

Hyperfusion Pricing Tool - API Integration Contract

Generated from implementation in backend/app/main.py

Deployment context

- Internal FastAPI route decorators: /health, /skus, /uplifts, /use-cases, /quote
- External deployed paths on Vercel: /api/health, /api/skus, /api/uplifts, /api/use-cases, /api/quote

1) Endpoint Contract Table

GET /health

- Auth required: No
- Defined in: backend/app/main.py -> health()
- Response fields:
 - ok: bool
 - sku_count: int
 - uplift_count: int
 - use_case_count: int
- Explicit HTTPException raised in function: none

GET /skus

- Auth required: Yes (Depends(require_api_key))
- Defined in: backend/app/main.py -> get_skus()
- Header required: x-api-key: string
- Response: array of objects:
 - sku_code: str
 - name: str
 - unit_label: str
 - base_unit_price: float
 - unit: float
- Explicit error codes raised in code path:
 - 401 Unauthorized
 - 500 Server API key is not configured

GET /uplifts

- Auth required: Yes
- Defined in: backend/app/main.py -> get_uplifts()
- Header required: x-api-key: string
- Response: array of objects:
 - uplift_name: str
 - percent_decimal: float
 - enabled: bool
- Explicit error codes raised in code path:
 - 401
 - 500

GET /use-cases

- Auth required: Yes
- Defined in: backend/app/main.py -> get_use_cases()
- Header required: x-api-key: string
- Response: array of strings
- Explicit error codes raised in code path:
 - 401
 - 500

POST /quote

- Auth required: Yes
- Defined in: backend/app/main.py -> quote(payload: QuoteRequest)
- Header required: x-api-key: string
- Request body model: QuoteRequest (see section 2)
- Response (mode="sku"):

```

sku: { sku_code: str, name: str, unit_label: str }
quantity_raw: float
unit_multiplier: float
relative_units: float
base_unit_price: float
base_cost: float
discount_decimal: float
discounted_cost: float
uplift_decimal: float
final_cost: float
applied_uplifts: array of { uplift_name: str, percent_decimal: float }
- Response (mode="use_case"):
mode: "use_case"
use_case: str | None
hours: float | None
grand_total_usd: float
breakdown: array of {
    sku_code: str,
    sku_name: str,
    unit_label: str,
    units_per_hour: float,
    units_total: float,
    cost_usd: float
}
- Explicit error codes raised in code path:
401 (auth)
500 (missing server key)
400 (invalid mode/fields/unknown uplifts/unknown use_case)
404 (unknown sku_code)

```

2) QuoteRequest Model

Source: backend/app/main.py class QuoteRequest

```

- mode: str, default "sku", no Field validation
- sku_code: str | None, default None
- quantity: float | None, default None, validation gt=0
- use_case: str | None, default None
- hours: float | None, default None, validation gt=0
- customer: str | None, default None
- gpu_type: str | None, default None
- region: str | None, default None
- rank_results_by: str | None, default None
- uplift_names: list[str] | None, default None

```

3) Business Logic

Volume discount logic

```

- Implemented in backend/app/main.py _calculate_sku_quote()
- discount_decimal starts at 0.0
- For each volume discount:
    if min_units <= relative_units and discount_decimal is greater than current, replace current.
- discounted_cost = base_cost * (1 - discount_decimal)

```

Uplift logic

```

- Implemented in _resolve_uplifts() and _calculate_sku_quote()
- If uplift_names is None, use all enabled uplifts.
- Else validate provided names; unknown names raise 400.
- uplift_decimal = sum(selected uplifts percent_decimal)
- final_cost = discounted_cost * (1 + uplift_decimal)

```

Use case mapping logic

- Parsing in `_parse_use_case_mappings()`
- `use_cases` = headers excluding "SKU Code"
- Values parsed with `_parse_float`
- `use_case` quote path in `quote()`:
 - for each mapped SKU:
 - `units_per_hour` = `mapping[use_case]` default 0
 - `units_total` = `units_per_hour * hours`
 - skip when `units_total <= 0`
 - skip when sku missing in `store.skus`
 - price using `_calculate_sku_quote(sku_code, units_total, uplift_names)`
 - aggregate `grand_total_usd` and `breakdown`

4) Verified Example Requests

GET /health

```
curl -X GET "https://<host>/api/health"
```

GET /skus

```
curl -X GET "https://<host>/api/skus" -H "x-api-key: <PRICING_API_KEY>"
```

GET /uplifts

```
curl -X GET "https://<host>/api/uplifts" -H "x-api-key: <PRICING_API_KEY>"
```

GET /use-cases

```
curl -X GET "https://<host>/api/use-cases" -H "x-api-key: <PRICING_API_KEY>"
```

POST /quote (sku)

```
curl -X POST "https://<host>/api/quote" \\  
  -H "Content-Type: application/json" \\  
  -H "x-api-key: <PRICING_API_KEY>" \\  
  -d '{"mode":"sku","sku_code":"example-sku","quantity":100,"uplift_names":["example-uplift"]}'
```

POST /quote (use_case)

```
curl -X POST "https://<host>/api/quote" \\  
  -H "Content-Type: application/json" \\  
  -H "x-api-key: <PRICING_API_KEY>" \\  
  -d '{"mode":"use_case","use_case":"example-use-case","hours":100,"uplift_names":["example-uplift"]}'
```

5) Minimal Integration Card

- Public endpoint: /health
- Protected endpoints: /skus, /uplifts, /use-cases, /quote
- Required header on protected endpoints: x-api-key
- Server-side key env var: PRICING_API_KEY
- Frontend build-time key env var: VITE_PRICING_API_KEY