**R&D AI Tax Agent**

3/10/2025

Agent development research

<https://www.youtube.com/watch?v=bZzyPscbtI8>

<https://www.youtube.com/watch?v=hKVhRA9kfeM>

https://github.com/alejandro-ao/agents-no-framework

**1. Install Python and pip**

1. **Download Python Installer**:
   * Go to the Python Downloads page and download the latest version of Python for Windows.
2. **Run the Installer**:
   * Locate the downloaded installer file (e.g., python-3.x.x-amd64.exe) and run it.
   * Make sure to check the box that says "Add Python to PATH" before clicking "Install Now".
3. **Verify Python Installation**:
   * Open Command Prompt and run the following command to verify the installation:
4. py --version
5. **Verify pip Installation**:
   * After installing Python, pip should be installed automatically. Verify by running: bash py -m pip --version

**2. Set Up VS Code**

1. **Install VS Code**:
   * If you haven't already, download and install Visual Studio Code from the VS Code website.
2. **Install Python Extension**:
   * Open VS Code.
   * Go to the Extensions view by clicking the Extensions icon in the Activity Bar on the side of the window or by pressing Ctrl+Shift+X.
   * Search for "Python" and install the official Python extension by Microsoft.
3. **Create a New Project Directory**:
   * Open a terminal in VS Code by going to View > Terminal or pressing `Ctrl+``.
   * Create a new directory for your project: bash mkdir rd\_tax\_ai cd rd\_tax\_ai

**3. Install Required Libraries**

1. **Install Libraries**:
   * In the terminal, run the following command to install the necessary libraries: bash py -m pip install openai pandas flask joblib

**4. Prepare Your Data**

1. **Create a Sample CSV File**:
   * In your project directory, create a file named rd\_tax\_data.csv with the following content: csv project\_name,expenditure,industry,eligible Project A,100000,Tech,1 Project B,50000,Health,0

**5. Develop a Basic Machine Learning Model**

1. **Create a New Python File (model.py)**:
   * In VS Code, create a new file named model.py and add the following code:
2. import pandas as pd
3. from sklearn.model\_selection import train\_test\_split
4. from sklearn.ensemble import RandomForestClassifier
5. from sklearn.metrics import accuracy\_score
6. import joblib
7. # Load your data
8. data = pd.read\_csv('rd\_tax\_data.csv')
9. # Preprocess the data
10. X = data.drop('eligible', axis=1)
11. y = data['eligible']
12. # Convert categorical data to numerical
13. X = pd.get\_dummies(X)
14. # Split the data
15. X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)
16. # Train the model
17. model = RandomForestClassifier()
18. model.fit(X\_train, y\_train)
19. # Make predictions
20. y\_pred = model.predict(X\_test)
21. # Evaluate the model
22. accuracy = accuracy\_score(y\_test, y\_pred)
23. print(f'Accuracy: {accuracy}')
24. # Save the model
25. joblib.dump(model, 'model.pkl')

**6. Integrate OpenAI GPT-4**

1. **Create a New Python File (openai\_integration.py)**:
   * Create another file named openai\_integration.py and add the following code:
2. import openai
3. openai.api\_key = 'your\_openai\_api\_key'
4. def generate\_project\_description(data):
5. prompt = f"Generate a project description for the following data: {data}"
6. response = openai.Completion.create(
7. engine="text-davinci-003",
8. prompt=prompt,
9. max\_tokens=150
10. )
11. return response.choices[0].text.strip()
12. def answer\_question(question):
13. response = openai.Completion.create(
14. engine="text-davinci-003",
15. prompt=question,
16. max\_tokens=150
17. )
18. return response.choices[0].text.strip()

**7. Create a Simple Web Interface**

1. **Create a New Python File (app.py)**:
   * Create a file named app.py and add the following code:
2. from flask import Flask, request, jsonify
3. import pandas as pd
4. import joblib
5. import openai\_integration
6. app = Flask(\_\_name\_\_)
7. # Load the trained model
8. model = joblib.load('model.pkl')
9. @app.route('/predict', methods=['POST'])
10. def predict():
11. data = request.get\_json()
12. df = pd.DataFrame(data, index=[0])
13. df = pd.get\_dummies(df)
14. prediction = model.predict(df)
15. return jsonify({'eligible': int(prediction[0])})
16. @app.route('/generate\_description', methods=['POST'])
17. def generate\_description():
18. data = request.get\_json()
19. description = openai\_integration.generate\_project\_description(data)
20. return jsonify({'description': description})
21. @app.route('/answer\_question', methods=['POST'])
22. def answer\_question():
23. question = request.get\_json().get('question')
24. answer = openai\_integration.answer\_question(question)
25. return jsonify({'answer': answer})
26. if \_\_name\_\_ == '\_\_main\_\_':
27. app.run(debug=True)

**8. Run Your Web App**

1. **Start Your Flask App**:
   * In the terminal, run: bash py app.py

**9. Test Your API**

1. **Use a Tool Like Postman or curl to Test Your API**:

curl -X POST -H "Content-Type: application/json" -d '{"project\_name": "Project C", "expenditure": 75000, "industry": "Tech"}' http://127.0.0.1:5000/predict

curl -X POST -H "Content-Type: application/json" -d '{"project\_name": "Project C", "expenditure": 75000, "industry": "Tech"}' http://127.0.0.1:5000/generate\_description

curl -X POST -H "Content-Type: application/json" -d '{"question": "What are R&D tax credits?"}' http://127.0.0.1:5000/answer\_question