Anki cards Logic:

The algorithm must track and adjust the following for each card:

1. **Ease Factor (EF)**:
    - Indicates how "easy" the card is for the user (default: 2.5).
    - Updates after every review.
2. **Review Interval**:
    - The number of days (or minutes for very short intervals) until the card reappears.
3. **Repetition Count**:
    - Number of consecutive correct reviews for this card.
    - Starts at 0 and increases with correct answers.
4. **Lapses**:
    - Number of times the user selects "Again."

Key Parameters

Each button ("Again", "Hard", "Good", "Easy") affects the following parameters:

- **Again**:
    - Resets repetition count to 0.
    - Resets interval to the minimum (e.g., 10 minutes).
- **Hard**:
    - Applies a shorter interval (e.g., 1.2× current interval).
    - Slightly decreases the ease factor.
- **Good**:
    - Sets the next interval using the formula: Next Interval=Current Interval×Ease Factor\text{Next Interval} = \text{Current Interval} \times \text{Ease Factor}Next Interval=Current Interval×Ease Factor
    - Keeps EF stable.
- **Easy**:
    - Multiplies the interval by a larger factor (e.g., 1.3× EF).
    - Slightly increases the ease factor.

Algorithm Logic

Here's how you can implement the logic in Python:

Initialization

Each card starts with:

- `EF = 2.5` (default ease factor)
- `Interval = 1` (in days)
- `Repetition = 0`
- `Lapses = 0`

API Endpoint

Create a REST API for the Anki microservice:

1. **Endpoint**: `POST /update-card`

Input:
json
Copy code

```
{
    "card_id": "12345",
    "button": "Good",
    "current_interval": 2,
    "ef": 2.5,
    "repetitions": 3,
    "lapses": 0
```

2. `}`