev..
Finally, the **EffectCardSwiper.module.css** (also located in same folder as EffectCardSwiper.js) was re-coded to the following:

```
.swiperButtonPrev,
.swiperButtonNext {
  z-index: 10;
  position: absolute;
```

```
  top: 50%;
  transform: translateY(-50%);
  cursor: pointer;
  background: rgba(0,0,0,0.5);
  border-radius: 50%;
  width: 40px;
  height: 40px;
  display: flex;
  align-items: center;
  justify-content: center;
}

.swiperButtonPrev {
  left: -40px;
}

.swiperButtonNext {
  right: -40px;
}
```

In the key changes made, I ensured that buttons stay below dialogs with a z-index of 10. Positioning is also set to absolute so that it does not move around during rendering, and stays below any stacking order.

Final Result

The modifications made to the code provides a fix to the two critical issues. As a result we get the following output screens in the following section.

The Metamask logo now appears correctly and the previous arrow button no longer overlaps with the wallet connection dialog.

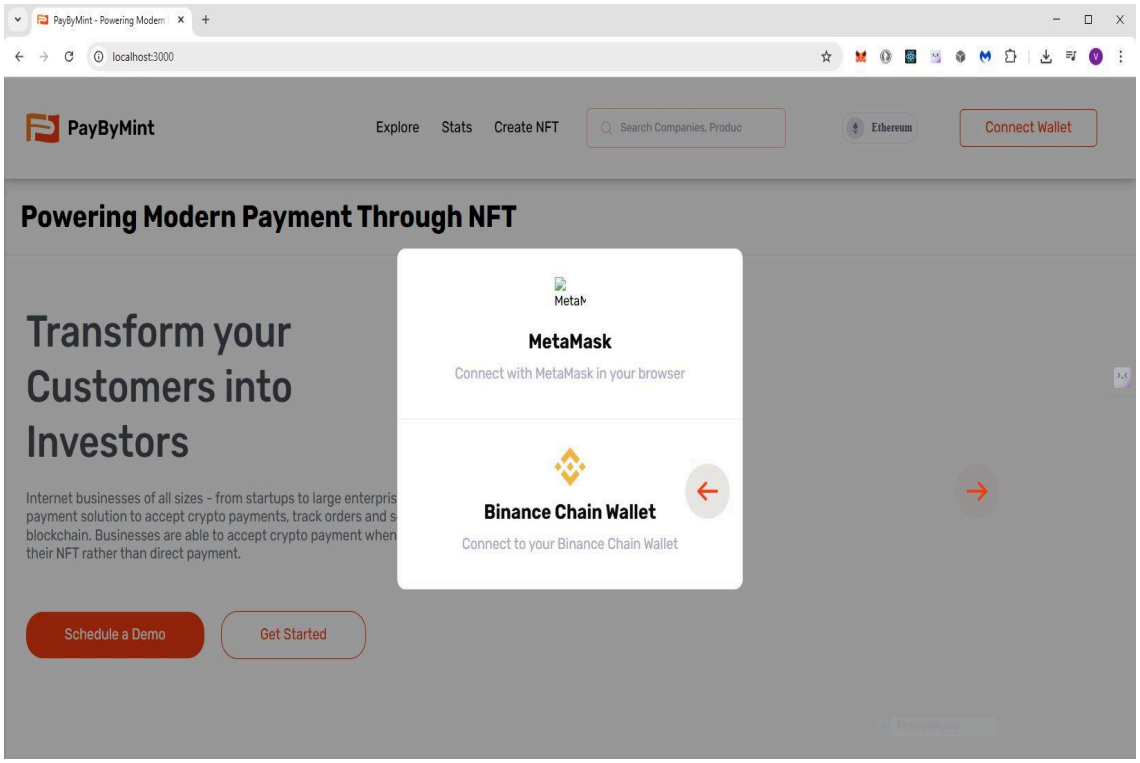

Figure A. This was before the code was modified.

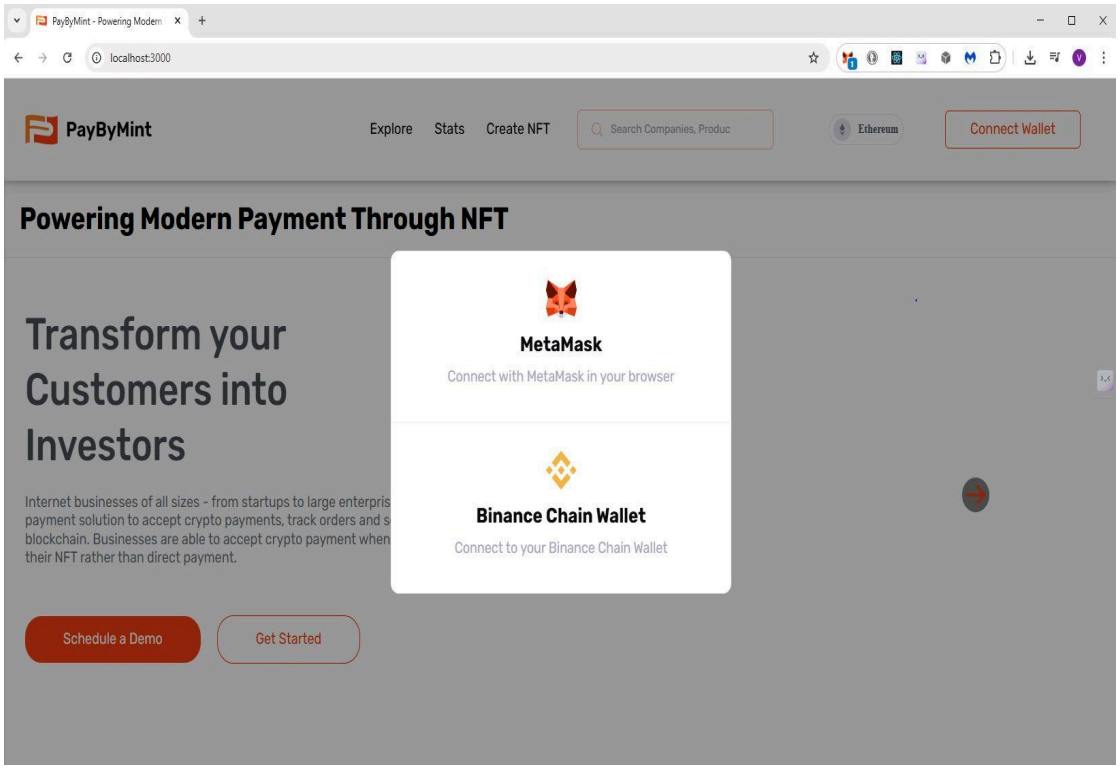

Figure B. This was after the code was modified.

Roadmap

To meet the vision to expand this project into a fully functional NFT marketplace, comparable to Magic Eden, the challenge is with the backend integrations.

I have noticed (but not fully tested due to time constraints) the following features on the frontend:

- *Wallet Connection*: Users can connect their Metamask and Binance wallet to make purchases.
- *NFT Creation*: A way to tokenize digital content, add metadata, and mint as NFT.
- *Simple UI*: Clean interface with React UI components for a more responsive and fast rendering, especially if used from a mobile app (still can be improved).
- *Gallery to display user's NFTs*: Important to showcase creations from digital artists and graphics content creators.

Since the app has been already built we can begin a roadmap to further its development. I think the following enhancements can help to create a platform that can provide a way to allow digital content creators and interested buyers to interact.

Part 1

This section will focus on interface and features enhancements on users.

- More improved design for the aesthetics of the app. The look should be appealing to both digital creators and NFT collectors.
- More blockchain integration through connectivity from other wallets, like wallet connect services, in order to support more users.
- Implement simple buy/sell functionality with buttons
- Add loading states and more or better error handling which is important in order to prevent loss of digital assets.
- Include transaction history to make sure that buyers see exactly what they have been purchasing on the platform, while sellers can see what they have been

selling. This would be similar to OpenSea.

- Add categories or tags for NFTs or queries with an index, to allow users to search for different asset types, collectibles, rarities, and trendsetters.

- Display market listings of popular collections and their price value, volume trade, and number of sales. The more indicators that can be added the more information is provided for users.

- Creation of login for user/admin. This will provide dashboards where users can manage all their transactions and assets.

- Provide escrow contracts if users need a third-party for a secure trade or arbitration. Ideal for high-value NFTs.

- Add a slideshow of popular and upcoming NFTs with a preview.

Part 2

Next there must be thorough testing and audits.

- Test wallet connections

- Test all minting functionalities

- Test IPFS and metadata

- Test UI/UX for consistency

- Test all dashboards

- Test overall security (backend, frontend, digital certificates, etc.)

- Test for load balancing and infrastructure stability

Part 3

Deployment to EVM-blockchain (Ethereum) and production.

- Choose appropriate backend platform (e.g. AWS, Azure, etc.)

- Launch the frontend app on the web and mobile version.
- Use an orchestration tool like Kubernetes or a different approach to manage the deployment and provisioning of systems for the frontend.
- Deploy smart contracts by framework (Hardhat or Foundry).
- Secure SSL connection.
- Legal compliance for matters like royalty payments and high-value trades.

Part 4

These are considerations for the future.

- Multi-chain integration with Solana and other blockchains.
- Establishing a token for rewards or digital governance.
- NFT collateralization for lending and borrowing.
- Web3 gaming integration of NFT assets.
- Metaverse avatar NFT to access accounts and for authorization of digital identity.
- AI curation of NFT assets on the marketplace for tracking and associations.

The focus should be on creation of an MVP app with enhanced UI/UX and ease of use. A stable backend and simple frontend will support the marketplace so it can scale and handle multiple users.

The future offers many possibilities to explore in the Web3 and DeFi space which I outline. The opportunities would be worth exploring after the main product has been built and put into production.