# Implement a URL Shortener with Expiry and Analytics

**Objective**: Build a Python-based URL shortener system that shortens URLs, tracks usage analytics, and allows for link expiration.

---

## Requirements:

1. **Core Functionality**:
    - Create a shortened URL for any given long URL.
    - Each shortened URL should be unique and have a base URL (e.g., `https://short.ly/abc123`).
2. **Expiry**:
    - Allow users to specify an expiration time for the shortened URL (in hours).
    - If no expiry is set, default to 24 hours.
    - Ensure expired URLs no longer redirect to the original URL.
3. **Analytics**:
    - Track the number of times each shortened URL has been accessed.
    - Log the timestamp and IP address of each access.
4. **Storage**:
    - Use a simple database (SQLite) to store:
        - Original URL.
        - Shortened URL.
        - Creation timestamp.
        - Expiration timestamp.
        - Access logs (shortened URL, timestamp, IP address).
5. **CLI or API**:
    - Provide a command-line interface (CLI) or a simple REST API with endpoints for:
        - `POST /shorten`: Create a shortened URL.
        - `GET /<short_url>`: Redirect to the original URL if not expired.
        - `GET /analytics/<short_url>`: Retrieve analytics data for a specific shortened URL.
6. **Constraints**:
    - Ensure the same long URL always generates the same shortened URL (idempotent).
    - Validate input URLs to ensure they are well-formed.
    - Use modular code to allow easy expansion of features in the future.

---

## Bonus:

- Use `hashlib` to create a hash-based short URL identifier.
- Add optional password protection for accessing certain shortened URLs.

---

## Deliverables:

- Python code implementing the system.
- A SQLite database with example data.
- A README with clear instructions to run the application.