PolicyEngine API Setup and Usage Guide

Introduction

The PolicyEngine household API simulates tax & benefit policy outcomes and reform impacts for households. In other words, given a household's characteristics (like income, family members, etc.), the API calculates applicable taxes and benefits under current law. This allows developers to integrate PolicyEngine's tax and benefit calculations into their own applications.

The PolicyEngine household API is not public; you must contact PolicyEngine to obtain credentials for a personalized testing application. Using the PolicyEngine household API requires a three-part process. First, PolicyEngine will provide you with a Client ID and Client Secret; these do not expire. Then, you can use these to obtain access tokens, which are valid for 30 days. Note that testing applications are **capped at 100 authorization token requests per month**, so you should reuse tokens until they expire instead of requesting a new token for every call. Finally, you can add an access token as a header when making a request of the household API. We'll walk through this process in more detail below.

Step 1: Request a Client ID and Client Secret

PolicyEngine's API uses an OAuth 2.0 client credentials flow with JWT (JSON Web Token) bearer tokens for authentication. To begin this process, PolicyEngine will create and give you a Client ID and Client Secret.

These items are your own and do not expire. They are also private; please use them only for making requests for access tokens. These should be kept private and given only to those needing them to request access tokens.

Step 2: Use Client ID and Secret to request access tokens

You will now use your Client ID and Client Secret to request a time-limited access token. Here's how to get your token:

Make an HTTP POST request to our token endpoint, "https://policyengine.uk.auth0.com/oauth/token". In the request body, include your Client ID as "client_id", Client Secret as "client_secret", and the grant type, which should be set to "client_credentials".

This is what the request would look like when done on the command line:

```
curl --request POST \
  --url https://policyengine.uk.auth0.com/oauth/token \
  --header 'content-type: application/json' \
  --data
'{"client_id":"YOUR_CLIENT_ID","client_secret":"YOUR_CLIENT_SECRET","audien
ce":"https://household.api.policyengine.org","grant_type":"client_credentia
ls"}'
```

And in Python:

```
import http.client

conn = http.client.HTTPSConnection("policyengine.uk.auth0.com")

payload =
"{\"client_id\":\"YOUR_CLIENT_ID\",\"client_secret\":\"YOUR_CLIENT_SECRET\",\
"audience\":\"https://household.api.policyengine.org\",\"grant_type\":\"cli
ent_credentials\"}"

headers = { 'content-type': "application/json" }

conn.request("POST", "/oauth/token", payload, headers)
res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

If your credentials are correct, the auth server will return a JSON response containing an access token, composed of a long string (also known as a JWT token) and a token type, which should be “Bearer”. This token has a lifetime of roughly 30 days, a length which is subject to change.

Here is an example response, formatted as a JSON object:

```
{
  "access_token": "JWT_TOKEN",
  "token_type": "Bearer"
}
```

Treat the access token like a password. Store it securely in your application (e.g., in memory or a secure cache) and do not expose it publicly. You will use this token to authorize API calls.

Step 3: Use access tokens to query API

Once you have an access token, you can call PolicyEngine's API endpoints to perform calculations. All API requests must include your access token in the Authorization header.

Below is an example of what that request code might look like in Python, where JWT_TOKEN represents the token you received from Step 2. We'll come back to what A_HOUSEHOLD_OBJECT looks like in the next section.

```python
import requests
import json

household_output = requests.post(
  "https://household.api.policyengine.org/calculate",
  json={
    household: A_HOUSEHOLD_OBJECT
  },
  headers={
    "Authorization": "Bearer " + JWT_TOKEN
  }
)
```

Household objects

To use the PolicyEngine household API, a user must pass to the API a "household object," a tiered structure containing inputs and requested outputs that PolicyEngine uses within its open-source tax-benefit model. Throughout this section, we'll gradually create a US household composed of a married couple with two children living in Arizona. We'll add a few standard inputs (income taxes, age, etc.) and we'll look to calculate this household's Earned Income Tax Credit value for the year 2025.

A household object is a tiered structure composed of the following levels:

```
Entity Group
  Entity
    Variable
      Year
        Value
```

Entity group refers to a set of one or more of the six tax entities that currently exists under US law. These terms are always plural, and can include one or many individual entities. All households emitted to the API must contain each of these six:

- **People** - this is used for all policies calculated for individuals, such as income tax
- **Households** - this represents one tax household
- **Families** - this represents a household; each family often, but does not always, map to one household
- **Tax units** - many tax credits are provided to this entity, such as the Earned Income Tax Credit
- **Marital units** - this represents any individual or married couple
- **SPM units** - this is a special unit used in the calculation of some in-kind benefits

Let's start building our household by filling in our required entity groups:

```
household_object = {
  "people": {},
  "households": {},
  "families": {},
  "tax_units": {},
  "marital_units": {},
  "spm_units": {}
}
```

Within each entity group is an **entity**. These are always in the singular form. Each entity group may contain one or more entities. Additionally, each entity (other than "person") can contain more than one individual, defined by a "members" list. Many households computed with the household API will contain only one household, family, tax unit, and SPM unit.

Let's fill in the entities representing our sample household. Note how all entities except those inside "people" also include a "members" array specifying all members included in the entity. Note, too, that we can name each entity whatever we want.

```
household_object = {
  "people": {
    "head_of_household": {},
    "spouse": {},
    "child1": {},
    "child2": {}
  },
  "households": {
    "our_household": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ]
    }
  },
  "families": {
    "our_family": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ],
    }
  },
  "tax_units": {
    "our_tax_unit": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ]
    }
  },
  "marital_units": {
    "parent_marital_unit": {
      "members": [
        "head_of_household",
        "spouse",
      ]
    }
  },
  "spm_units": {
    "our_spm_unit": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ]
    }
  }
}
```

Next, each entity contains **variables** you want to supply as input or receive as output. Inputs will specify a fixed value, while outputs will be set to “null” in JSON or “None” in Python to signify that you would like PolicyEngine to calculate them. Variables must be inserted into the proper entity to ensure correct calculation. We provide information about what entities each variable belongs to through our API’s metadata endpoint. We can also help your team to identify the proper variables and their entities to fit your needs.

For our sample household, let’s fill in employment income, an input, and the Earned Income Tax Credit, an output, which we abbreviate as “eitc” below. We’ll also add the state, referred to as “state_name”.

```
household_object = {
  "people": {
    "head_of_household": {
      "employment_income": {}
    },
    "spouse": {},
    "child1": {},
    "child2": {}
  },
  "households": {
    "our_household": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ]
    }
  },
  "families": {
    "our_family": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ],
      "state_name": {}
    }
  },
  "tax_units": {
    "our_tax_unit": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ],
      "eitc": {}
    }
  },
  "marital_units": {
    "parent_marital_unit": {
      "members": [
        "head_of_household",
        "spouse",
      ]
    }
  },
  "spm_units": {
    "our_spm_unit": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ]
    }
  }
}
```

Finally, we'll add in the fourth tier, the **year**. Each variable requiring calculation must have a year that corresponds with a value: a fixed value for inputs, or "None"/"null" for desired outputs.

Below is our finished household object, complete with an input for employment income in 2025 and a desired output for the EITC.

```
household_object = {
  "people": {
    "head_of_household": {
      "employment_income": {
        "2025": 15000
      }
    },
    "spouse": {},
    "child1": {},
    "child2": {}
  },
  "households": {
    "our_household": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ]
    }
  },
  "families": {
    "our_family": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ],
      "state_name": {
        "2025": "AZ"
      }
    }
  },
  "tax_units": {
    "our_tax_unit": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ],
      "eitc": {
        "2025": None
      }
    }
  },
  "marital_units": {
    "parent_marital_unit": {
      "members": [
        "head_of_household",
        "spouse",
      ]
    }
  },
  "spm_units": {
    "our_spm_unit": {
      "members": [
        "head_of_household",
        "spouse",
        "child1",
        "child2"
      ]
    }
  }
}
```

To use this as part of a request, merely add it in place of A_HOUSEHOLD_OBJECT within the sample code above.