

# LendBit: Localised Lending Diamonds

---

LendBit brings collateralised crypto lending to local markets. The protocol is built on the [EIP-2535 Diamond](#) standard, supports multi-asset vaults, Chainlink-powered price feeds, tenured loans, local-currency abstractions, and now an Aave-integrated yield layer that keeps collateral productive.

This repository hosts the on-chain contracts, deployment scripts, and Foundry test suite.

## Core Features

- **Diamond Architecture** – Modular facets ([Protocol](#), [VaultManager](#), [Liquidation](#), [PriceOracle](#), [PositionManager](#), [YieldStrategy](#), etc.) expose isolated functionality while sharing storage through [LibAppStorage](#).
- **Collateral & Vaults** – Security council can list collateral tokens, configure LTVs, and deploy ERC4626-style vaults that track deposits/borrows plus reserve factors.
- **Tenured Loans** – Borrowers lock principal for a specified tenure, accrue interest + penalties, and can repay early or be liquidated when health factors fall below thresholds.
- **Local Currency Support** – Map currencies such as NGN, KES, UGX to stable on-chain representations and price feeds to unlock regional borrowing experiences.
- **Chainlink Functions** – Integrations for price discovery and off-chain computation relayed through router + DON configuration held in storage.
- **Yield Strategy Facet** – A configurable slice of collateral is deployed into an Aave-compatible pool to earn interest, automatically splitting yield between the borrower and the protocol treasury. See [docs/yield-strategy.md](#) for the full design.

## Repository Layout (abridged)

```
contracts/
  Diamond.sol           # Diamond proxy entry point
  TokenVault.sol        # ERC4626-style vault for lender
liquidity
  facets/               # Individual facet contracts
  libraries/            # Lib* files with core protocol logic
  models/               # Shared structs, errors, constants
  mocks/                # Test doubles (e.g., MockAavePool)
scripts/
  Deploy.s.sol          # Foundry deployment script
  deploy.js             # Hardhat deployment script
test/
  *.t.sol               # Foundry test suites (Protocol,
VaultManager, YieldStrategy...)
docs/
  yield-strategy.md     # Detailed description of the Aave
integration
```

---

## Prerequisites

- Node.js >= 18 (used for selector generation via `scripts/genSelectors.js`)
- Yarn or npm
- Foundry toolchain (`forge`, `cast`, `anvil`)
- (Optional) Hardhat for JS deployments

## Installation

```
git clone https://github.com/LendBit-p2p/lendbit-localised.git
cd lendbit-localised
# install JS deps for selector scripts
npm install --legacy-peer-deps
# pull Foundry dependencies
forge install
```

## Useful Commands

### Compile

```
forge build
# or
npx hardhat compile
```

### Test

```
forge test # run entire suite (140+ tests)
forge test --match-path test/YieldStrategy.t.sol
```

### Lint & Format

```
forge fmt
npx solhint "contracts/**/*.sol"
```

### Deploy (examples)

```
# Hardhat (network config in hardhat.config.js)
npx hardhat run scripts/deploy.js --network <network>

# Foundry script
```

```
defaultAnvilRpc=http://localhost:8545
forge script scripts/Deploy.s.sol \
  --fork-url $defaultAnvilRpc \
  --broadcast
```

## Yield Strategy Overview

The `YieldStrategyFacet` routes a configurable portion of each position's collateral into an Aave-style pool:

- Configure via `configureYieldToken(token, pool, aToken, allocationBps, protocolShareBps)`.
- Deposits automatically allocate, withdrawals/liquidations automatically unwind.
- Borrowers claim rewards with `claimYield`, protocol treasury harvests via `harvestProtocolYield`.
- Documentation lives in `docs/yield-strategy.md`; tests in `test/YieldStrategy.t.sol`.