Building out your PolyKye Onchain system requires several steps across smart contract development, off-chain processing, and frontend creation. Here's a comprehensive plan to implement the complete system:

Smart Contract Development

1. **Create the PolyKye Smart Contract**
    - Open Remix IDE (https://remix.ethereum.org)
    - Create a new Solidity file named `PolyKye.sol`
    - Implement the core functions:
        - `submitTarget`: Records disease targets
        - `submitResult`: Stores ligands with IPFS references
        - `getResult`: Retrieves result data
    - Define necessary data structures (structs, mappings) and events
    - Compile the contract in Remix
2. **Deploy the Smart Contract**
    - Connect MetaMask to your chosen network (testnet or mainnet)
    - In Remix, select "Injected Provider" in the deployment tab
    - Deploy the contract and save the contract address

Off-Chain Processing System

3. **Create Event Listener Service**
    - Set up a Node.js project for the listener
    - Install required packages: `ethers`, `axios`, etc.
    - Create a script that connects to your contract
    - Implement event listeners for `TargetSubmitted` events

- Set up error handling and reconnection logic

4. **Implement IPFS Integration**
    - Create functions to upload synthesis pathways to IPFS
    - Use Pinata, Infura, or another IPFS service
    - Ensure proper pinning for data persistence
5. **Build PolyKye Processing Pipeline**
    - Implement the core AI/ML processing logic
    - Create a function to process targets and generate:
        - Optimal ligand (SMILES string)
        - Synthesis pathway
        - Score/metadata
    - Set up a system to handle the processing queue
6. **Create Result Submission Service**
    - Implement a function to submit results back to the blockchain
    - Include proper error handling and transaction management
    - Set up a system to track job status

Frontend Application

7. **Set Up React Project**
    - Create a new React project using Vite:

      text

```
npm create vite@latest polykye-frontend -- --template react
cd polykye-frontend
npm install
```

    -

- Install necessary packages:

  text

```
npm install ethers axios react-router-dom
```

8. **Implement Core Components**
   - Create a wallet connection component
   - Build a target submission form
   - Implement a results display component
   - Add an IPFS content viewer
9. **Connect Frontend to Blockchain**
   - Integrate ethers.js for contract interaction
   - Implement functions to call contract methods
   - Set up event listeners for real-time updates
10. **Add User Interface Features**
    - Create a dashboard for submitted targets
    - Implement a results explorer
    - Add visualization for molecular structures
    - Include IPFS link handling

Integration and Deployment

11. **Set Up Continuous Integration**
    - Create a deployment pipeline for the frontend
    - Set up monitoring for the event listener service
    - Implement logging for all components

12. **Deploy the Complete System**
    - Deploy the event listener to a server
    - Host the frontend on a service like Vercel or Netlify
    - Set up environment variables for sensitive data
13. **Test End-to-End Workflow**
    - Submit a test target through the frontend
    - Verify the event is captured by the listener
    - Confirm the processing pipeline executes correctly
    - Check that results are properly stored on IPFS
    - Verify the blockchain transaction for result submission
    - Test the result retrieval through the frontend

Optional Enhancements

14. **Implement Automated Processing**
    - Create a script to automatically process the top 100 diseases
    - Store results as example data
15. **Add User Authentication**
    - Implement a system to track user contributions
    - Create a reputation system based on submission quality
16. **Improve Result Visualization**
    - Add 3D molecular visualization
    - Implement interactive synthesis pathway diagrams

By following these steps, you'll build a complete PolyKye Onchain system that allows users to submit disease targets, processes them through your AI platform, stores detailed results on IPFS, and records essential data on the blockchain for transparency and provenance.