**BlackSilk Blockchain Project Proposal**

**Blockchain Architecture**

**BlackSilk will be a proof-of-work chain using Monero’s RandomX RX/0 algorithm optimized for CPU mining. Blocks will target ~2–3 minutes each (90–145 seconds), with difficulty retargeting automatically to stay in this window. The genesis block will award 86 BLK, and the reward will halve by 50% every 125,000 blocks (to throttle inflation). This design mirrors Bitcoin’s cap (21 million coins) and halving mechanism, but on a shorter cycle. After ~21 million BLK are mined, a permanent tail emission of 0.5 BLK per block will remain, similar to Monero’s “tail emission” to keep miners incentivized. The 21M supply cap (and halving schedule) ensures long-term scarcity, with fixed emissions thereafter to sustain network security.**

**Privacy & Cryptography Features**

**BlackSilk’s transactions will use ring signatures and stealth addresses (à la Monero) so that outputs are untraceable. Each spend will be signed over a “ring” of decoy public keys, making it *computationally infeasible* to tell which participant actually signed. Likewise, every wallet has one public address but each payment goes to a random one-time stealth address, so incoming funds cannot be linked on-chain. Transaction amounts and asset types will be confidentially encrypted (Confidential Transactions): outsiders only see commitments, not values. For example, on Liquid, CTs hide amounts so that only sender/recipient know the true balance and preserve on-chain privacy. We will use Bulletproofs (or Bulletproofs+ range proofs) to prove transaction sums and ranges in zero-knowledge without a trusted setup. More advanced ZK proofs (e.g. zk-SNARKs) could be added later: as Zcash demonstrates, zero-knowledge proofs can verify that balances and spendings are correct without ever revealing the amounts or the history. Wallet software will by default encrypt all private keys on disk under a strong passphrase, preventing theft of secrets if a device is compromised. Finally, all node‐to‐node and wallet‐node communication will use TLS with Perfect Forward Secrecy (PFS) – for example, Dero’s network already uses self-signed TLS so that peers’ traffic is encrypted end-to-end. In short, every transaction on BlackSilk is privacy-preserving by design: inputs, outputs, and amounts are hidden from everyone but the participants.**

**Network Layer & Anonymity**

**All network traffic is routed anonymously. BlackSilk nodes and wallets will be configurable as Tor (.onion) and I2P hidden services, so that every P2P and RPC connection is end-to-end encrypted and the real IPs are never exposed. Monero documentation notes that Tor/I2P hidden services “are end-to-end encrypted and private by default” when used for wallet RPC and P2P. Tor hidden services in particular “facilitate anonymous, decentralized communication, obfuscating the client’s and server’s real IP addresses”. In practice, a BlackSilk node will host separate Tor ports (e.g. 1776 or 18084) for P2P relay and wallet RPC, with TLS on top. This way *no node or exit relay ever sees a real IP*: as illustrated below, a client’s traffic is wrapped in successive layers of encryption at the Entry, Middle, and Exit Tor relays, and the final Exit only sees an IP it was forwarded from【52†】. Similarly, I2P tunnels ensure “no one can see where traffic is coming from, where it is going, or what the contents are”.**

**With nodes exposed as hidden services and all links inside Tor/I2P, BlackSilk’s P2P network is fully opaque. We will additionally encrypt all P2P messages (like Dero does) with TLS+PFS to defend against eavesdropping and tampering. In this layered design, an attacker would have to break multiple cryptographic layers to learn anything: Tor/I2P hide IPs, and TLS guards against MITM on each hop. In effect, BlackSilk traffic is as anonymous as Tor itself, giving users complete decoupling from any real-world identity or network trace.**

**By default, all connections (transactions, block sync, wallet RPC) go through these anonymity networks. Users do not connect over clearnet IP addresses unless explicitly opting out. With PFS-enabled TLS, even if a node’s long-term keys are compromised later, past communications remain secure. Thus the design ensures *end-to-end* encrypted routing and service discovery – for example, Tor’s decentralized onion address resolution – preventing anyone from linking BlackSilk activity to a person’s location or ISP. In sum, the network layer is built to eliminate metadata leakage.**

**Marketplace Design**

**BlackSilk’s marketplace frontend will be a static HTML/JS web app (optionally built with a privacy-centric SSG like Hugo or Next.js in static-export mode) served by secure backend servers. The UI will mimic the classic Silk Road layout (dark theme, clear categories and search), but with modern responsive design. Every listing, order, and message will be recorded either on-chain or in encrypted backend databases, ensuring consistency with the blockchain. Payments flow directly on BlackSilk coins: a buyer sends funds into a smart escrow contract (or multisig address) rather than to the seller directly. For example, a 2-of-3 multisig (buyer/seller/arbiter) can be created so that funds are released to the seller only when both buyer and moderator sign off, or refunded if a dispute is resolved in the buyer’s favor. This trustless escrow logic, inspired by blockchain escrow patterns, allows fast settlement yet safeguards both parties. Users will build reputation scores stored on-chain or in the backend, and disputes will be handled via an arbitration process (possibly decentralized with voting or an on-chain DAO).**

**Sellers will be free to list any products or services; we expect moderate content rules only to exclude real-world threats (e.g. violent intimidation). The system will auto-balance load across multiple backend instances or use container orchestration for scalability. Static assets and images (e.g. listing photos) may be stored on IPFS (to distribute bandwidth and avoid censorship). In effect, the marketplace backend will be decentralized and stateless: many servers (potentially run by volunteers) can serve content and API endpoints behind a load balancer or service mesh. All API calls (for searching or posting listings) will be over Tor/I2P and authenticated by blockchain wallets, so the web frontend never touches user keys or secrets. The overall architecture resembles a microservices setup or distributed Dapp, but with full integration to BlackSilk’s blockchain for payments and escrow.**

**Development Stack Suggestions**

* **Core node (blockchain engine): *Rust* or *C++*. Both are proven for cryptocurrency clients (C++ runs Bitcoin/Monero; Rust is used by Polkadot/Substrate). Rust offers memory safety and good concurrency, while C++ has mature crypto libraries. Either language should be used with strict code reviews and fuzz-testing.**
* **CLI wallet & miner: *Rust* or *Go*. Rust shares code with the core or can use Serde for serialization; Go (golang) compiles to single binaries easily and has good cryptography libraries. Cross-compilation builds should cover Windows and Linux.**
* **Frontend web UI: Static HTML/CSS/JS. Use a privacy-preserving framework (e.g. Hugo, Eleventy, or React/Next.js in static export mode). The site must avoid third-party trackers; do not include Google Analytics or ads. Inline or minimal JS can handle AJAX calls to backend APIs.**
* **Backend APIs: *Rust* (e.g. Actix-web, Rocket) or *Python* (e.g. FastAPI, Flask with gunicorn). These services run over Tor/I2P and provide JSON endpoints for listings, user profiles, etc. FastAPI is nice for auto-generating OpenAPI docs (Swagger) to help third parties. Actix or Rocket give high performance with Rust’s safety.**
* **Database: Use an encrypted PostgreSQL instance or LMDB. The database will store user accounts, ratings, and off-chain metadata. Its storage should sit on a Linux LUKS-encrypted volume (PostgreSQL supports filesystem or tablespace encryption). Alternatively, a key-value store like LMDB or RocksDB with filesystem encryption could be used for lighter-weight use.**

**Security Enhancements**

* **Anti-tracking & fingerprint resistance: The web UI will disable cookies, block known trackers, and not include any analytics by default. All external calls (fonts, maps, libraries) are either bundled or anonymized. Browser security headers (CSP, HSTS, NoReferrer) will be strict. We may offer a Tor Browser plugin mode. In short, the marketplace pages should behave like a privacy tool: avoid revealing any data about the user’s browser or usage. Indeed, web trackers often gather IP, User-Agent, fonts and cookies to build a profile – we actively prevent that.**
* **Smart escrow and contracts: The escrow logic will be encoded in a simple, audited contract. We avoid full Turing-complete contracts to reduce bugs, using fixed multisig or time-locked-release patterns. For example, a smart contract can hold buyer funds and automatically release them if both parties confirm delivery (or after a timeout). Third-party escrow or dispute resolution could be facilitated by an on-chain DAO or by multi-signature arbitration. Importantly, all escrow transactions use stealth addresses or one-time addresses (as per Monero’s stealth pattern) so that metadata like “this user used escrow X” is hidden.**
* **API & extensibility: All backend APIs will be authenticated (via wallet signatures) and documented. We will publish a full OpenAPI (Swagger) specification so independent developers can build plugins or mobile apps. Code signing keys or secure boot practices should be used so that wallets and nodes can verify official releases.**
* **Hardware wallet support: The system will support signing transactions with hardware wallets (e.g. Ledger, Trezor) for key security. Monero already has hardware wallet support, demonstrating that Bulletproofs and ring-CT can be handled by devices. We will implement a BlackSilk app for hardware wallets and ensure all signing paths are compatible.**
* **Metadata protection: Servers and nodes will minimize logs. Wherever possible, logs are either turned off or encrypted at rest and rotated. The node software should run as a “Torified” service (like Tails OS *amnesiac* mode): on shutdown it should wipe any cache or history to leave zero traces. We also avoid placing sensitive info (like user IDs) in URLs or database indices.**

**Optional Enhancements**

* **IPFS content delivery: Use IPFS for hosting static content (listing images, product manuals, e-books, etc.). IPFS is a peer-to-peer file system where content is addressed by hash, so once uploaded, files can be retrieved from any node in the network. This makes delivery censorship-resistant: even if our servers go down, any peer can serve the files by hash. For example, you could let a seller upload a file to IPFS and store the CID on-chain, so buyers fetch it from the global IPFS network.**
* **Onion-only access: In addition to a clearnet domain (if used at all), the marketplace will run exclusively as a Tor hidden service (and optionally an I2P eepsite). This ensures that even name resolution is decentralized: Tor’s network will resolve the .onion address via distributed directories.**
* **“Zero-trace” operation: All marketplace code will avoid persistent logs and use in-memory data where possible. Inspired by Tails (the amnesic live OS), our goal is that a full system reboot (or container restart) loses all memory of prior transactions. Any required logging (for debugging) should be strictly ephemeral or opt-in.**

**Naming Conventions & Inspiration**

**Consistent with our freedom-centric ethos, we’ll pepper the codebase and protocol with symbolic names: for example, default P2P port 1776 (the year of American independence) and block size limit 1984 KB (echoing Orwell’s *1984* as a warning against surveillance). Cryptocurrency tickers like “BLK” (BlackSilk Coin) and function names can reflect civil rights history (e.g. liberty\_tx() or emancipation\_reward). The genesis block ID or network magic number might be chosen to commemorate a notable date in anti-censorship history (e.g. 0x1D670 for July 26, 1953 – Cuban 26th of July Movement). Such references (documented in the whitepaper) reinforce the project’s values of privacy, free speech, and resistance to tyranny.**

**Documentation Deliverables**

**We will produce:**

* **Technical Whitepaper: A formal description of the protocol, consensus rules, coin economics, and privacy mechanisms (serving as the definitive reference).**
* **Architecture Diagram & Breakdown: Visual and textual explanation of system components (nodes, wallets, frontend, backend, anonymity networks, etc.) and how they interact.**
* **Codebase Design Philosophy: Commentary on why we chose Rust/C++, the abstractions for privacy (e.g. using Tornado, Ricochet libraries, etc.), and coding best practices (fuzzing, audits).**
* **Node & Wallet Build Instructions: Step-by-step guides to compile and run the full node and CLI wallet on Windows and Linux, including prerequisite software (Rust/C++ toolchain, libraries), configuration for Tor/I2P, and security hardening tips. Example commands (e.g. cargo build --release, or CMake instructions) will be included.**
* **API Documentation for Marketplace: A machine-readable OpenAPI/Swagger file plus human guides for each endpoint (how to list an item, how to query listings, how to place an order, etc.). This will enable third-party apps (mobile wallets, bots, browser extensions) to integrate.**

**By combining proven privacy protocols (ring signatures, ZK proofs) with modern anonymity networks (Tor/I2P) and a robust, open implementation stack, BlackSilk aims to offer a highly secure, censorship-resistant decentralized marketplace. The complete documentation will lay out every aspect – from block-level parameters to user-facing interfaces – so the project is auditable, extendable, and aligned with its ethos of freedom and privacy.**

**Sources: Monero’s Moneropedia and blogs for ring signatures, stealth addresses, RandomX, Bulletproofs; Blockstream and Zcash docs on Confidential Transactions and ZKPs; Monero docs on Tor/I2P integration; Dero’s encrypted networking; I2P official site; Silk Road overviews; blockchain escrow pattern; privacy anti-tracking studies; PostgreSQL encryption docs; IPFS official description; and various community/historical references as noted.**