1. Definition & Core Concept

A watchlist is a curated list used to monitor, track, or manage specific items (e.g., entities, assets, or data) for purposes like security, investment, or personal organization. Its functionality varies by context:

Security & Compliance: Lists of high-risk entities (e.g., terminated employees, suspicious IPs) for threat detection .

Finance: Personalized stock/investment trackers for market analysis .

Personal Use: Web/mobile apps to manage movies, books, or tasks .

2. Key Applications

A. Cybersecurity (Microsoft Sentinel)

Threat Correlation: Import IPs, file hashes, or terminated employee lists via CSV to enrich security logs and automate alerts .

Alert Fatigue Reduction: Create allowlists to suppress benign alerts (e.g., authorized IP ranges) .

Limitations: Max 10M active items per workspace; not for large datasets .

B. Financial Tracking

Google Finance (Historical): Tracked stocks/ETFs with real-time data, though discontinued in 2020 .

Features: Price alerts, performance benchmarking, and news integration .

C. Personal Productivity

Flask/Spring Boot Apps: Manage movie watchlists with ratings, priorities, and comments .

File Monitoring: Tools like watchlist (Node.js) track directory changes and trigger commands .

D. Physical Security (Genetec ClearID)

Visitor Screening: Block/notify based on individual/company watchlists (e.g., VIPs or banned entities) .

3. Technical Implementation

Data Formats: CSV/TSV imports for bulk data (e.g., IP lists) .

APIs & Integration:

Microsoft Sentinel uses REST APIs for watchlist management .

Movie apps fetch ratings via external APIs (e.g., IMDb) .

Search Optimization: Designated `itemsSearchKey` (e.g., IP column) speeds up queries .

4. Common Challenges & Solutions

Installation: Node.js tools require `npm install` and directory setup .

Filtering: UI filters (e.g., Wikipedia’s iOS app) allow customization (e.g., "Latest Revision" or "Bot edits") .

Ignoring Files: Use `--ignore` flags in CLI tools to exclude directories

5. Best Practices

Security: Regularly update watchlists (e.g., every 12 days in Sentinel) .

Usability: Modular design and clear documentation (e.g., Flask tutorials for beginners) .

One: Database

📘 **users**

- id: Primary Key
- email: String (unique)
- password_hash: String

📘 **watchlists**

- id: Primary Key
- user_id: FK → users(id)
- symbol: e.g., "EURUSD"

Constraint: (user_id, symbol) must be unique (no duplicates)

📘 **price_alerts**

- id: Primary Key
- user_id: FK
- symbol: e.g., "EURUSD"
- condition: gt / lt / eq
- target_price: Decimal
- is_triggered: Boolean

Two: API

1. GetWatchlist()

Core Method: GET

Example Return:

```
{
    "watchlist": ["EURUSD", "USDJPY"]
}
```

2. AddWatchlistPair()

Core Method: POST

Example Return:

```
{
    "symbol": "GBPUSD"
}
```

3. RemoveWatchlistPair()

Core Method: POST

Example Return:

```
{
    "message": "Pair USDJPY removed from watchlist."
}
```

4. GetWatchlistPrices()

Core Method: GET

Example Return:

```
{
    "data": [
        {
            "symbol": "EURUSD",
            "bid": 1.1012,
            "ask": 1.1014,
            "spread": 0.0002,
            "change_24h": -0.14
        },
        {
            "symbol": "USDJPY",
            "bid": 157.18,
            "ask": 157.23,
            "spread": 0.05,
            "change_24h": +0.07
        }
    ]
}
```

5. CreatePriceAlert()

Core Method: GET

Example input:

```
{
    "symbol": "EURUSD",
    "condition": "gt",      // or "lt", "eq"
    "target_price": 1.1100
}
```

6. DeleteAlert()

Core Method: POST