# Technical Specification (TZ): Real Estate Fractional Web3 Platform

The project "RealEstate Fractional" is a Web3-based decentralized platform enabling fractional ownership of real estate through blockchain technology. Each property is tokenized using the ERC-1155 standard, allowing it to be divided into 100 equal ownership shares. Users can purchase these shares using Ethereum (ETH), making property investment accessible, transparent, and secure.

## 1. Objective

The goal is to create a decentralized real estate investment platform using smart contracts, allowing fractional ownership and transparent transactions recorded on the Ethereum blockchain.

## 2. Core Functionality

1. Smart contract development (ERC-1155) to handle fractional property tokens.  
2. Tokenization of real estate assets with metadata stored via IPFS.  
3. Purchasing shares of properties using ETH.  
4. Withdrawal of funds by property creators (owners).  
5. Web3 frontend interaction through Wagmi, Viem, and RainbowKit.  
6. Integration of AI Assistant (Gemini API) with fallback pattern matching.  
7. Visualization of investment statistics via interactive charts.

## 3. Smart Contract Logic (ERC-1155)

- createProperty(): Register a new property as an ERC-1155 token.  
- purchaseShares(): Buy fractional shares of a property using ETH.  
- withdrawPropertyProceeds(): Allow creators to withdraw earnings.  
- getUserOwnedProperties(): View owned shares by wallet address.

## 4. Fallback AI Logic

If the Gemini API fails to respond, the AI assistant switches to fallback mode with predefined responses covering:  
- Greetings  
- Help requests  
- Platform workflow  
- Pricing and investment details  
- Legal and blockchain questions

## 5. Frontend & UI

- Marketplace: Displays all tokenized properties with filters.  
- Property Detail: Shows property details and purchase form.  
- Portfolio: Displays user’s owned shares and statistics.  
- Statistics: Interactive charts for price trends and ownership data.  
- AI Assistant: Provides conversational assistance and guidance.

## 6. Security Measures

- Smart contract validation for transactions, ownership, and proceeds.  
- Input validation on frontend.  
- Prevent double withdrawals.  
- Full transparency of transactions via Ethereum blockchain.

## 7. Deployment Configuration

- Network: Sepolia Testnet  
- Hosting: Replit / Vercel (production)  
- API Keys: Gemini API, Infura / Alchemy for blockchain connection.  
- Environment Variables: DEPLOYER\_PRIVATE\_KEY, GEMINI\_API\_KEY

## 8. Technologies

- Solidity (Smart Contracts)  
- React + TypeScript (Frontend)  
- Wagmi / Viem / RainbowKit (Web3 Integration)  
- Gemini API (AI Assistant)  
- IPFS (Metadata Storage)  
- Chart.js / Recharts (Data Visualization)

## 9. Deliverables

1. Fully functional ERC-1155 smart contract  
2. Deployed frontend Web3 application  
3. AI Assistant with fallback system  
4. Project documentation and README.md