



Oracle ACE Virtual Conference

April 2, 7:00 am – 9:40 am PDT

Register for this free event:

<https://apex.oracle.com/go/oracleacevirtualconference>

Building a Specialized ChatGPT for Oracle APEX Development with Fine-Tuned OpenAI Models

Fine-Tuning OpenAI Models for Oracle Apex



Satwik Reddy Jambula
Oracle ACE Associate

Building a Specialized ChatGPT for Oracle APEX Development with Fine-Tuned OpenAI Models

Title **Senior Application Developer**

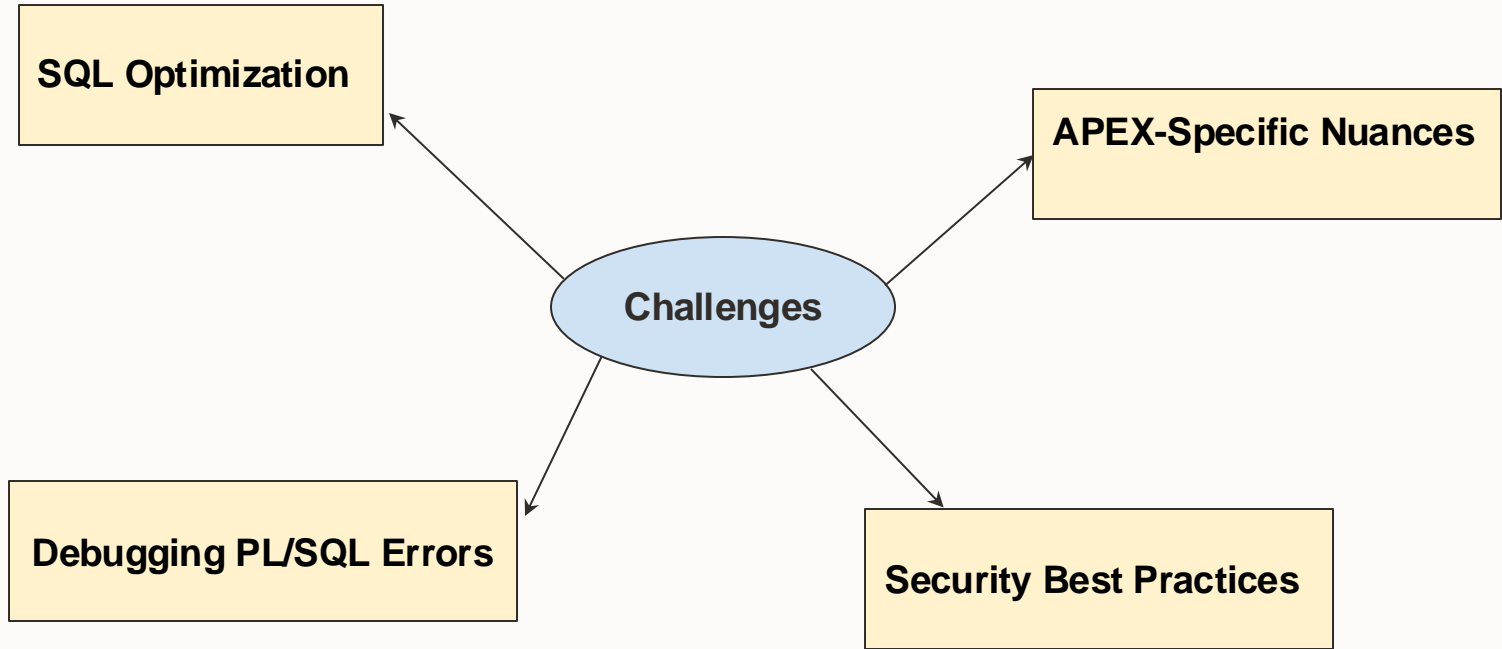
Company **UBER**

Date **2nd April 2025**

Agenda

- Challenges in Oracle APEX Development
- What is an LLM
- Why Fine-Tune an OpenAI Model?
- Data Preparation for APEX-Specific Training
- Fine-Tuning the OpenAI Model
- Integration with Oracle APEX via REST APIs
- Future Trends

Challenges in Oracle APEX Development



What is an LLM?

Conceptually massive
model with many
parameters...

...and are meant to mimic
real-word human like
behaviour when given a
text input

Large Language Model

...That performs
language tasks
predicting the next
word ...

Why Fine-Tune an LLM?

Fine-tuning means giving an already smart AI model some extra, focused training so it gets even better at **Our specific needs**.

Domain Expertise

Teach models APEX components

Accuracy

Reduce hallucinations using APEX documentation and code samples

Cost Efficiency

Fewer tokens

The Fine-Tuning Process

- Collecting and preparing training Data
 - Oracle APEX.
 - SQL.
 - SQL Tuning Guide.
 - PLSQL
 - Coding Standards
- Converting it into the required format (JSONL Files)
- Running the fine tuning process
- Evaluating the results

Data Preparation

Sources

- Oracle APEX 24.2 documentation.
- PL/SQL code samples (e.g., validation blocks, page processes).
- Historical support tickets (error-resolution scenarios).

Tools

- OCI Data Labeling: For tagging APEX-specific workflows.
- OpenAI CLI: Validate JSONL formatting.

DataSet Structure

```
{
  "messages": [
    {"role": "system", "content": "You are an APEX expert."},
    {"role": "user", "content": "How to validate an email in PL/SQL?"},
    {"role": "assistant", "content": "BEGIN IF NOT REGEXP_LIKE(:P1_EMAIL, '...')..."},
    {"role": "user", "content": "How to validate email in APEX?"},
    {"role": "assistant", "content": "Use REGEXP_LIKE(:P1_EMAIL, '^[\\w.-]+@[\\w-]+\\.\\.[a-z]{2,}$')..."}
  ]
}
```

More examples

<https://gist.github.com/jsr4850/e48302c1333548e32df3dc3204acc3c5>

Fine-Tuning the OpenAI Model

Upload Dataset to OpenAI

```
from openai import OpenAI

# Initialize OpenAI client
client = OpenAI(api_key='your-api-key') # Replace with OCI Vault-managed key

# Upload dataset
file_response = client.files.create(
    file=open("apex_finetune_data.jsonl", "rb"),
    purpose="fine-tune"
)
file_id = file_response.id # Store for training
```

More sample of Data - [apex_finetune_data.jsonl](#)

[Execute Code in Local Environment](#)

Start Fine-Tuning

```
job = client.fine_tuning.jobs.create(  
    training_file=file_id,  
    model="gpt-4o-2024-08-06", # Latest model for APEX 24.1 features  
    hyperparameters={  
        "batch_size": 8,  
        "learning_rate_multiplier": 0.05,  
        "n_epochs": 4  
    },  
    suffix="apex-specialized-model" # Unique identifier  
)
```

Configure Hyperparameters

- Batch Size: 8 (processes 8 examples per update).
- Learning Rate Multiplier: 0.05 (5% of the base model learning rate).
- Epochs: 4 (the model cycles through the dataset 4 times).

Monitor and Evaluate Training

Track metrics like training loss (lower = better)

```
def ask_apex_assistant(prompt):
    response = client.chat.completions.create(
        model=job.model, # Your fine-tuned model
        messages=[
            {"role": "system", "content": "You are an Oracle APEX assistant."},
            {"role": "user", "content": prompt}
        ]
    )
    return response.choices[0].message.content

# Example 1: SQL Optimization
prompt = "Optimize this query: SELECT * FROM sales WHERE department_id = 100;"
print(ask_apex_assistant(prompt))
# Output: "Add an index on department_id: CREATE INDEX sales_dept_idx ON sales(department_id);"

# Example 2: Debugging
prompt = "ORA-01476: divisor is zero in salary calculation."
print(ask_apex_assistant(prompt))
# Output: "Use NVL(total_employees, 1) to handle null denominators."
```

Integration with Oracle APEX

Workspace Utilities \ Web Credentials \ Create/Edit

Web Credentials

Cancel Delete **Apply Changes**

Attributes

Name

Static ID

Authentication Type

Credential Name

Credential Secret

Valid for URLs

Note that changing this value requires re-entering the client secret.

Advanced

Prompt On Install

Comments

Setting Up Web Credentials

Integration with Oracle APEX

Generative AI Service APEX_AI_Assistant Cancel Delete Apply Changes

Show All Identification Settings Advanced

Identification

AI Provider: Open AI

Name: APEX_AI_Assistant

Static ID: APEX_AI

Settings

Used by App Builder:

Base URL: https://api.openai.com/v1/

Credentials Test Connection

Credential: OPEN_AI_CRED

Advanced Additional Attributes

AI Model: gpt-4o-mini

HTTP Headers

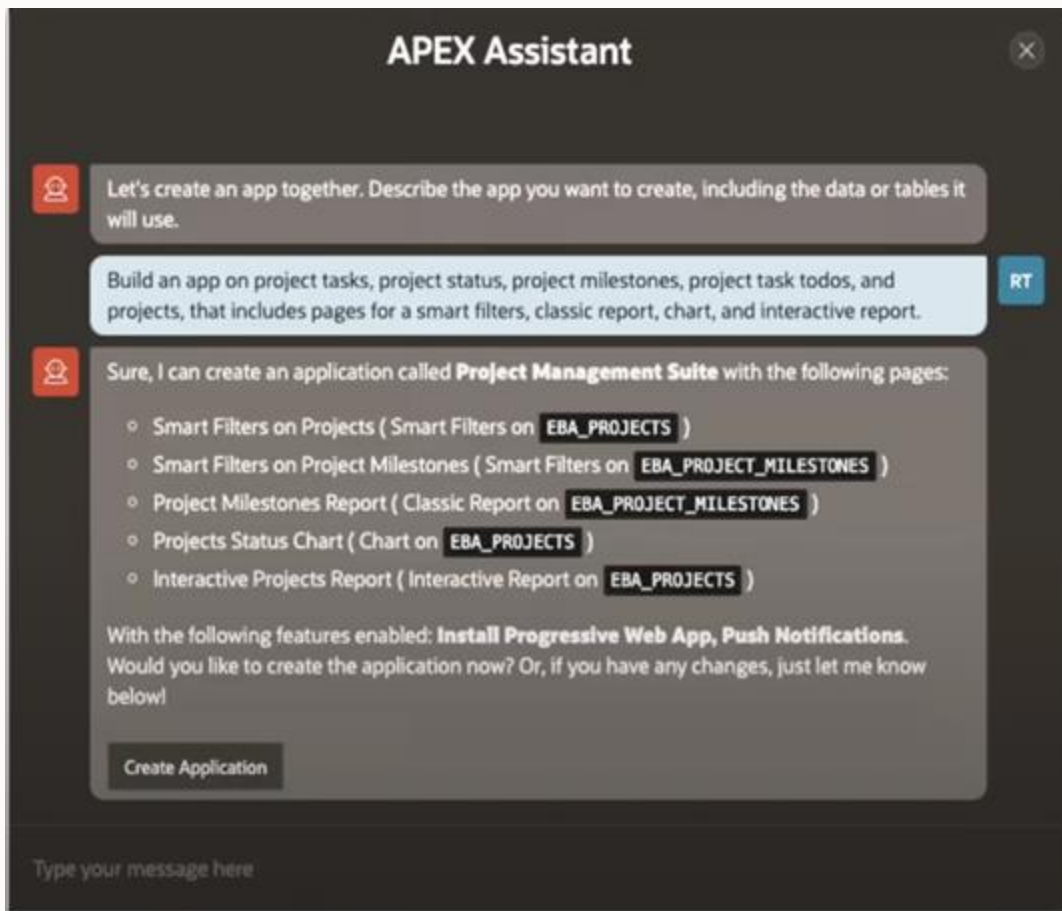
Comments

Generative AI Service

Build an AI-Powered Page

1. User Input: Add a text area (**P1_USER_PROMPT**) for questions.
2. Dynamic Action: Call OpenAI on button click:
3. Display Response: Show **P1_AI_RESPONSE** in a Classic Report or Modal Dialog.

```
DECLARE
  v_response CLOB;
BEGIN
  v_response := apex_web_service.make_rest_request(
    p_url => 'https://api.openai.com/v1/chat/completions',
    p_http_method => 'POST',
    p_body => json_object(
      'model' VALUE 'ft:gpt-4o:your-org:apex-specialized-model',
      'messages' VALUE json_array(
        json_object('role' VALUE 'user', 'content' VALUE :P1_USER_PROMPT)
      )
    ),
    p_credential_static_id => 'OPENAI_CRED'
  );
  :P1_AI_RESPONSE := apex_json.get_varchar2(v_response, '$.choices[0].message.content');
END;
```



Generate **Application Blueprint**
Based on a Prompt

Leverage **Existing Tables**
(APEX **Dictionary** Cache)

Enhance an **Existing Blueprint**
with **Additional Instructions**

Generate a Related **Master-Detail App**

Build an App Using Simple Natural Language

APEX AI Assistant

AI-Assisted Debugging

Conclusion and Future Directions

- **Faster Development Cycles:** APEX's low-code environment reduces AI integration time from weeks to days .
- **Adaptive Intelligence:** Models that evolve with business needs through continuous training pipelines.
- **Governance and Compliance:** Built-in Oracle security features like Data Safe and VPD ensure AI outputs adhere to organizational policies.

Thank you

Satwik Reddy Jambula