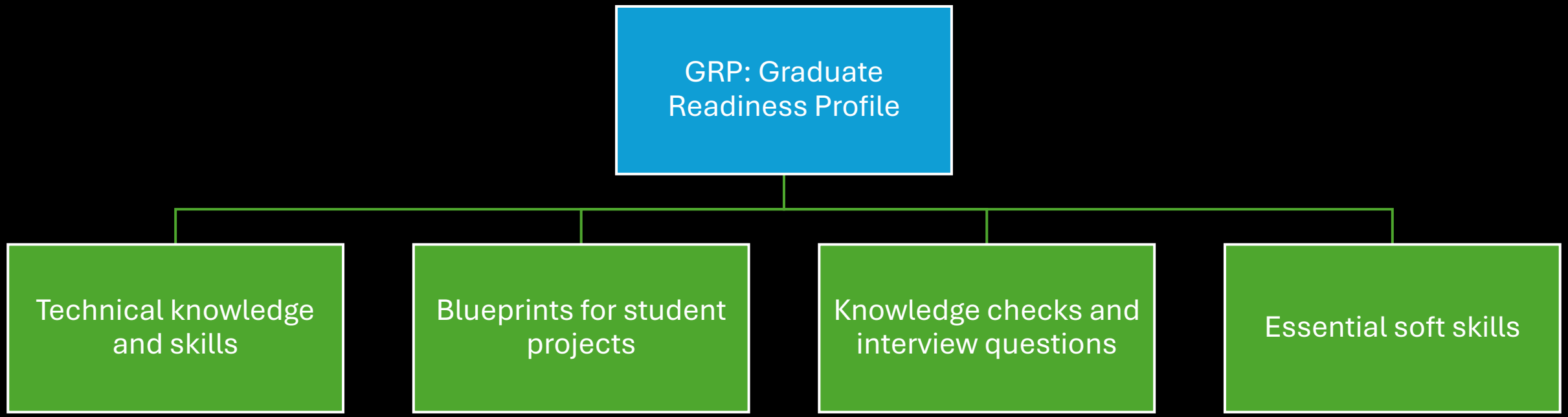
מכון טכנולוגי חולון
Holon Institute of Technology

GRP: Graduate Readiness Profile

Job Role: AI Software Engineering

Alexander(Sasha) Apartsin

What must graduates show to pass entry-level job interviews?

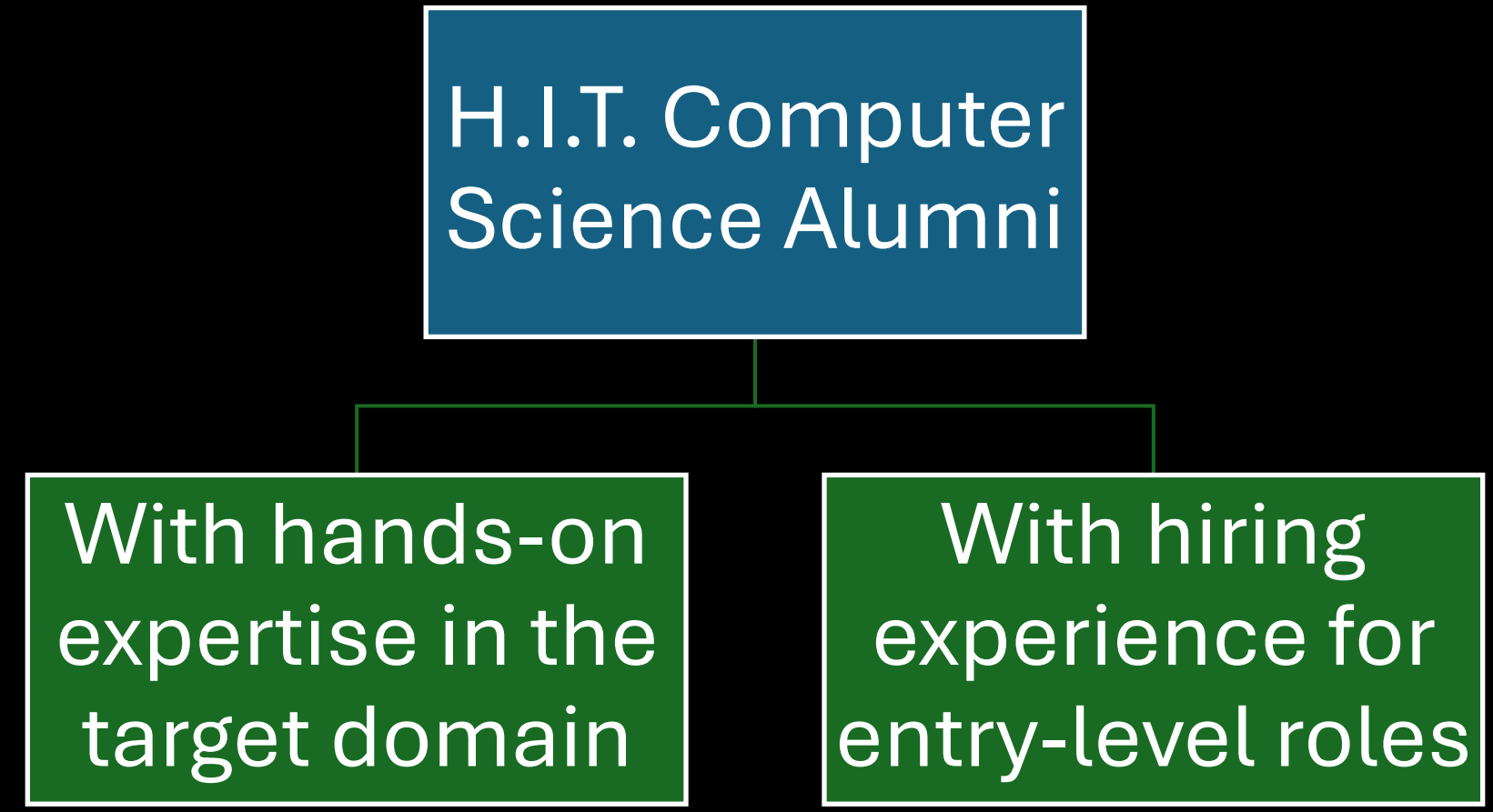


Objectives

Faculty: help shape the curriculum to ensure graduate readiness

Students: help plan studies and choose courses for confident job market entry

GRP Industry Advisory Board(IAB)



AI IAB Members

- Principal Software Engineer @**SalesForce**
- Expert in big data and machine learning
- H.I.T. CS Class of 2006

Rami Mankevich

- AI/ML Applied Researcher @ **Fortellix**
- Expert in deep learning and computer vision
- H.I.T. CS Class of 2017

Shahaf Wagner

- R&D Group Manager @ **Sygnia**
- Expert in generative AI and NLP
- H.I.T. CS Class of 2020

Shai Volovksy

The launch of the industry advisory board was made possible thanks to Libat Avishai, Head of Alumni & Friends Engagement at H.I.T.

I. Technical Knowledge and Skills

Core Knowledge Domains

#	Domain	Description
1	**Data Engineering**	Involves building scalable ETL/ELT pipelines, data lakes, and warehouses using tools like Apache Spark, Airflow, and Snowflake to support data-intensive applications and machine learning workflows
2	**Machine Learning**	Focuses on training and evaluating models (e.g., random forests, SVMs, gradient boosting) using libraries like scikit-learn and XGBoost for predictive tasks on structured and unstructured data
3	**Deep Learning**	Centers on designing and optimizing deep neural networks (e.g., CNNs, RNNs, Transformers) with frameworks such as PyTorch and TensorFlow for tasks like vision, NLP, and time-series analysis
4	**Large Language Models**	Transformer-based models (e.g., GPT, BERT) trained on massive text data to perform tasks like text generation, summarization, and question answering, using techniques such as pretraining, fine-tuning, and prompting with Hugging Face or OpenAI APIs
5	**Agentic AI**	Develops autonomous LLM-based agents using frameworks like LangChain or AutoGen that integrate reasoning (ReAct), planning, tool use (e.g., code execution, web search), and memory (e.g., FAISS, ChromaDB) to perform multi-step tasks
6	**MLOps**	Applies DevOps principles to the machine learning lifecycle by automating model training, deployment, monitoring, and versioning using tools like MLflow, Kubeflow, Docker, and CI/CD pipelines to ensure reproducibility, scalability, and reliable production performance

1. Data Engineering

#	Skill/Knowledge	Description	Tools	Use Case	Interview Question
1	**Basic Data Programming**	Writing foundational Python code for data workflows, including scripts for loading, cleaning, looping, and using generators, functions, and classes.	Python 3.9	Automating ETL jobs and reusable data cleaning functions for small- to medium-scale datasets.	Write a Python generator function that yields the square of each number in a given list.
2	**Data Manipulation**	Transforming, filtering, aggregating, and reshaping raw data into usable forms for analysis and modeling.	Pandas, NumPy	Cleaning missing values, grouping by user ID, reshaping wide/long formats for modeling pipelines.	Given a DataFrame with user_id, purchase_amount, and purchase_date, how would you compute the total yearly spend per user for 2024?
3	**Using AI Tools for Development**	Leveraging AI-assisted tools to accelerate code generation, validate queries, generate documentation, and debug pipelines with prompt engineering.	Cursor (GitHub Copilot)	Speeding up Spark development and SQL query validation while ensuring correctness and avoiding AI hallucinations.	How would you integrate Copilot or other AI tools in your engineering workflow? What are the limitations and how would you mitigate risks like code errors or security gaps?
4	**Massive Parallel Data Computation**	Processing large-scale data across distributed systems using parallel computation frameworks.	Apache Spark	Aggregating 1TB of log data by time window across distributed storage using Spark.	How would you use Spark to compute metrics from terabytes of log data stored in a distributed filesystem?
5	**Data Modeling & Schema Design**	Designing schemas for OLTP and OLAP systems with appropriate data types, partitioning, indexing, and normalization/denormalization strategies.	SQL Redshift, Snowflake	Designing a Snowflake schema to balance analytics query performance and storage costs in a BI pipeline.	When would you use a star vs. snowflake schema? How does partitioning impact query speed in large analytical tables?

2. Machine Learning

#	Skill/Knowledge	Description	Tools	Use Case	Interview Question
1	**Model Evaluation**	Understanding and applying evaluation metrics like precision, recall, F1-score, and domain-specific measures to assess model performance.	scikit-learn, HuggingFace eval	Evaluating classification or QA models using standard and task-specific metrics.	How do you evaluate the quality of a classification or QA model? What are the trade-offs between precision and recall?
2	**Supervised Learning**	Core ML principles for building and validating models on labeled data for tasks like classification and regression.	scikit-learn, XGBoost	Predicting customer churn, fraud detection, or property value estimation.	Walk through how you would build and validate a logistic regression model using scikit-learn.
3	**Unsupervised Learning**	Techniques to find structure in unlabeled data using clustering or dimensionality reduction.	scikit-learn, HDBSCAN, UMAP	Customer segmentation, anomaly detection in logs or IoT sensor data.	Explain how DBSCAN clustering works. When is it preferred over K-Means?
4	**Hyperparameter Optimization**	Methods for tuning model parameters to avoid underfitting or overfitting, and for improving performance or explainability.	SHAP, Optuna	Optimizing model generalization and analyzing feature importance.	What are common strategies for hyperparameter tuning, and how do SHAP values assist in interpreting model decisions?
5	**Bias and Fairness**	Techniques to identify and reduce bias in ML models, ensuring fairness across demographic groups in decision-making tasks.	Fairlearn, AI Fairness 360	Addressing fairness in hiring, lending, and healthcare applications.	Name a metric used to measure fairness in classification. How would you mitigate detected bias?

3. Deep Learning

#	Skill/Knowledge	Description	Tools	Use Case	Interview Question
1	**MLP (Multilayer Perceptron) Networks**	Building and training fully connected neural networks to solve supervised learning problems like classification and regression. Emphasizes learning patterns from tabular or structured data.	PyTorch	Building fraud detection systems, recommendation engines, or tabular risk scoring models.	How does an MLP differ from linear regression? Describe how you would implement an MLP classifier in PyTorch.
2	**Word Embeddings**	Representing words as dense vectors in a semantic space (e.g., Word2Vec, GloVe) to capture syntactic and semantic relationships in NLP tasks.	Gensim	Text similarity, clustering, and analogy-based search or recommendation.	Explain how Word2Vec learns word relationships. How does it capture semantic similarity through context windows?
3	**Sequence Models for Text and Time Series**	Modeling sequential data using RNN, LSTM, or GRU architectures for capturing dependencies over time in language or sensor data.	PyTorch	Time series forecasting, speech-to-text, language modeling.	What are the main differences between RNNs and LSTMs? When would you choose one over the other?
4	**Convolutional Neural Networks (CNNs)**	Utilizing convolutional layers to extract spatial hierarchies in images for tasks like classification, detection, or segmentation.	PyTorch, OpenCV	Face detection, vehicle classification, medical image diagnosis.	How do convolutional layers operate in CNNs, and what makes them effective for image-related tasks?
5	**Diffusion Models**	Generative models that iteratively denoise random noise into coherent data samples (e.g., images), with training based on learning the noise reversal process.	HuggingFace diffusers	Generating photorealistic images from text (e.g., Stable Diffusion), image restoration and editing.	What is the purpose of noise scheduling in diffusion models? How does it influence image generation quality?

4. Large Language Models

#	Skill/Knowledge	Description	Tools	Use Case	Interview Question
1	**Transformers**	Fundamental architecture powering LLMs. Explains how self-attention allows models to capture long-range dependencies for understanding and generating text.	HuggingFace Transformers	Used in machine translation, text summarization, code generation, and chatbots.	Explain how self-attention works in Transformers. Why is it more effective than RNNs for long sequences?
2	**LLM Fine-Tuning**	Techniques to adapt a pretrained LLM to a domain-specific task using supervised data, enhancing performance for targeted use cases.	HuggingFace Transformers, peft	Adapting general LLMs to specific use cases like legal, medical, or financial document analysis.	What is fine-tuning in LLMs, and how is it different from pretraining? When should you use it?
3	**Prompt Engineering Techniques**	Crafting inputs to guide LLM behavior using few-shot, zero-shot, and chain-of-thought (CoT) prompting, including rationales for interpretability.	LangChain, OpenAI	Prompting techniques improve LLM accuracy for tasks like math reasoning, SQL generation, or classification.	What are zero-shot, few-shot, and CoT prompting? When would you use each, and how do they differ?
4	**RAG: Retrieval-Augmented Generation**	Combines LLMs with external knowledge via a retriever-reader framework to improve factual correctness and reduce hallucinations.	LlamaIndex	Building chatbots or assistants that use internal or external document sources to provide context-aware answers.	Describe how a RAG pipeline works and why it improves the factual reliability of LLM responses.
5	**Advanced Evaluation of LLM Systems**	Techniques for evaluating LLM outputs using both generic (e.g., precision, F1) and task-specific metrics for retrieval-augmented or generative tasks.	RAGAS, DeepEval	Evaluating RAG-generated answers for correctness, relevance, and faithfulness using specialized tools.	How do you assess the quality of answers from a RAG-based system? What metrics and tools would you use?

5. Agentic AI

#	Skill/Knowledge	Description	Tools	Use Case	Interview Question
1	**ReAct Agents**	Understanding agent architectures that combine reasoning (thought tracing) and external action execution to interact step-by-step with the environment or user queries.	LangGraph	An AI medical assistant diagnoses symptoms by thinking through each step before acting (e.g., suggesting a test or referring a doctor).	What's the difference between a prompt-response LLM and a ReAct-style agent? How would you build an agent that books a restaurant and provides directions?
2	**Function Calling and Tool Use**	Enabling language models to interface with external services like APIs, databases, and scripts to complete real-world tasks through multi-step calls.	LangChain Agents, MCP	A travel assistant uses tools to fetch flight data, weather updates, and book accommodations using sequential tool invocations (multi-call planning).	How would you architect an agent that integrates with external tools to solve a multi-step task? What is the role of MCP and how does it differ from REST/gRPC?
3	**Agent State Management & Memory**	Maintaining context, memory, and task state across sessions or interactions to support long-term coherence, personalization, and continuity.	LangChain Memory	A virtual customer support agent recalls unresolved issues and user preferences across multiple support sessions.	Why is persistent memory important in LLM agents? How would you implement memory for user context across interactions?
4	**Agent Evaluation & Benchmarking**	Systematic evaluation of AI agents using standard benchmarks and tasks to assess accuracy, reliability, efficiency, and tool usage quality.	LangSmith	An LLM agent is tested on reasoning benchmarks like GSM8K and HumanEval to measure multi-step reasoning and tool-use accuracy.	How would you evaluate the effectiveness of a tool-using agent? What metrics or benchmarks would you apply for fair comparison?
5	**Multi-Agent Collaboration Systems**	Designing collaborative AI systems where multiple agents communicate and coordinate to accomplish a shared task using agent-to-agent (A2A) protocols.	AutoGen, CrewAI	A team of LLM agents cooperates to build a web app: one handles frontend, another backend, and a third performs code review and testing.	How would you enable communication and task delegation between agents in a multi-agent setup solving a joint task?

6. MLOps

#	Skill/Knowledge	Description	Tools	Use Case	Interview Question
1	**ML System Design & Lifecycle Management**	Covers the full lifecycle of ML systems including design, implementation, deployment, and monitoring in production environments. Emphasizes scalability, versioning, and rollback strategies.	OpenAI API, AWS SageMaker, Vertex AI	Building a fraud detection system in a bank with model lifecycle management, monitoring, and rollback mechanisms.	What are the core components of an ML system in production? How do you manage model versioning and rollbacks when a new model underperforms?
2	**Multi-Tenant ML Architecture**	Designing ML systems that support both shared (global) and client-specific (isolated) models in SaaS or multi-client environments.	LangSmith, Kubernetes, Feature Store	A SaaS platform providing sentiment analysis to multiple enterprise customers must decide between global vs. tenant-specific model deployment.	What are the pros and cons of global vs. tenant-specific ML models? How would you structure infrastructure to support both in a scalable way?
3	**Data Privacy & Compliance**	Ensuring ML workflows comply with data protection regulations (e.g., GDPR, HIPAA). Focus on secure data handling, anonymization, and privacy-preserving techniques.	MS Presidio, Google DP, SmartNoise, AWS KMS	A healthcare company uses sensitive patient data to train ML models and must ensure compliance with privacy regulations during development.	How would you ensure HIPAA/GDPR compliance when using personal data in ML systems? Name specific privacy-enhancing technologies you'd employ.
4	**Model Monitoring & Drift Management**	Techniques for tracking model performance over time, identifying data drift and concept drift, and triggering alerts or retraining when necessary.	Grafana, MLFlow, EvidentlyAI	E-commerce platform uses recommendation models that need continuous monitoring during sales periods when user behavior changes rapidly.	How do you detect and address model or data drift in a production ML pipeline? What tools and metrics would you use?
5	**Experiment Tracking & Reproducibility**	Best practices to make model training reproducible, including version control of data, code, hyperparameters, and results. Supports model comparison and auditability.	MLflow, DVC, Weights & Biases	A team developing regression models must replicate results months later and track differences across multiple experiment versions.	How do you ensure reproducibility in ML experiments? What tools and processes do you use to track code, data, and results?

II. Blueprints for student projects

1. Project Blueprint: Classical ML

Step	Task	Description	Tools	Skills to Demonstrate	Example Scope
1	**Data Preparation**	Collect, clean, and format raw or unstructured data for modeling. This includes removing duplicates, handling missing values, encoding categorical variables, and feature scaling.	Pandas, NumPy	Data cleaning, preprocessing pipelines, dealing with missing or inconsistent data, data type conversions.	Prepare a real estate dataset (e.g., house prices) by cleaning and engineering features like square footage, location, and date.
2	**Model Selection & Hyperparameter Optimization (HPO)**	Choose appropriate regression models (e.g., linear regression, decision trees, or ensemble methods), and fine-tune hyperparameters to improve performance.	Scikit-learn	Model selection logic, cross-validation setup, GridSearchCV or RandomizedSearchCV for hyperparameter tuning.	Compare linear regression, random forest, and gradient boosting models on training data to predict housing prices.
3	**Model Training**	Train the selected model(s) on preprocessed training data. Monitor training curves if applicable and ensure correct data split practices.	Scikit-learn	Fit models, split data using train-test or train-validation-test, manage pipelines and transformers.	Train a regression model using 80/20 data split. Explore performance across training epochs if applicable.
4	**Evaluation**	Evaluate model performance using appropriate metrics (e.g., MAE, MSE, RÂ^2). Analyze errors, residuals, and assess whether the model generalizes well.	Scikit-learn metrics	Evaluation metrics, overfitting/underfitting analysis, residual error interpretation, visualization of actual vs. predicted.	Evaluate the best model on test set. Identify any signs of overfitting and visualize prediction errors.
5	**Visualization & Storytelling**	Present the results through plots and narrative. Communicate findings, highlight model performance, and explain decision impacts.	Matplotlib, Seaborn	Data and model visualization, storytelling, generating report-style charts (e.g., bar, scatter, residual plots).	Create charts comparing model predictions vs. actual prices. Present feature importances and model limitations.

2. Project Blueprint: Deep Learning

Step	Task	Description	Tools	Skills to Demonstrate	Example Scope
1	**Design & Implement Model**	Define the prediction problem, select input features, and design the architecture of a neural network (e.g., MLP). Implement the model and training loop.	PyTorch, Jupyter, Python	Designing neural network architectures, writing forward and training loops.	Build a neural network to predict house prices from tabular features (e.g., square footage, number of rooms).
2	**Data Preparation**	Load data from CSV, clean and normalize features, handle missing values, and split the dataset into training and validation sets.	Pandas, NumPy, Python	Data ingestion, scaling, encoding, stratified splits.	Prepare a structured dataset for training by normalizing numeric features and one-hot encoding categorical features.
3	**Train**	Train the neural network while monitoring training and validation loss. Tune hyperparameters like learning rate, number of hidden layers, and batch size.	PyTorch, Jupyter, Python	Training loop implementation, backpropagation, loss curve monitoring, tuning network depth/width.	Train an MLP on tabular customer data to predict churn risk.
4	**Evaluate**	Test the trained model on a hold-out test set. Use metrics like MAE, RMSE, or RÂ². Analyze prediction errors and diagnose issues like overfitting.	Pandas, NumPy, Python	Applying evaluation metrics, interpreting residuals, visualizing predictions vs. actuals.	Evaluate the trained model on unseen test data and identify patterns in mispredictions.
5	**Deploy & Monitor**	Package the model as a simple Streamlit app or REST API. Visualize predictions, enable input from users, and simulate monitoring.	Streamlit, Python	App development, model loading, basic UI design, handling user input.	Build a Streamlit app where users input tabular data and receive predictions with explanation.

3. Project Blueprint: Transformer

Step	Task	Description	Tools	Skills to Demonstrate	Example Scope
1	**Design & Implement Model**	Implement core components of a Transformer encoder block including Multi-Head Self-Attention, Feedforward layers, Layer Normalization, and Positional Encoding from scratch.	PyTorch	Understanding Transformer architecture, coding attention mechanisms, normalization layers, and custom modules.	Manually implement a minimal Transformer encoder block for character-level or word-level language modeling.
2	**Data Preparation**	Prepare a small text corpus for training. This includes text cleaning, tokenization, vocabulary creation, encoding, and batching, all of which are suitable for sequence tasks.	Python	Text preprocessing, building tokenizers, encoding sequences, batching for training.	Process a public domain text (e.g., Shakespeare or news snippets) into training-ready format.
3	**Train & Tune**	Train the implemented Transformer on a small dataset. Tune parameters such as learning rate, number of layers, and batch size. Monitor loss trends across epochs.	PyTorch	Writing training loops, loss calculation, optimizer setup, monitoring overfitting.	Train the model for next-word prediction on a tiny toy dataset.
4	**Evaluate**	Evaluate the trained model using loss curves and optionally perplexity. Visualize training vs. validation loss. Interpret learning behavior.	Matplotlib	Metric tracking, visualization, debugging over/underfitting, understanding convergence.	Plot training and validation loss, comment on training dynamics.
5	**Deploy & Monitor**	Wrap the trained model into a simple app using Streamlit. Allow users to input text and see next-token or text generation outputs. Simulate deployment.	Streamlit, Python	App/UI design, model inference integration, serving predictions.	Build a minimal web app that generates one word/token at a time given input text using the trained Transformer.

4. Project Blueprint: LLM API

Step	Task	Description	Tools	Skills to Demonstrate	Example Scope
1	**Problem Definition**	Identify a practical NLP challenge. Specify input/output behavior and success criteria.	NA	Problem formulation, requirement gathering, critical analysis	Define a product description generator based on bullet points
2	**System Design**	Select suitable frameworks (LangChain, RAG, agents). Design a modular architecture using prompt templates, memory, and context handling.	NA	System architecture, evaluating trade-offs, few-shot prompt engineering	Design a system to summarize documents using retrieval-augmented generation
3	**Implementation**	Code the pipeline using prompt tools and APIs. Include data parsing, chaining modules, and logic flow.	Python, LangChain, OpenAI API, Pandas, NumPy, Jupyter	API integration, chaining prompts, control flow, prompt debugging	Implement a Q&A assistant over a PDF or a company knowledge base
4	**Evaluation & Iteration**	Measure system quality using standard or custom metrics. Iterate by adjusting prompts or models to improve results.	RAGAS, Python	Evaluation metrics, hallucination detection, prompt refinement, model comparison	Use automated metrics and human judgment to improve chatbot answers
5	**Deployment & Storytelling**	Build a simple front-end to demo the tool. Highlight user input/output, purpose, and quality indicators.	Streamlit, Python	UI/UX design, data storytelling, presenting AI pipelines	Build a Streamlit app where users type a question and see generated answers with context

5. Project Blueprint: Agents

Step	Task	Description	Tools	Skills to Demonstrate	Example Scope
1	**Design and Implement Agent**	Define your agent's reasoning process, create prompt templates, and integrate tools. Develop chain-of-thought workflows and decision logic.	LangChain, LangSmith	Reasoning design, system integration, chain-of-thought prompting	Build an agent that summarizes and reasons over user manuals
2	**Integrate RAG**	Enable retrieval-augmented generation by linking your agent to a vector database or search API. Improve factual grounding.	Pinecone, Weaviate, Chroma, FAISS	Vector DB integration, prompt retrieval, query transformation	Connect to Pinecone and retrieve chunks from documentation for question answering
3	**Test and Evaluate**	Assess the accuracy and robustness of agent outputs. Inspect reasoning steps and validate end-to-end logic.	LangSmith, pytest	Agent evaluation, trace inspection, bug diagnosis	Write tests for evaluating agent outputs with edge cases and hallucination checks
4	**Implement Guardrails and Safety**	Add runtime safety features such as prompt filters, input sanitization, and content moderation to avoid harmful or biased outputs.	Guardrails.ai, Rebuff	Prompt safety, filtering policies, risk mitigation	Add rejection conditions for inappropriate or unsafe inputs and outputs
5	**Deploy and Monitor**	Package the final agent, add observability tools (e.g., logging, latency tracking), and deploy it as a service or application.	LangSmith, FastAPI, Streamlit	Deployment, monitoring, service design	Deploy the agent on a web app with logs, feedback capture, and retry logic

6. Project Blueprint: Computer Vision

Step	Task	Description	Tools	Skills to Demonstrate	Example Scope
1	**Data Preparation**	Collect, clean, and resize images; split them into train/validation folders; optionally create text prompts or captions.	Python, OpenCV, Pillow	Image I/O, resizing, augmentation, dataset organization	Load a custom image set, resize, save in a structured directory for model training.
2	**Model Selection & Hyperparameter Tuning**	Select an appropriate diffusion-based vision model (e.g., Stable Diffusion, DreamBooth) and design a hyperparameter search strategy (including learning rate, batch size, and scheduler).	HuggingFace diffusers, Optuna	Understanding vision diffusion architectures, configuring training arguments, running HPO loops	Compare SD v1.5 and SDXL checkpoints; run a grid search over learning rates.
3	**Training / Fine-Tuning**	Finetune the chosen diffusion model on the prepared dataset; monitor loss and sample intermediate generations.	diffusers pipelines, PyTorch Lightning, Accelerate	Writing training scripts, managing GPUs, checkpointing, sampling latents to images	Fine-tune Stable Diffusion so it generates branded product photos that match company style.
4	**Evaluation**	Assess model quality using quantitative metrics (e.g., FID, CLIP score) and qualitative visual inspection; analyze prompt fidelity.	Matplotlib, CLIP, Seaborn	Computing vision metrics, plotting distributions, critiquing generated images	Plot FID vs. training epochs and review a gallery of generated images for accuracy to prompts.
5	**Visualization & Demo**	Build an interactive demo to showcase the model; allow users to enter prompts and view generated images.	Streamlit, Python	Web UI creation, model inference API, storytelling with visuals	Deploy a Streamlit app where users enter a prompt and see four generated images side-by-side.

III. Essential Soft Skills

Soft Skills

#	Skill	Extended Description	Example in Student Project
1	**Justify technical choices**	Explain and defend why a particular model, tool, or architecture is chosen over alternatives based on performance, scalability, or interpretability.	Student compares Random Forest vs XGBoost and selects XGBoost due to better validation performance on imbalanced dataset.
2	**Critical thinking**	Identify flaws or inefficiencies in code, data, or logic and propose a reasoned fix or redesign.	Student finds that the model overfits due to data leakage and fixes the pipeline to prevent target leakage.
3	**Planning project**	Create a timeline, define work packages, estimate effort, and monitor progress using task tracking tools or spreadsheets.	Student outlines a Gantt chart with stages for data prep, modeling, evaluation, and app deployment.
4	**Data-based storytelling**	Use visuals (charts, dashboards) to present insights, trends, or outcomes in a compelling narrative.	Student builds a Streamlit dashboard showing model predictions and confidence intervals for user inputs.
5	**Framing app to tech task**	Translate high-level user needs into specific machine learning problems and measurable objectives.	Student turns the need for 'auto-tagging customer complaints' into a multi-label classification task.
6	**Obtain stakeholder buy-in**	Prepare proposals and communicate value clearly to team members, instructors, or external reviewers.	Student pitches a project idea to peers, articulating goals, benefits, and feasibility.
7	**Business acumen**	Understand how project decisions impact costs, resources, or business KPIs.	Student selects a simpler model with slightly lower accuracy but 5x faster inference for low-latency deployment.
8	**Product management**	Write product requirement documents, define user stories and iterate based on feedback.	Student defines a user story for adding a file-upload feature to test predictions on unseen data.
9	**Innovation / Creativity**	Propose novel ideas or tackle open-ended problems without a known solution path.	Student experiments with adding synthetic data to improve robustness of model to rare events.
10	**Literature / Technology research**	Identify and summarize relevant academic papers, blog posts, or repositories to guide implementation.	Student reviews recent Hugging Face blog posts to understand prompt tuning for sentiment classification.