

tbex

Terminal Blockchain Explorer

User Manual

Author: Joss Duff jod323@lehigh.edu

Course: CSE 411 — Advanced Programming Techniques

Institution: Lehigh University

Semester: Fall 2025

Version: 0.1.0

Date: December 17, 2025

Contents

1	Introduction	2
1.1	Key Features	2
2	System Requirements	2
3	Installation	2
4	Usage	3
4.1	Search	3
4.2	Keyboard Controls	3
4.3	Views	3
5	Architecture	3
5.1	Technology Stack	3
5.2	Module Structure	4
5.3	State Machine	4
5.4	Data Flow	4
5.5	RPC Client	4
5.6	Testing	4
6	References	5
6.1	Libraries	5
6.2	Technical References	5
6.3	Tools	5

1 Introduction

tbex (Terminal Blockchain Explorer) is a keyboard-driven Ethereum blockchain explorer for the terminal. It provides rapid access to blocks, transactions, and addresses through a responsive TUI, connecting via JSON-RPC to Ethereum nodes.

1.1 Key Features

- Keyboard-driven navigation with vim-style controls
- ENS name resolution (forward and reverse lookups)
- Automatic decoding of function selectors and event signatures
- ERC-20 token transfer and balance display
- Block builder detection (Flashbots, rsync, Beaver, etc.)
- Persistent search history

2 System Requirements

Requirement	Details
Operating System	Linux, macOS, or Windows (WSL recommended)
Terminal	UTF-8 and color support
Rust Toolchain	Rust 1.70+ (for building)
RPC Endpoint	Ethereum JSON-RPC endpoint

RPC options include public endpoints (Cloudflare, Ankr), commercial providers (Alchemy, Infura), or a local node (Geth, Reth, Besu).

3 Installation

```
# Install Rust if needed
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh

# Build and install
git clone https://github.com/yourusername/tbex.git
cd tbex
cargo install --path .

# Run
tbex
```

On first run, tbex prompts for an RPC endpoint. Configuration is saved to `~/.config/tbex/config.toml`.

4 Usage

4.1 Search

Query Type	Format	Example
Block Number	Decimal or hex	19000000, 0x121eac0
Transaction Hash	0x + 64 hex chars	0x5c504ed4...
Address	0x + 40 hex chars	0xd8dA6BF2...
ENS Name	.eth domain	vitalik.eth

4.2 Keyboard Controls

Key	Action
↑/↓ or j/k	Navigate
Enter	Select/follow link
Tab	Toggle view mode
b	Go back
h	Go home
Esc/q	Quit

4.3 Views

Block View: Block metadata, miner/builder info, gas statistics, transaction list. Press Tab to toggle between info and transaction list.

Transaction View: Status, from/to addresses with ENS, value, decoded method, gas details, token transfers, decoded event logs. Addresses are navigable links.

Address View: For EOAs: balance, nonce, token balances. For contracts: code size, proxy implementation, token info (name, symbol, decimals).

5 Architecture

5.1 Technology Stack

Component	Technology	Purpose
Language	Rust 2021	Performance, safety
TUI	ratatui 0.29	Terminal rendering
Async	Tokio	Asynchronous I/O
Blockchain	Alloy 1.x	Ethereum RPC client
Config	serde + toml	Serialization

5.2 Module Structure

Module	Purpose
app.rs	Application state machine
rpc/	Blockchain RPC client with retry logic
search.rs	Query parsing
config.rs	Configuration management
ui/	Screen rendering (block, tx, address pages)

5.3 State Machine

The application uses a state machine with screens: Home, Loading, BlockResult, TxResult, AddressResult, and Error. Navigation history enables back traversal.

```

1 pub enum Screen {
2     Home,
3     Loading(String),
4     BlockResult(BlockResult),
5     TxResult(TxResult),
6     AddressResult(AddressResult),
7     Error(String),
8 }
```

5.4 Data Flow

1. User enters query
2. Query parsed by search.rs
3. State transitions to Loading
4. RPC client fetches data asynchronously
5. State transitions to result screen
6. UI renders; user can navigate links

5.5 RPC Client

Features exponential backoff retry for rate limits, batch ENS resolution via the ReverseRecords contract, and automatic decoding of common function selectors and event signatures.

5.6 Testing

99 total tests: 69 unit tests (formatting, parsing, state machine) and 30 integration tests (UI rendering via ratatui's test backend).

```

cargo test          # All tests
cargo test --test ui    # UI tests only
```

6 References

6.1 Libraries

- **Ratatui** — <https://ratatui.rs/> (MIT)
- **Alloy** — <https://github.com/alloy-rs/alloy> (Apache-2.0/MIT)
- **Tokio** — <https://tokio.rs/> (MIT)
- **tui-input** — <https://github.com/sayanarijit/tui-input> (MIT)

6.2 Technical References

- Ethereum JSON-RPC Specification — <https://ethereum.org/en/developers/docs/apis/json-rpc/>
- EIP-1559, EIP-4844 — <https://eips.ethereum.org/>
- ENS Documentation — <https://docs.ens.domains/>

6.3 Tools

- **Claude Opus 4.5** (Anthropic) — AI assistant used for development and documentation
- **Etherscan** — Reference blockchain explorer