

CFAPI Endpoint Guide

Base URL: `http://localhost:3000`

All dates are returned as `YYYY-MM-DD`. Requests accept `YYYY-MM-DD` (ISO timestamps also accepted).

System

GET `/health` Health check

```
{
  "ok": true
}
```

GET `/docs` OpenAPI JSON

Returns OpenAPI JSON schema

Auth

POST `/auth/register` Create a user account

```
# Request
{
  "email": "demo@demo.com",
  "password": "password123"
}

# Response
{
  "id": "user_id",
  "email": "demo@demo.com"
}
```

POST /auth/login Get a JWT token

```
# Request
{
  "email": "demo@demo.com",
  "password": "password123"
}

# Response
{
  "token": "jwt_token_here"
}
```

Users (Admin only)

GET /users List all users

```
# Response
[
  {
    "id": "user_id",
    "email": "demo@demo.com",
    "createdAt": "2024-01-10"
  }
]
```

GET /users?email=demo@demo.com Filter users by email

```
# Response
[
  {
    "id": "user_id",
    "email": "demo@demo.com",
    "createdAt": "2024-01-10"
  }
]
```

GET /users/:id Get a user by id

```
# Response
{
  "id": "user_id",
  "email": "demo@demo.com",
  "createdAt": "2024-01-10"
}
```

```
"id": "user_id",  
"email": "demo@demo.com",  
"createdAt": "2024-01-10"  
}
```

PATCH /users/:id Update email and/or password

```
# Request  
{  
  "email": "new@demo.com",  
  "password": "newpassword123"  
}  
  
# Response  
{  
  "id": "user_id",  
  "email": "new@demo.com",  
  "createdAt": "2024-01-10"  
}
```

DELETE /users/:id Delete a user

```
# Response  
204 No Content
```

Debts

GET /debts List debts

```
# Response  
[  
  {  
    "id": "debt_id",  
    "name": "Credit Card",  
    "principalDollars": 1200.5,  
    "aprBps": 1999,  
    "minPaymentDollars": 35.25,  
    "estimatedMonthlyPaymentDollars": 150.75,  
    "dueDayOfMonth": 12,  
    "estimatedPayoffDate": "2025-02-12",  
    "tags": ["credit-card"]  
  }  
]
```

```
}  
]
```

POST /debts Create a debt

```
# Request  
{  
  "name": "Credit Card",  
  "principalDollars": 1200.5,  
  "aprBps": 1999,  
  "minPaymentDollars": 35.25,  
  "estimatedMonthlyPaymentDollars": 150.75,  
  "dueDayOfMonth": 12,  
  "tags": ["credit-card", "high-apr"]  
}  
  
# Response  
{ ...debt }
```

PATCH /debts/:id Update a debt

```
# Request  
{  
  "minPaymentDollars": 50,  
  "tags": ["loan"]  
}  
  
# Response  
{ ...debt }
```

DELETE /debts/:id Delete a debt

```
# Response  
204 No Content
```

POST /debts/:id/payments Record a payment

```
# Request  
{  
  "amountDollars": 150.75,  
  "paymentDate": "2024-01-20"
```

```
}

# Response
{
  "debt": { ...debt },
  "payment": {
    "id": "payment_id",
    "debtId": "debt_id",
    "amountDollars": 150.75,
    "paymentDate": "2024-01-20"
  }
}
```

GET /debts/:id/payments List debt payments (optional startDate/endDate)

```
# Response
[
  {
    "id": "payment_id",
    "debtId": "debt_id",
    "amountDollars": 150.75,
    "paymentDate": "2024-01-20"
  }
]
```

Bills

GET /bills List bills

```
# Response
[
  {
    "id": "bill_id",
    "name": "Rent",
    "amountDollars": 1200.75,
    "dueDayOfMonth": 1,
    "frequency": "MONTHLY",
    "isEssential": true,
    "autopay": true
  }
]
```

POST /bills Create a bill

```
# Request
{
  "name": "Rent",
  "amountDollars": 1200.75,
  "dueDayOfMonth": 1,
  "frequency": "MONTHLY",
  "isEssential": true,
  "autopay": true
}

# Response
{ ...bill }
```

Subscriptions

GET /subscriptions List subscriptions

```
# Response
[
  {
    "id": "subscription_id",
    "name": "Spotify",
    "amountDollars": 10.99,
    "billingDayOfMonth": 18,
    "frequency": "MONTHLY",
    "cancelable": true
  }
]
```

POST /subscriptions Create a subscription

```
# Request
{
  "name": "Spotify",
  "amountDollars": 10.99,
  "billingDayOfMonth": 18,
  "frequency": "MONTHLY",
  "cancelable": true
}

# Response
{ ...subscription }
```

Income Streams

GET /income-streams List income streams

```
# Response
[
  {
    "id": "income_id",
    "name": "Paycheck",
    "amountDollars": 2500.5,
    "cadence": "BIWEEKLY",
    "nextPayDate": "2024-01-05",
    "amountChangeStatus": "NO_HISTORY",
    "tags": ["job"]
  }
]
```

POST /income-streams Create an income stream

```
# Request
{
  "name": "Paycheck",
  "amountDollars": 2500.5,
  "cadence": "BIWEEKLY",
  "nextPayDate": "2024-01-05",
  "tags": ["job", "primary"]
}

# Response
{ ...incomeStream }
```

PATCH /income-streams/:id Update an income stream

```
# Request
{
  "amountDollars": 3100.75,
  "tags": ["job", "bonus"]
}

# Response
{ ...incomeStream }
```

Savings Goals

GET /savings-goals List savings goals

```
# Response
[
  {
    "id": "goal_id",
    "name": "Emergency Fund",
    "targetDollars": 1500,
    "currentDollars": 0,
    "ruleType": "FIXED_PER_PAYCHECK",
    "ruleValueBpsOrDollars": 75,
    "priority": 1
  }
]
```

POST /savings-goals Create a savings goal

```
# Request
{
  "name": "Emergency Fund",
  "targetDollars": 1500,
  "currentDollars": 0,
  "ruleType": "FIXED_PER_PAYCHECK",
  "ruleValueBpsOrDollars": 75,
  "priority": 1
}

# Response
{ ...savingsGoal }
```

POST /savings-goals/:id/contributions Add a contribution

```
# Request
{
  "amountDollars": 50
}

# Response
{ ...savingsGoal }
```


POST /savings-goals/auto-allocate Auto-allocate by rules for a date

```
# Request
{
  "date": "2024-02-01"
}

# Response
{
  "allocations": [
    { "goalId": "goal_id", "amountDollars": 75 }
  ]
}
```

Mandatory Savings

GET /mandatory-savings?startDate=2024-01-01 Get mandatory savings summary

```
# Response
{
  "id": "mandatory_id",
  "monthsToSave": 3,
  "targetDollars": 1200,
  "currentDollars": 250,
  "monthlyContributionDollars": 150,
  "percentFunded": 20.83
}
```

PUT /mandatory-savings Create or update settings

```
# Request
{
  "monthsToSave": 3,
  "currentDollars": 250
}

# Response
{ ...mandatorySavings }
```

POST /mandatory-savings/contributions Add a contribution

```
# Request
{
  "amountDollars": 50
}

# Response
{ ...mandatorySavings }
```

Transactions

GET /transactions?startDate=2024-02-01&endDate=2024-02-29

List transactions by date range

```
# Response
[
  {
    "id": "tx_id",
    "date": "2024-02-01",
    "amountDollars": -45.99,
    "merchant": "Grocery Store",
    "tags": ["food", "essentials"]
  }
]
```

GET /transactions?tag=food Filter by tag

```
# Response
[ { ...transaction } ]
```

POST /transactions Create a transaction

```
# Request
{
  "date": "2024-02-01",
  "amountDollars": -45.99,
  "merchant": "Grocery Store",
  "tags": ["food", "essentials"]
}
```

```
# Response
{ ...transaction }
```

PATCH /transactions/:id/tags Replace tags (optional learnRule)

```
# Request
{
  "tags": ["food"],
  "learnRule": {
    "enabled": true,
    "sourceField": "MERCHANT",
    "matchType": "CONTAINS",
    "pattern": "Starbucks",
    "name": "Coffee shops"
  }
}

# Response
{ ...transaction }
```

Categorization Rules

GET /tag-rules List rules

```
# Response
[ { ...tagRule } ]
```

POST /tag-rules Create a rule

```
# Request
{
  "name": "Coffee shops",
  "pattern": "Starbucks",
  "matchType": "CONTAINS",
  "sourceField": "MERCHANT",
  "tags": ["coffee", "food"]
}

# Response
{ ...tagRule }
```

PATCH /tag-rules/:id Update a rule

```
# Request
{
  "name": "Coffee (updated)",
  "tags": ["coffee"]
}

# Response
{ ...tagRule }
```

DELETE /tag-rules/:id Delete a rule

```
# Response
204 No Content
```

Budgets

GET /budgets?date=2024-01-01 List budgets with status

```
# Response
[
  {
    "id": "budget_id",
    "name": "Food",
    "amountDollars": 500,
    "period": "MONTHLY",
    "tagName": "food",
    "status": {
      "periodStart": "2024-01-01",
      "periodEnd": "2024-01-31",
      "spentDollars": 120,
      "remainingDollars": 380,
      "isOverBudget": false,
      "overspendDollars": 0
    }
  }
]
```

POST /budgets Create a budget

```
# Request
{
  "name": "Food",
  "amountDollars": 500,
  "period": "MONTHLY",
  "tagName": "food"
}

# Response
{ ...budget }
```

PATCH /budgets/:id Update a budget

```
# Request
{
  "amountDollars": 600
}

# Response
{ ...budget }
```

DELETE /budgets/:id Delete a budget

```
# Response
204 No Content
```

GET

/budgets/suggestions?basis=TAG&monthsBack=3&includeUncategorized=true

Get adaptive suggestions

```
# Response
{
  "basis": "TAG",
  "summary": {
    "monthsBack": 3,
    "fromDate": "2023-11-01",
    "toDate": "2024-01-31",
    "incomeMonthly": 4000,
    "fixedObligationsMonthly": 1500,
    "discretionaryMonthly": 2500,
    "avgSpendMonthly": 1200,
    "scaleFactor": 2.08,
```

```
"warnings": []
},
"suggestions": [
  {
    "keyId": "tag_id",
    "keyName": "food",
    "avgSpendMonthly": 300,
    "suggestedMonthly": 625,
    "percentOfSpend": 25
  }
]
}
```

Cashflow

POST /cashflow/forecast Forecast cashflow

```
# Request
{
  "startDate": "2024-01-01",
  "horizonMonths": 3,
  "startingBalanceDollars": 500,
  "minBufferDollars": 0
}

# Response
{
  "summary": {
    "startDate": "2024-01-01",
    "endDate": "2024-03-31",
    "startingBalanceDollars": 500,
    "endingBalanceDollars": 1200,
    "minBalanceDollars": 200,
    "shortfallCount": 0
  },
  "timeline": [
    {
      "date": "2024-01-05",
      "type": "INCOME",
      "amountDollars": 2500,
      "runningBalanceDollars": 3000
    }
  ],
  "alerts": []
}
```

Plans

POST /plans/preview Preview a plan

```
# Request
{
  "name": "My Plan",
  "strategy": "AVALANCHE",
  "startDate": "2024-01-01",
  "horizonMonths": 24,
  "rules": {
    "savingsFloorDollarsPerMonth": 0,
    "minCheckingBufferDollars": 500,
    "allowCancelSubscriptions": false,
    "treatNonessentialBillsAsSkippable": false
  }
}

# Response
{
  "disclaimer": "NOT FINANCIAL ADVICE",
  "summary": {
    "debtFreeDate": "2025-06-01",
    "totalInterestDollars": 320,
    "months": 24,
    "missedBillsCount": 0,
    "missedDebtMinsCount": 0
  },
  "schedule": [
    {
      "date": "2024-01-01",
      "type": "INCOME",
      "amountDollars": 2500
    }
  ],
  "endingBalances": {
    "debts": [],
    "savings": [],
    "cashBufferDollars": 800
  },
  "warnings": []
}
```

POST /plans Create a plan

```
# Request
{ ...same as /plans/preview }

# Response
```

```
{
  "id": "plan_id",
  "disclaimer": "NOT FINANCIAL ADVICE",
  "summary": { ... },
  "schedule": [ ... ],
  "endingBalances": { ... },
  "warnings": []
}
```

GET /plans List saved plans

```
# Response
[
  {
    "id": "plan_id",
    "name": "My Plan",
    "strategy": "AVALANCHE",
    "horizonMonths": 24,
    "startDate": "2024-01-01",
    "summary": { ... },
    "createdAt": "2024-01-01"
  }
]
```

GET /plans/:id Get a plan with schedule

```
# Response
{
  "id": "plan_id",
  "name": "My Plan",
  "strategy": "AVALANCHE",
  "horizonMonths": 24,
  "startDate": "2024-01-01",
  "summary": { ... },
  "warnings": [],
  "schedule": [ ... ],
  "disclaimer": "NOT FINANCIAL ADVICE"
}
```

Plaid

POST /plaid/link-token Create a Plaid link token


```
# Response
{
  "linkToken": "link-sandbox-..."
}
```

POST /plaid/exchange Exchange public token

```
# Request
{
  "publicToken": "public-sandbox-..."
}

# Response
{
  "itemId": "item_id"
}
```

POST /plaid/transactions/sync Sync Plaid transactions

```
# Request
{ "forceFullSync": false }

# Response
{ "itemId": "item_id", "added": 10, "modified": 2, "removed": 0 }
```

GET /plaid/items List connected items

```
# Response
[ { ...item } ]
```

POST /plaid/webhook no auth Plaid webhook handler

```
# Response
{ "ok": true }
```

Health Score

GET

/health-score?

horizonMonths=3&startingBalanceDollars=500&minBufferDollars=0

Financial health score with explainability

```
# Response
{
  "score": 82,
  "grade": "B",
  "generatedAt": "2024-02-01",
  "factors": {
    "debtToIncome": {
      "value": 0.14,
      "trend": -0.1,
      "score": 0.87,
      "explanation": "Debt-to-income: 14% (down 10%).",
    },
    "savingsRate": {
      "value": 0.08,
      "trend": 0.04,
      "score": 0.4,
      "explanation": "Savings rate: 8% (up 4%).",
    },
    "essentialCoverageRatio": {
      "value": 1.3,
      "trend": null,
      "score": 0.6,
      "explanation": "Essential coverage ratio: 1.3x.",
    },
    "billRisk": {
      "value": 0.12,
      "trend": null,
      "score": 0.6,
      "shortfallCount": 2,
      "maxShortfallDollars": 180,
      "explanation": "Bill risk: 12% of event days projected below buffer.",
    }
  },
  "nudges": [
    "Move $120 from 'fun' to 'groceries' this month to avoid a shortfall."
  ]
}
```

Notifications

GET /notifications?unreadOnly=true List notifications

```
# Response
[
  {
    "id": "note_id",
    "type": "SAVINGS_MILESTONE",
    "message": "Emergency Fund reached 25%",
    "createdAt": "2024-02-01",
    "readAt": null
  }
]
```

PATCH /notifications/:id/read Mark notification as read

```
# Response
{
  "id": "note_id",
  "type": "SAVINGS_MILESTONE",
  "message": "Emergency Fund reached 25%",
  "createdAt": "2024-02-01",
  "readAt": "2024-02-05"
}
```