Contents

Instructions 1
Generative-AI Enterprise Grade Skilling 1
Intel SW for AI 2
  Models 3
  Components 3
  Data-Flows 4
Learning Management System 5
  Installation instructions 5
    Prerequisites 5
    Setting Up the LMS on a New Laptop 5
    Updating the LMS in Case of DB Update 6
  Curriculum 7
  Labs 9
Projects 10
  Sample Project 10
FAQ 11

Instructions

1. Install the LMS using the instructions in the section Learning Management System – Installation Instructions
2. Execute the Project – Sample Project
3. Choose your projects from the list in Projects.
4. Review the components as articulated in under Intel SW for AI – Components
5. Think about your component-based design. Document your design. Get it reviewed.
6. Build it.
7. Make a streamlit application to demo it.
8. Make dockers for each component, and retest.
9. Add a FastAPI interface to the services of your application.
10. Build a ReactJS application to now add a web-based front end.
11. Integrate the front end to the APIs.

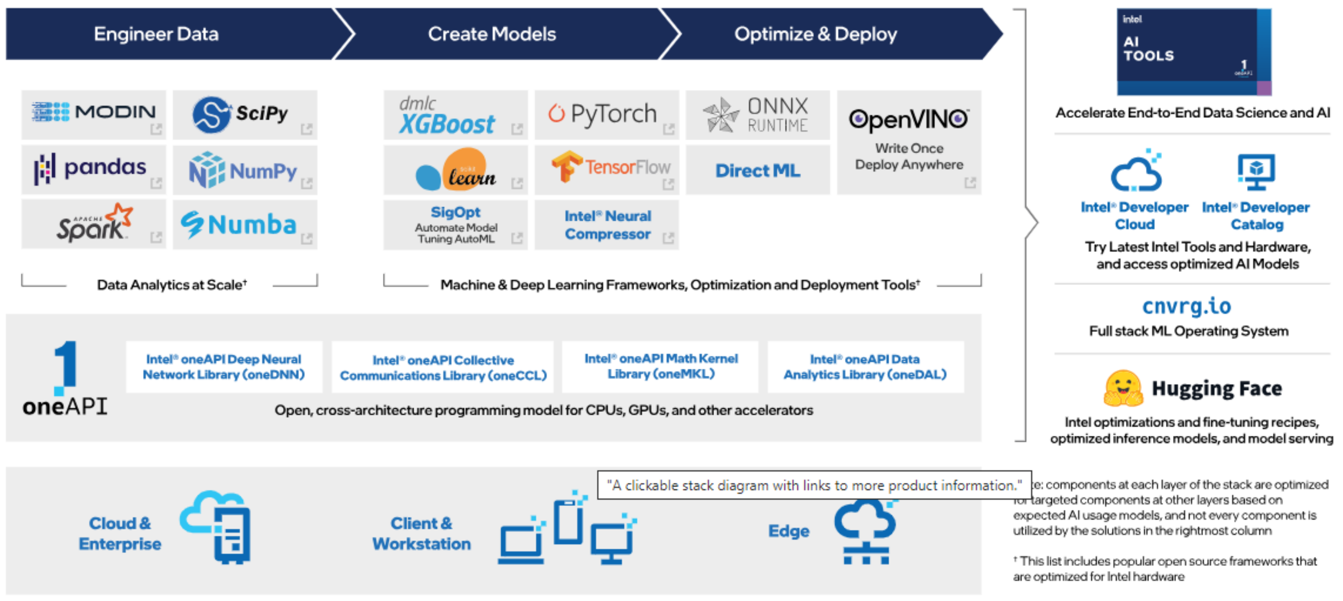
Generative-AI Enterprise Grade Skilling

AI Skills are fundamental for any computer science professional today. This is because AI techniques are being incorporated across the industry and have a role to play in modernizing almost every business process, business application. AI skills can be broadly categorized into Machine Learning, Computer Vision, Generative AI. Any AI relies on data engineering, both structured and unstructured data. Structured data coming from Operational/Analytical data stores; Unstructured data like PDF documents, Logs, Videos, Audio files, Twitter feeds and more. Data is the input for any AI, and it's engineering thus becomes the foundation for any AI work. AI technique is a part of an overall application use-case. Applications that are predominantly full-stack applications. Data engineering, model-inference, model-training needs to be orchestrated on infrastructure using Dockers, Kubernetes or VMs. The deployment and management of the workloads on infrastructure requires Dev/Ops, ML/Ops skills.

Thus, a well-rounded AI engineer who is industry ready, the skills that are necessary will include – Machine Learning, Computer Vision, Generative AI, Data Engineering, Full-Stack-Development and Dev/Ops or ML/Ops.

The Training objective of the AI-Labs-&-Projects is to provide a breadth of these skills – both theoretical and hands-on. The theoretical content includes – documents, code examples, videos, research papers. The Hands-On experience is a twostep process. The first step being executing on the pre-created labs and learning by observation. The second step is the application by building projects that solve business problems pertaining to the organization. The projects built are the seeds of innovation, the second objective of the AI-Labs-&-Projects.

Intel SW for AI


Engineer Data
Create Models
Optimize & Deploy
MODIN
SciPy
pandas
NumPy
Spark
Numba
Data Analytics at Scale†
dmlc XGBoost
PyTorch
ONNX RUNTIME
OpenVINO
Write Once Deploy Anywhere
learn
TensorFlow
Direct ML
SigOpt
Automate Model Tuning AutoML
Intel® Neural Compressor
Machine & Deep Learning Frameworks, Optimization and Deployment Tools†
oneAPI
Intel® oneAPI Deep Neural Network Library (oneDNN)
Intel® oneAPI Collective Communications Library (oneCCL)
Intel® oneAPI Math Kernel Library (oneMKL)
Intel® oneAPI Data Analytics Library (oneDAL)
Open, cross-architecture programming model for CPUs, GPUs, and other accelerators
Cloud & Enterprise
Client & Workstation
Edge
"A clickable stack diagram with links to more product information."
intel AI TOOLS
Accelerate End-to-End Data Science and AI
Intel® Developer Cloud
Intel® Developer Catalog
Try Latest Intel Tools and Hardware, and access optimized AI Models
cnvrg.io
Full stack ML Operating System
Hugging Face
Intel optimizations and fine-tuning recipes, optimized inference models, and model serving
e: components at each layer of the stack are optimized for targeted components at other layers based on expected AI usage models, and not every component is utilized by the solutions in the rightmost column
† This list includes popular open source frameworks that are optimized for Intel hardware

This is the foundational SW from Intel that serves the base platform to stand up the various applications.
https://www.intel.com/content/www/us/en/developer/topic-technology/artificial-intelligence/overview.html

Some of the key assets are:

Foundational – Intel OneAPI:
Intel® oneAPI Deep Neural Network Library (oneDNN)
Intel® oneAPI Collective Communications Library
Accelerate Fast Math with Intel® oneAPI Math Kernel Library
Speed Up ML & DL with Intel® oneAPI Data Analytics Library

Accelerate Data Processing and Machine Learning with:
Intel® Distribution for Python* - Intel Community
Intel® Distribution of Modin
Intel Optimization of SciPy, Pandas, NumPy, SPARK, Numba
Intel® Optimization for XGBoost*
Intel® Extension for Scikit-learn*

Intel Optimized DeepLearning Frameworks:
PyTorch Optimizations from Intel
https://www.intel.com/content/www/us/en/developer/tools/oneapi/optimization-for-tensorflow.html
Perform Model Compression Using Intel® Neural Compressor

Intel Tools:
Intel® Distribution of OpenVINO™ Toolkit
SigOpt – Hyperparameter Optimization
BigDL 2.0 (intel.com)

Models

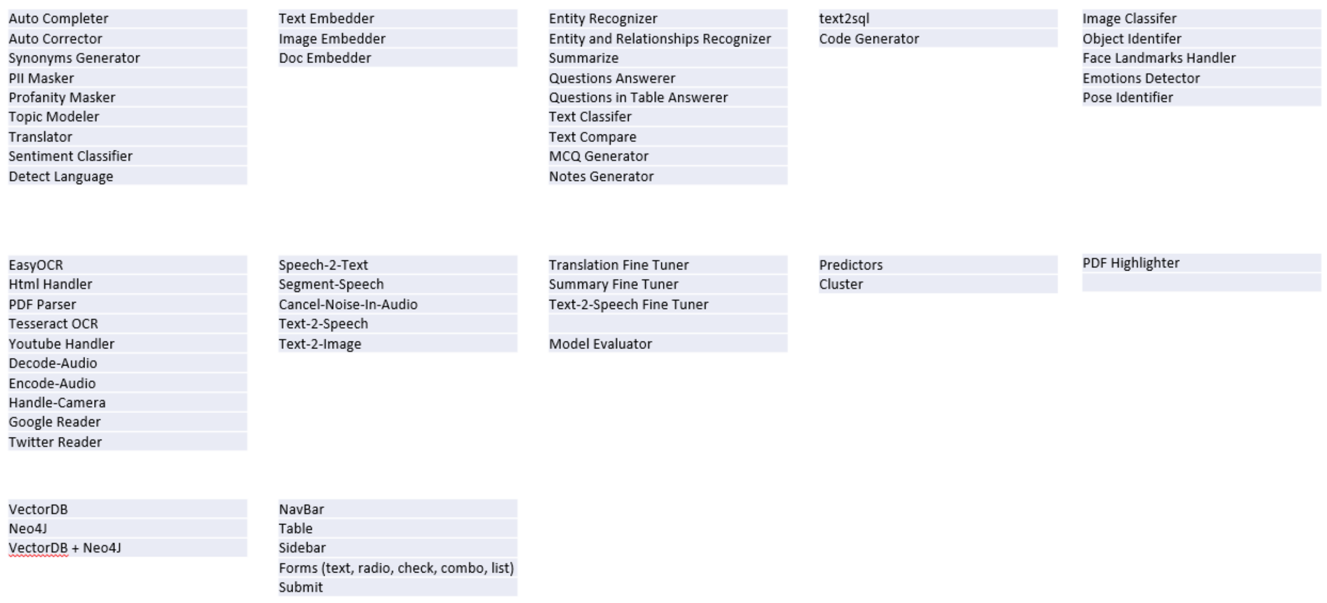
Foundational models are critical for build out of innovative solutions. The figure below shows an expanding list of optimized models that are provided for consumption within an application, or as hosted. The hosting for inference is possible with Trinton Inference Server, TGI, Pytorch Serve. Intel closely collaborates with HuggingFace to optimize the models therein for Intel platforms.

Intel optimized models on github and HuggingFace.
https://github.com/openvinotoolkit/open_model_zoo
https://github.com/IntelAI/models
https://huggingface.co/Intel

Components

Building use-cases can be greatly accelerated by Components that encapsulate functions. The table below illustrates the evolving collection of components.

1. **Readme.md** file that provides an overview
2. **requirements.txt** file that lists necessary dependencies
3. **main.py** file that contains the primary functionality
4. **client.py** file designed for user interface and interaction



Auto Completer	Text Embedder	Entity Recognizer	text2sql	Image Classifer
Auto Corrector	Image Embedder	Entity and Relationships Recognizer	Code Generator	Object Identifer
Synonyms Generator	Doc Embedder	Summarize		Face Landmarks Handler
PII Masker		Questions Answerer		Emotions Detector
Profanity Masker		Questions in Table Answerer		Pose Identifier
Topic Modeler		Text Classifer		
Translator		Text Compare		
Sentiment Classifier		MCQ Generator		
Detect Language		Notes Generator		

EasyOCR	Speech-2-Text	Translation Fine Tuner	Predictors	PDF Highlighter
Html Handler	Segment-Speech	Summary Fine Tuner	Cluster	
PDF Parser	Cancel-Noise-In-Audio	Text-2-Speech Fine Tuner		
Tesseract OCR	Text-2-Speech			
Youtube Handler	Text-2-Image	Model Evaluator		
Decode-Audio				
Encode-Audio				
Handle-Camera				
Google Reader				
Twitter Reader				

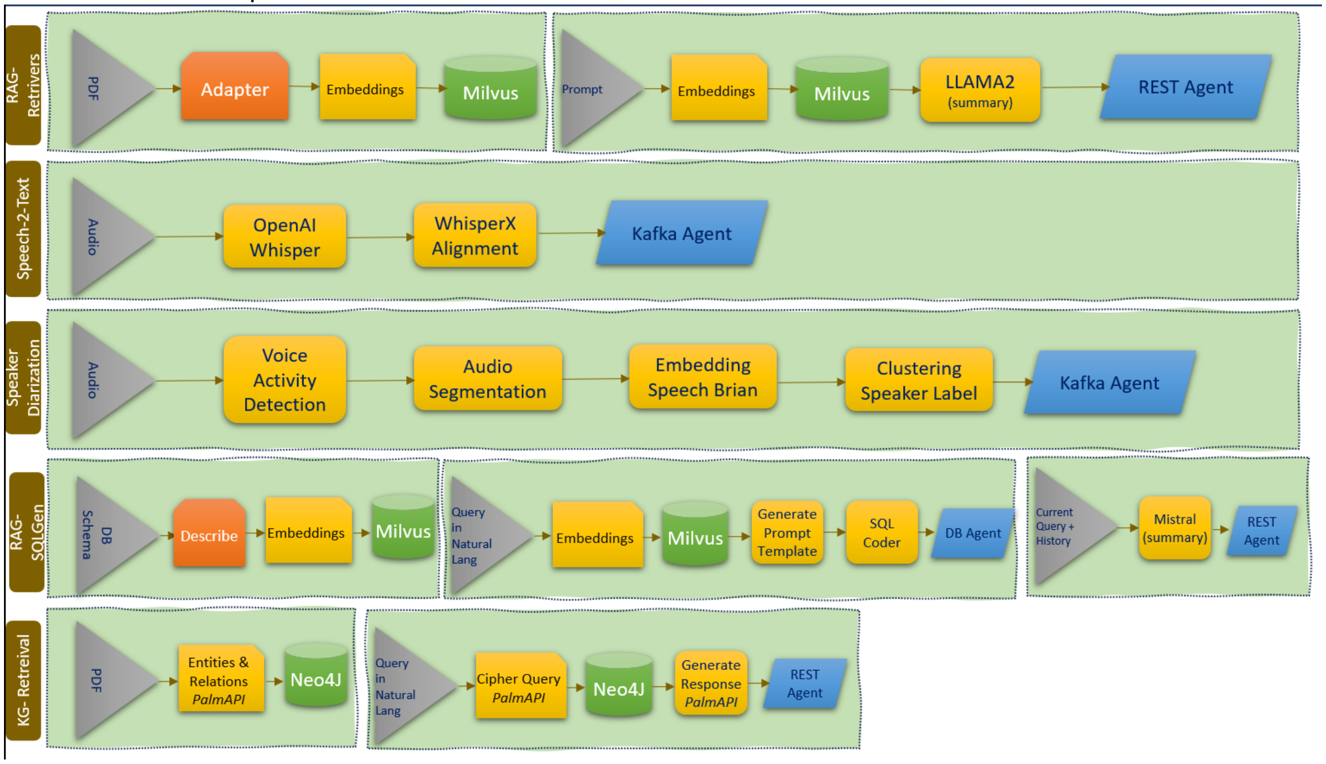
VectorDB	NavBar
Neo4J	Table
VectorDB + Neo4J	Sidebar
	Forms (text, radio, check, combo, list)
	Submit

Input Handler	https://github.com/navchetna/summer-school-components/tree/master/input_handler
RAG	https://github.com/navchetna/summer-school-components/tree/master/rag
Tasks-Text	https://github.com/navchetna/summer-school-components/tree/master/tasks/text
Tasks-Vision	https://github.com/navchetna/summer-school-components/tree/master/tasks/vision
Tasks-Audio	https://github.com/navchetna/summer-school-components/tree/master/tasks/audio

Components could have a mapping of models that are employed to provide the service. For examples the "Summarize" component can be implemented with a BERT, LLAMA2, or a PHI2.
The Component, basis the compute needs, basis the model selected could run on CORE (wsl2) or XEONs.
Also packaged is optimization for the component depending on the silicon that it is targeted to run.

Data-Flows

The use-cases for Gen-AI would follow a few repeatable, re-usable data-flow patterns. The foundation should consider these to allow for a quick use-case build out.



Learning Management System

A learning management system is key to structured execution of the curriculum. LMS that can help organize the content for learning. Videos and documents. Structured by Courses and Chapters.

Installation instructions

Prerequisites

1.System - 11th Generation Core i5 or above
8 GB RAM
500 GB HDD (XX GB required for installation)
2.Operating System - Windows 11 or above

Setting Up the LMS on a New Laptop

1. Setup WSL2:
   a. Download WSL from the Microsoft Store.
   b. Open PowerShell.
   c. Execute: `wsl --list --online`.
   d. Install the required version: `wsl --install Ubuntu-22.04`.
   e. Follow the prompts to set up your username and password.

2. Setup Docker Desktop: (https://docs.docker.com/desktop/install/windows-install/).

3. Configure Docker Desktop:
   - Open Docker Desktop, navigate to Settings.
   - Ensure WSL2 is selected and the distribution is set to Ubuntu.

4. Download VSCode:(https://code.visualstudio.com/).

5. Open WSL:
   - Launch WSL from the Start menu.

6. Create a Docker Compose file:
   - Open a terminal and execute: “””nano docker-compose.yaml”””.
   - Paste the provided commands into the file.

7. Install VSCode Extensions:
   - Inside VSCode, navigate to Extensions on the left panel.
   - Install the WSL and Prettier extensions.

8. Build Docker Images:
   - Execute: “””docker compose build”””.

9. Start Containers:
   - Execute: “””docker compose up -d”””.

10. Wait for Setup:
    - Allow 2-3 minutes for the setup to complete.

11. Access LMS:
    - Go to: (http://localhost:7010/register).

12. After registering, log in with your credentials

Updating the LMS in Case of DB Update

1. Access WSL:
   - Open WSL from the Start menu.

2. Navigate to Project Directory:
   - Use `cd` to move to the folder containing “””docker-compose.yaml”””.

3. Shutdown Containers:
   - Execute: “””docker compose down --rmi all –volumes”””.

4. Pull Updated Images:
   - Execute: “””docker compose pull”””.

5. Restart Containers:
   - Execute: “””docker compose up -d”””`.

6. Access LMS After Update:
   - Go to: (http://localhost:7010).

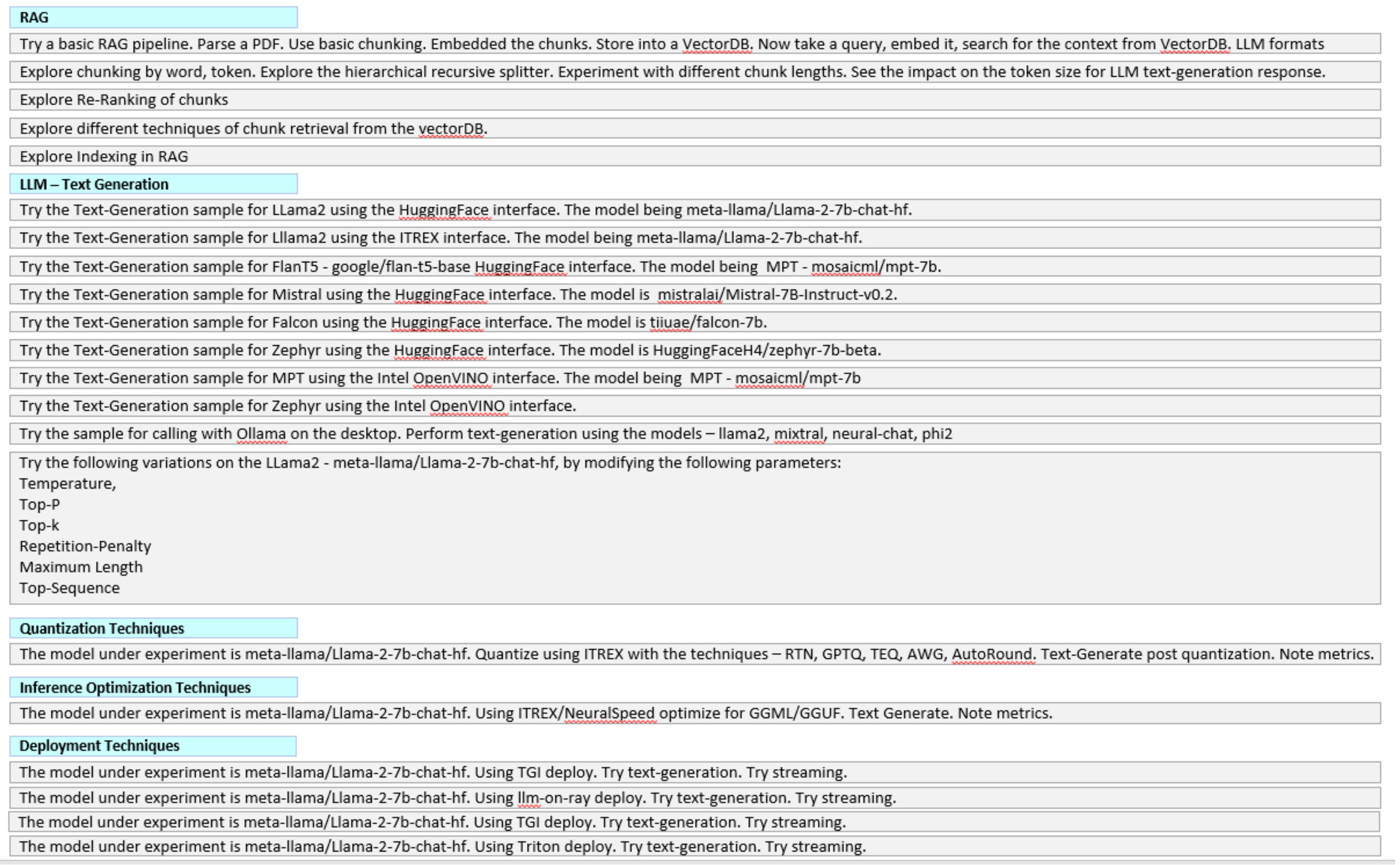
Curriculum

Course	Topic
AI Introduction	Introduction to Machine Learning, Computer Vision and Generative AI
	Introduction to Data Processing
	Introduction to Frameworks - Pytorch, TensorFlow, HuggingFace and ONNX
	Introduction to the HW for Edge, DataCenter, Cloud. CPUs and GPUs
	Introduction to Intel OpenVino, Intel OneAPI
	AI Usecases
Python Programming	Using Python Tools - Jupyter-Notebook
	Data Structures Programming
	Pandas, NumPy
	Proficiency with Gradio
	Python and Dev/Ops
	Python and OpenCV
RAG	Introduction to RAG
	Introduction to vectors, embeddings and vectorDB
	Ingestion from Data Sources
	Adapters - Chunking
	Generation of Embedding - Embeddings Models
	Proficiency with VectorDB
	Buildind a basic RAG application
	Advanced RAG Techniques - Pre-Retrieval Techniques, Post Retrieval Techniques
Prompt Engineering	What is Prompt Engineering
	General Guidelines to Prompt Engineering
	Examples of Propmt
	Types of Prompt Techniques
	Advance Prompt Techniques
	Mastering Prompt Engineering
LLM Architecture	Introduction to Transformer Architecture
	High Level overview
	Tokenization
	Attention Mechanism
	Text Generation
	Introduction to training of LLMs
LLM for text Generation	LLM Inferencing using Huggingface
	Flan T5
	GPT 2
	MPT
	Mistral
	Llama2
LLM for Code Generation	CodeLLama

	StarCoder MagicCoder SQLCoder Wizard Coder
Speech Processing	ASR ASR Models - Whisper, Wav2Vec, Parakeet ASR for Indian Languages - AI4Bharat, Speech Lab IIT Madras TTS Getting Familiar with TTS Models - MMS-TTS, microsoft/speecht5_tts Speaker Diarization
Multi Modals Models	CLIP Stable Diffusion BLIP
MultiModals LLM	Llava CogVLM
Quantization	Base Techniques - FP32, FP16, INT8 GGUF GPTQ and EXL2 AWQ
Fine Tuning	Full Fine Tuning LoRA QLoRA SLoRA Introduction to TRL - RLHF/PPO and DPO
Inference Optimization	Pruning Distillation Flash Attention KV Cache
Models Deployment Strategies	TGI Triton Pytorch Serve OVMS LLM on Ray
Model Performance Engineering	Model Topologies Models Evaluation Frameworks Performance Tuning with Intel OneAPI Model Tuning

Labs

Hands-on labs as examples, learning done by observing are a focused approach to learning nuggets of concepts. The labs included for now are:



RAG
Try a basic RAG pipeline. Parse a PDF. Use basic chunking. Embedded the chunks. Store into a VectorDB. Now take a query, embed it, search for the context from VectorDB. LLM formats
Explore chunking by word, token. Explore the hierarchical recursive splitter. Experiment with different chunk lengths. See the impact on the token size for LLM text-generation response.
Explore Re-Ranking of chunks
Explore different techniques of chunk retrieval from the vectorDB.
Explore Indexing in RAG
LLM – Text Generation
Try the Text-Generation sample for LLama2 using the HuggingFace interface. The model being meta-llama/Llama-2-7b-chat-hf.
Try the Text-Generation sample for Lllama2 using the ITREX interface. The model being meta-llama/Llama-2-7b-chat-hf.
Try the Text-Generation sample for FlanT5 - google/flan-t5-base HuggingFace interface. The model being MPT - mosaicml/mpt-7b.
Try the Text-Generation sample for Mistral using the HuggingFace interface. The model is mistralai/Mistral-7B-Instruct-v0.2.
Try the Text-Generation sample for Falcon using the HuggingFace interface. The model is tiiuae/falcon-7b.
Try the Text-Generation sample for Zephyr using the HuggingFace interface. The model is HuggingFaceH4/zephyr-7b-beta.
Try the Text-Generation sample for MPT using the Intel OpenVINO interface. The model being MPT - mosaicml/mpt-7b
Try the Text-Generation sample for Zephyr using the Intel OpenVINO interface.
Try the sample for calling with Ollama on the desktop. Perform text-generation using the models – llama2, mixtral, neural-chat, phi2
Try the following variations on the LLama2 - meta-llama/Llama-2-7b-chat-hf, by modifying the following parameters: Temperature, Top-P Top-k Repetition-Penalty Maximum Length Top-Sequence
Quantization Techniques
The model under experiment is meta-llama/Llama-2-7b-chat-hf. Quantize using ITREX with the techniques – RTN, GPTQ, TEQ, AWG, AutoRound. Text-Generate post quantization. Note metrics.
Inference Optimization Techniques
The model under experiment is meta-llama/Llama-2-7b-chat-hf. Using ITREX/NeuralSpeed optimize for GGML/GGUF. Text Generate. Note metrics.
Deployment Techniques
The model under experiment is meta-llama/Llama-2-7b-chat-hf. Using TGI deploy. Try text-generation. Try streaming.
The model under experiment is meta-llama/Llama-2-7b-chat-hf. Using llm-on-ray deploy. Try text-generation. Try streaming.
The model under experiment is meta-llama/Llama-2-7b-chat-hf. Using TGI deploy. Try text-generation. Try streaming.
The model under experiment is meta-llama/Llama-2-7b-chat-hf. Using Triton deploy. Try text-generation. Try streaming.

The LMS Labs repository is: https://github.com/navchetna/lms-labs/tree/main

To get the downloadable link of any file, just use this format:
wget https://raw.githubusercontent.com/navchetna/lms_labs/main/{file_path}

Example:
wget https://raw.githubusercontent.com/navchetna/lms-labs/main/llm_infer_opt.zip
wget https://raw.githubusercontent.com/navchetna/lms-labs/main/llm_rag/rag00-simple-flow.zip

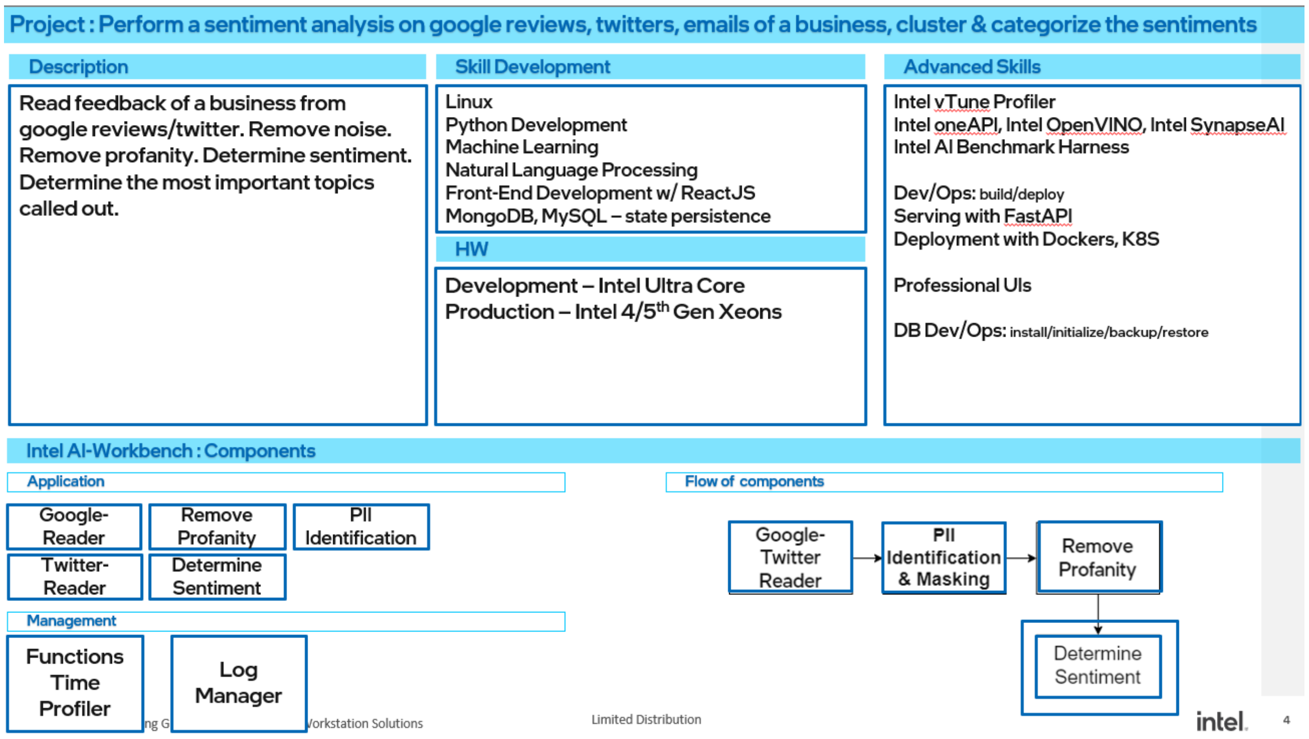
Projects

The application of the learning would be by building useful solutions. Some of the project ideas are articulated in the table below. The projects build out will leverage the components, data flows for a quick prototyping of the use-case. Some suggested projects are in the table below. Select the project that if of your interest.

Project-1 : A platform to create and curate content for the physically challenged
Project-2 : A platform to create marketing content
Project-3 : A platform to address voice-to-voice needs involving transcription and translations and done in real time
Project-4 : Online training experience enhancement solution
Project-5 : Perform a sentiment analysis on google reviews, twitters, emails of a business, cluster & categorize the sentiments
Project-6 : Automate the reading of receipts, filling of the expense report. Build a report on the expense categories. Predict future expenses.
Project-7 : Business contract validation. Classify content within the contract clauses. Determine deviations from templates and highlight them.
Project-8 : Government Circulars Analytics.
Project-9 : eCommerce catalog manager for an aggregator. Ingest vendor catalogs. Classify & Index. Build a text/image-based search functionality.
Project-10 : Capture the “voice-of-the-customer”. Detect Language. Transcribe. Determine sentiment. Route to the right department.
Project-11 : News In-shorts summary creator
Project-12 : Learning assistant; video transcriber; summarizer; mcq generator; notes generator
Project-13 : Tool to find similar documents in your device, similar images/photos
Project-14 : Hiring assistant; Classify resumes; extract contact details; corroborate claims in the resume; grade them
Project-15 : Generate musical patterns based on Indian classical raag grammars and Indian rhythms taal possibilities.
Project-16 : Website Builder. Code generation assistant.
Project-17 : Natural Language to SQL Generator
Project-18 : Algorithms Repository, with efficient implementations
Project-19 : Logs correlator; OS-2-Applications; Ability to then query the log repository
Project-20 : A dev/ops platform for the setup of databases, and their benchmarking.
Project-21 : A dev/ops platform for the deployment and manageability of kafka and spark cluster
Project-22 : A dev/ops platform for the deployment and manageability of a Elastic cluster
Project-23 : Ayur Doc. A recommendation engine .

Sample Project

This is a simple flow to do sentiment analysis on google reviews.



Project : Perform a sentiment analysis on google reviews, twitters, emails of a business, cluster & categorize the sentiments

Description

Read feedback of a business from google reviews/twitter. Remove noise. Remove profanity. Determine sentiment. Determine the most important topics called out.

Skill Development

Linux
Python Development
Machine Learning
Natural Language Processing
Front-End Development w/ ReactJS
MongoDB, MySQL – state persistence

HW

Development – Intel Ultra Core
Production – Intel 4/5th Gen Xeons

Advanced Skills

Intel vTune Profiler
Intel oneAPI, Intel OpenVINO, Intel SynapseAI
Intel AI Benchmark Harness

Dev/Ops: build/deploy
Serving with FastAPI
Deployment with Dockers, K8S

Professional UIs

DB Dev/Ops: install/initialize/backup/restore

Intel AI-Workbench : Components

Application

Google-Reader | Remove Profanity | PII Identification
Twitter-Reader | Determine Sentiment

Management

Functions Time Profiler | Log Manager

Flow of components

$$\begin{tikzcd}
\text{Google-Twitter Reader} \arrow[r] & \text{PII Identification \& Masking} \arrow[r] & \text{Remove Profanity} \arrow[d] \\
 & & \text{Determine Sentiment}
\end{tikzcd}$$

ng G Vorkstation Solutions Limited Distribution intel. 4

The components required are in the picture above, and a flow of those components is given, so now one just has to connect these components together to build the project. Note: one can add more components to this project to add more functionality to this project. This becomes the simplest interpretation of the requirement. This can be further built on based on your creativity. Now with this design policy you can start thinking about your selected projects.

To executed the project, the first step is to test the components that make up these components. This would be using the asset: google_sentiment_classifier_components.tar
Here you understand the working of the component by running client.py.
The visualization of it would be by running the streamlit.

The next step is to experience how these components have been strung together. This would be using the asset: google_sentiment_analysis.tar.

Use: git clone https://github.com/navchetna/google-review-project.git

FAQ

1. Use the projects as an opportunity to hone your skills on GenAI, Computer Vision, Dev/Ops, Full-Stack-Development.
2. Expand the scope of the projects – as per your creativity.
3. Successful projects – we will provide certificates.
4. No need to register.
5. No time limits.
6. Use your personal compute, and use WSL2.
7. No PPO.
8. No limit to the number of projects you can do.
9. Feel free to improve the components.