Table of Contents

# Bound Market – Project Guide

**Bet Range Bound Volatility on Crypto Markets**

## What is Bound Market?

Bound Market is a decentralized platform that lets users bet on the volatility of Bitcoin (BTC) prices. Instead of traditional options or complex derivatives, users simply predict whether BTC will stay within a certain price range (“Stay In”) or break out of it (“Breakout”) over a 24-hour period. The platform is built on the Solana blockchain for speed and low fees, and uses real-time price data from the Pyth Network.

## Why Bound Market?

* **Simplicity:** Makes professional trading strategies (like volatility trading) accessible to everyone.
* **Transparency:** All trades and settlements are on-chain, so everything is verifiable.
* **No Deep Crypto Knowledge Needed:** The workflow is intuitive, and you don’t need to understand blockchain internals to use or explain the platform.

## How Does It Work? (User Flow)

1. **Connect Wallet:** User connects their Solana wallet (e.g., Phantom) to the platform.
2. **Place Order:** User chooses a position type:
   * **STAY\_IN:** Predict BTC will stay within a chosen price range for 24 hours.
   * **BREAKOUT:** Predict BTC will break out of the range at any time within 24 hours.
3. **Order Matching:** The backend matches the user’s order with a counterparty.
4. **Position Creation:** The smart contract creates a position on-chain for both parties.
5. **Monitoring:** The backend monitors BTC price in real time using Pyth Network.
6. **Settlement:** If the conditions are met (price stays in or breaks out), the contract settles the position automatically.
7. **Claim Payout:** Users can claim their winnings directly from the platform.

## Platform Architecture

The platform consists of three main components:

### 1. Smart Contract (/contract)

* **Role:** Handles all on-chain logic (deposits, withdrawals, position creation, settlement, payouts).
* **Tech:** Solana, Anchor framework.
* **Key Features:**
  + Manages user vaults (where funds are stored).
  + Creates and settles positions based on real-time price data.
  + Ensures only the rightful owner can claim payouts.
  + Uses Pyth Network for secure, reliable BTC price feeds.

### 2. Backend Service (/backend)

* **Role:** Acts as the “brain” of the platform, handling order matching, monitoring, and database sync.
* **Tech:** Node.js, Express, Supabase, Docker.
* **Key Features:**
  + Matches orders between users (STAY\_IN vs BREAKOUT).
  + Monitors open positions for settlement conditions.
  + Syncs on-chain data with the database for frontend display.
  + Provides health checks and service status endpoints.

### 3. Frontend Application (/frontend)

* **Role:** The user interface for trading, viewing positions, and managing accounts.
* **Tech:** Next.js, React, TailwindCSS, Chart.js, Supabase.
* **Key Features:**
  + Real-time BTC price charts and bound visualization.
  + Order book and position management UI.
  + Wallet integration for seamless Solana transactions.
  + Responsive design for desktop and mobile.

## Visual Overview

* **Architecture Diagram:** See docs/architecture.jpeg for a high-level system diagram.
* **Demo Video:** [Bound Market Technical Architecture](https://youtu.be/m5CbGHfHXys)

## Key Concepts

* **STAY\_IN Position:** Bet that BTC will stay within a price range for 24 hours. The longer it stays in, the higher the payout (up to 2x).
* **BREAKOUT Position:** Bet that BTC will break out of the range. The sooner it breaks out, the higher the payout (up to 2x).
* **Time-Weighted Payouts:** Payouts are calculated based on how long the position conditions are met.
* **Pyth Network:** Provides real-time, tamper-proof BTC price data to the smart contract.

## Integration & Data Flow

* **Frontend ↔ Backend:** Users interact with the frontend, which talks to the backend for order placement, status, and data.
* **Backend ↔ Smart Contract:** Backend matches orders, creates positions on-chain, and monitors for settlement.
* **All Components ↔ Supabase:** Used for persistent storage of orders, positions, and user data.
* **Smart Contract ↔ Pyth Network:** Fetches real-time BTC prices for settlement logic.

## Security

* **Funds Security:** All user funds are held in secure, program-derived accounts (PDAs) on Solana.
* **Data Integrity:** Price data comes from Pyth, a leading oracle network.
* **Access Control:** Only the original creator of a position can claim its payout.
* **On-Chain Clock:** All time-based logic uses Solana’s on-chain clock for fairness.

## Development & Deployment

### Prerequisites

* **Smart Contract:** Rust, Solana CLI, Anchor.
* **Backend:** Node.js 18+, Docker, Supabase account.
* **Frontend:** Node.js 20+, Yarn or npm, Solana wallet.

### Setup

* Each component has its own README with detailed setup instructions:
  + [Contract Setup](../contract/README.md#development-setup)
  + [Backend Setup](../backend/README.md#setup)
  + [Frontend Setup](../frontend/README.md#setup)

### Deployment

* **Smart Contract:** Deployed to Solana DevNet.
* **Backend:** Containerized, deployable to any cloud provider.
* **Frontend:** Optimized for Vercel, but can run anywhere.

## Explaining to Evaluators

* **Bound Market** is a simple, transparent way to bet on BTC volatility.
* **No prior blockchain knowledge is needed** to use or explain the platform.
* **Workflow:** User predicts → order matched → position created → price monitored → payout settled.
* **Security and transparency** are built-in via Solana and Pyth.
* **All code and logic** are open and verifiable.

## Where to Learn More

* **Architecture Diagram:** docs/architecture.jpeg
* **Demo Video:** [YouTube Link](https://youtu.be/m5CbGHfHXys)
* **Component Docs:** See individual READMEs in /contract, /backend, /frontend.

This guide should help your teammates quickly understand and confidently explain Bound Market to evaluators, even if they’re new to blockchain or the project!