TOPS TECHNOLOGIES
Training Outsourcing Placement Study Abroad

Data Science Lectureflow

The below given flow should be followed by each faculty while taking lectures. If the faculty decides to change the flow - he/she will need to first take permission from the Training coordinator at the HO (Ahmedabad office)

Module 1) Introduction to Data Science	**3**
• What is Data Science? • Importance and applications in various domains. Why is Data Science Important? Data-Driven Decision-Making: Enhances efficiency and effectiveness of business strategies. Innovation: Drives product development and technological breakthroughs. Competitive Advantage: Allows businesses to outpace competitors by leveraging insights. Applications Across Industries Healthcare Predictive analytics for disease outbreaks and patient health. Personalized medicine and drug discovery. Medical imaging analysis • Data Science lifecycle. Key Stages Problem Definition Framing the problem and setting objectives. Understanding constraints and success metrics. Data Collection Gathering relevant data from various sources (databases, APIs, web scraping). Ensuring data relevance and sufficiency. Data Preparation Cleaning and preprocessing (handling missing values, outliers, etc.). Feature selection and engineering. Data Exploration Identifying trends, patterns, and insights through exploratory data analy • What is Generative AI? A subset of artificial intelligence focused on creating new content (text, images, music, or code) rather than analyzing existing data. Examples include ChatGPT, DALL-E, and Stable Diffusion. How Generative AI Relates to Data Science Enhancing Data Quality Synthetic data generation to augment datasets for better model training. Filling gaps in imbalanced datasets (e.g., minority class oversampling). Accelerating Model Development Automating feature engineering and algorith	
Module 2) Introduction to Python	**14**
• Why Python?, Features of Python Programming, Style Installation, Print Function, Comments • Variable and data types • Operators in python • Arithmetic, Assignment, Logical, Comparison, Identity , Membership • collections • List, Tuple, Set, Dictionary • Conditional Statements • If, If-else, If-elif-else, Nested If-else • Looping Statements • for loop, while Loop, Nested loops, Range Function • Control Statements • break , Continue, pass • Functions • Definition, Types of Function, Defining a Function, Calling a Function, Function Arguments, Lambda function • Scope Of Variables • Global, Local • Modules • Introduction, How to import?, Math module, Random Module, Packages • Input - Output • Reading Input from Keyboard, Printing Output • Files and Exceptions Handling • File Operations: Opening and Closing, Read and Writing , Exceptions: try except finally • OOPS Concepts • Class, Objects, Inheritance, Polymorphism, Overloading	
Module 3) Working with Database	**10**

• Introduction to Databases: Types of databases: Relational (SQL) and Non-Relational (NoSQL). Key concepts: Tables, records, fields, primary keys, and foreign keys. Differences between SQL and NoSQL databases. • Setting Up Databases: Installing and configuring database management systems (e.g., MySQL, PostgreSQL, SQLite). Introduction to cloud databases (e.g., AWS RDS, Google Cloud SQL, Azure SQL). • Structured Query Language (SQL) Data Manipulation: Basics: SELECT, INSERT, UPDATE, DELETE. Filtering with WHERE, LIKE, IN, and BETWEEN. Sorting and limiting results (ORDER BY, LIMIT). • Data Aggregation: GROUP BY and HAVING clauses. Aggregation functions: COUNT, SUM, AVG, MIN, MAX. • Joins and Relationships: Types of joins: INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN. Using joins to combine data from multiple tables. Self-joins and subqueries. • Advanced SQL: Window functions (ROW_NUMBER, RANK, DENSE_RANK). Common Table Expressions (CTEs). Recursive queries. • SELECT statement, SELECT Statement with WHERE clause, SQL SELECT Statement with GROUP BY clause, SQL SELECT Statement with HAVING clause • NoSQL Databases Introduction to NoSQL: Key-value stores (e.g., Redis). Document stores (e.g., MongoDB). Column-family stores (e.g., Cassandra). Graph databases (e.g., Neo4j). • MongoDB: CRUD operations (Create, Read, Update, Delete). Querying collections using MongoDB Query Language (MQL). Indexing and aggregation framework in MongoDB. • Database Connectivity Connecting Databases with Python: Using libraries like sqlite3, PyMySQL, and psycopg2. Executing SQL queries programmatically. Handling database transactions. • ORM Frameworks: Introduction to Object-Relational Mapping (ORM). Using SQLAlchemy or Django ORM for database interactions. • Data Extraction and Transformation ETL Processes: Extracting data from multiple sources (e.g., APIs, flat files, web scraping). Transforming and cleaning data using SQL or Python. Loading processed data into a database. • Data Cleaning: Handling missing values, duplicates, and outliers in SQL. String manipulation using SQL functions (e.g., CONCAT, SUBSTRING, TRIM). Date and time operations in databases. • Database Optimization Indexing: Creating and using indexes for faster querying. Understanding the trade-offs of indexing. • Query Optimization: Analyzing query execution plans. Optimizing joins and subqueries for performance. • Database Design: Normalization and denormalization. Creating efficient schemas for data science workflows.	
Module 4) - Essential Statistics and Mathematics for Data Science	**6**

• Linear Algebra Basics Linear algebra is crucial for understanding data representations, transformations, and machine learning algorithms. Key Topics Vectors Definition: A vector is an ordered set of numbers, representing data points in n-dimensional space. Operations: Addition, subtraction, scalar multiplication. Dot Product: Measures the similarity between two vectors. Norms: Measure the length or magnitude of a vector (e.g., L1, L2 norms).
• Matrices Definition: A matrix is a two-dimensional array of numbers. Operations: Addition, multiplication, and transposition. Matrix Multiplication: Used in transformations and projections. Inverse and Determinant: Important for solving linear equations.
• Matrix Decompositions Eigenvalues and Eigenvectors: Essential for dimensionality reduction techniques like PCA (Principal Component Analysis). Singular Value Decomposition (SVD): Used in recommendation systems and noise reduction.
• Applications in Data Science Representing datasets as matrices. Feature transformations (e.g., scaling, PCA). Neural networks: Weights and activations are manipulated as matrices.
• Descriptive Statistics Measures of Central Tendency Mean: Average of data points. Median: Middle value when data is sorted. Mode: Most frequently occurring value.
• Measures of Dispersion Variance: Measure of data spread around the mean. Standard Deviation: Square root of variance. Range: Difference between the maximum and minimum values.
• Inferential Statistics Sampling Random Sampling: Selecting data points randomly to avoid bias. Sampling Distributions: Distribution of a statistic (e.g., sample mean).
• Hypothesis Testing Null Hypothesis (H?) and Alternative Hypothesis (H?). p-value: Probability of observing results at least as extreme as the current ones. Confidence Intervals: Range of values within which a population parameter lies with a certain confidence level.
• Correlation and Causation Correlation Coefficient (r): Measures linear relationship between two variables. Causation: Understanding if one variable causes another.
• Applications in Data Science Data summarization for EDA (e.g., mean, variance). Identifying relationships between variables using correlation. Performing A/B testing and drawing conclusions.
• Basics of Probability Definitions Experiment: A process with an uncertain outcome. Sample Space (S): All possible outcomes of an experiment. Event: A subset of the sample space.
• Rules of Probability Addition Rule: . Multiplication Rule: Complement Rule:
• Conditional Probability
• Bayes' Theorem
• Probability Distributions Discrete Distributions Bernoulli Distribution: For binary outcomes (e.g., success/failure). Binomial Distribution: Sum of several Bernoulli trials. Poisson Distribution: Number of events occurring in a fixed interval.
• Continuous Distributions Normal Distribution (Gaussian): Bell-shaped curve; central in statistics. Uniform Distribution: Equal probability across a range. Exponential Distribution: Time between events in a Poisson process.
• Applications in Data Science Predictive modeling (e.g., classification probabilities). Understanding distributions of data (e.g., normality assumption). Bayesian methods in machine learning (e.g., Naive Bayes classifier). Markov chains and probabilistic graphical models.
• Linear Algebra in Statistics and Probability: Covariance matrices, correlation coefficients. Eigenvalues in PCA for reducing dimensionality.
• Statistics and Probability in Machine Learning: Model evaluation using statistical metrics (e.g., precision, recall). Probability distributions in probabilistic models.

Module 5) - Data collection, Cleaning, Visualization and Analysis	**10**

• Data Collection and Cleaning
• Importing Data from Various Sources From CSV Files Using Python libraries like pandas to read CSV files. Handling large CSV files with efficient data-loading techniques. Understanding delimiter variations (e.g., ;, ,, tabs).
• From Databases Introduction to SQL queries for data extraction. Connecting Python to databases using SQLAlchemy or pyodbc. Retrieving data from relational databases (MySQL, PostgreSQL). Query optimization basics for large-scale data.
• From APIs Understanding REST APIs and their endpoints. Using Python libraries like requests and json for API calls. Handling authentication mechanisms (e.g., API keys, OAuth). Pagination and rate-limiting in APIs for large data sets.
• Handling Missing Data Identifying Missing Data Techniques for detecting missing data (isnull(), notnull() in pandas).
• Imputation Methods Mean, median, and mode imputation for numerical data. Forward-fill and backward-fill techniques for time-series data. Advanced methods using machine learning models.
• Deleting Missing Data When and why to drop rows or columns with missing values.
• Categorical Missing Data Strategies for handling missing categories (e.g., "Unknown" label).
• Handling Outliers Detecting Outliers Using boxplots, scatterplots, and histograms for visualization. Statistical techniques (e.g., Z-score, IQR method).
• Dealing with Outliers Winsorization and trimming techniques. Transformations (e.g., log, square root) to reduce outlier impact. Domain-specific methods for addressing outliers.
• Data Visualization Introduction to Visualization Tools Matplotlib Basics of plotting: Line, bar, histogram, scatter plots. Customization (titles, labels, legends, annotations). Subplots for comparing multiple variables.
• Seaborn Advanced visualizations (heatmaps, pair plots, violin plots, etc.). Statistical visualizations (regression plots, distributions). Customizing aesthetics using themes and palettes.
• Tableau Connecting Tableau to data sources. Creating interactive dashboards and worksheets. Incorporating filters, parameters, and calculated fields.
• Creating Dashboards for Data Storytelling Principles of good dashboard design (clarity, focus, and usability). Case studies: Designing dashboards for sales, finance, and marketing data. Incorporating interactive elements (filters, tooltips, and drill-downs). Publishing dashboards and sharing insights.
• Exploratory Data Analysis (EDA) Identifying Patterns and Trends Time-series analysis to identify seasonal or cyclical patterns. Trend analysis using rolling averages and moving windows.
• Analyzing Correlations Using correlation matrices and heatmaps to explore relationships. Scatterplot matrices for pairwise comparisons of variables. Statistical tests for assessing correlation significance (Pearson, Spearman).
• Insights Discovery Grouping and aggregating data for deeper insights (groupby() in pandas). Identifying key drivers or factors behind trends using feature importance. Using pivot tables to summarize and analyze data.
• Automating EDA Leveraging libraries like Pandas Profiling, Sweetviz, or D-Tale for quick insights. Creating reproducible EDA reports for stakeholders.
• What is NumPy and why is it used? Installing and importing NumPy Difference between NumPy arrays and Python lists
• array(), arange(), linspace() zeros(), ones(), empty(), full() Creating identity matrices (eye()) Random number generation (random.rand(), random.randn(), random.randint())
• Shape, size, data type ndim, shape, dtype, itemsize, nbytes
• 1D, 2D, and nD slicing Fancy indexing Boolean indexing (filtering with conditions) Reshaping arrays (reshape(), ravel(), flatten())
• Element-wise operations (addition, subtraction, multiplication, division) Matrix operations (dot(), matmul(), transpose()) Broadcasting
• sum(), mean(), median(), std(), var() min(), max(), argmin(), argmax() Axis-based aggregation
• Stacking (vstack(), hstack(), concatenate()) Splitting arrays (split(), hsplit(), vsplit()) Sorting arrays (sort(), argsort())
• Solving linear equations (linalg.solve()) Matrix inversion (linalg.inv()) Determinant (linalg.det()) Eigenvalues and eigenvectors (linalg.eig())
• Using np.nan, np.isnan(), and handling with np.nanmean(), np.nanstd() etc.

Module 6) - Supervised Machine Learning	**5**

• Machine learning is a core component of data science, enabling systems to learn patterns from data and make predictions or decisions. This module introduces supervised learning, unsupervised learning, and model evaluation techniques, forming the foundation of machine learning. • Supervised learning involves training models on labeled datasets, where input-output pairs are known. The goal is to predict the output for unseen inputs. • Linear Regression Predicting sales or revenue trends. Estimating relationships between variables (e.g., advertising spend vs. sales). • Decision Tree - Credit risk analysis. Customer segmentation. • Random Forests - Fraud detection. Predictive maintenance in manufacturing. • Performance Metrics - For Regression Models (Mean Squared Error (MSE): Average squared difference between actual and predicted values. Root Mean Squared Error (RMSE): Square root of MSE, in the same units as the target variable. Mean Absolute Error (MAE): Average absolute difference between actual and predicted values.) • Classification Algorithms: Logistic Regression K-Nearest Neighbors (KNN) Decision Trees Random Forest Support Vector Machines (SVM) Metrics: Confusion Matrix, Precision, Recall, F1 Score, ROC-AUC • Performance Metrics - For Classification Models (Accuracy: Proportion of correct predictions. Precision: True Positives / (True Positives + False Positives). Recall (Sensitivity): True Positives / (True Positives + False Negatives). F1 Score: Harmonic mean of precision and recall. ROC-AUC: Measures the trade-off between sensitivity and specificity.)	
Module 7) Unsupervized Machine Learning	**4**
• Unsupervised learning deals with unlabeled data, focusing on uncovering hidden patterns or structures. • Clustering Algorithms K-Means Clustering - Customer segmentation in marketing. Grouping similar items in recommendation systems. • DBSCAN (Density-Based Spatial Clustering of Applications with Noise) - Spatial data analysis (e.g., earthquake epicenter clustering). Anomaly detection in networks. • Model Evaluation - Model evaluation ensures the reliability and generalization of machine learning models on unseen data. • Train-Test Split • Cross-Validation • Performance Metrics - Confusion Matrix	
Module 8) Neural Network and Deep Learning	**10**

• What is Deep Learning - Definition Deep Learning leverages artificial neural networks with multiple layers (hence "deep") to extract high-level abstractions from data. It's particularly effective in processing large amounts of unstructured data like images, audio, and text. Key Characteristics Automatic Feature Extraction: Unlike traditional machine learning, it doesn’t require manual feature engineering. Scalability: Performs better as the size of data increases. Applications: Speech recogn
• Neural Networks: Basics and Architecture Neural Networks are the building blocks of deep learning, consisting of layers of nodes (neurons) that process data.
• Key Components Neurons (Nodes): Basic units that take inputs, apply weights, biases, and activation functions, then produce outputs.
• Layers: Input Layer: Receives raw data. Hidden Layers: Perform computations and extract features. Output Layer: Produces the final predictions or classifications.
• Weights and Biases: Parameters learned during training to minimize error.
• Activation Functions: Introduce non-linearity, enabling the model to learn complex patterns
• Forward and Backward Propagation Forward Propagation: Data passes through the network, generating predictions. Backward Propagation: Adjusts weights and biases using gradients from the loss function to minimize error.
• Architectures Feedforward Neural Networks (FNNs): Information flows in one direction (input to output). Convolutional Neural Networks (CNNs): Specialized for image data, extracting spatial hierarchies. Recurrent Neural Networks (RNNs): Process sequential data, like time series or text.
• Deep Learning frameworks like TensorFlow and PyTorch simplify model building and training. TensorFlow Overview: Developed by Google, TensorFlow provides extensive tools for designing and deploying deep learning models. Features: Tensor manipulation, automatic differentiation, and scalability. Keras API for high-level abstractions.
• Basic Workflow: Import libraries (tensorflow and keras). Define the model architecture. Compile the model with an optimizer and loss function. Train and evaluate the model.
• PyTorch Overview: Developed by Facebook, PyTorch is widely used for research due to its flexibility and dynamic computation graph. Features: Easy debugging and customizability. Integrates well with Python’s native libraries.
• Basic Workflow: Import torch and torch.nn. Define a custom neural network class. Specify the loss function and optimizer. Train and evaluate the model.
• Comparison of TensorFlow and PyTorch
• Image Classification with Convolutional Neural Networks (CNNs)
• CNN Architecture: Convolutional Layers: Extract features using filters/kernels. Convolution Operation: Slide kernels over the image to create feature maps. Pooling Layers: Reduce spatial dimensions to make computation efficient. Example: Max pooling selects the highest value in a region. Fully Connected Layers: Flatten feature maps and connect to output neurons.
• Computer Vision Definition and significance Applications in industries (e.g., healthcare, autonomous vehicles, retail) Relation to AI and Machine Learning, Basics of Image Processing (Digital image representation: pixels, resolution Color spaces (RGB, HSV, Grayscale) Image transformations: scaling, rotation, translation Filters: edge detection, blurring, sharpening Image segmentation and thresholding),
• Key Algorithms and Techniques Feature extraction: SIFT, SURF, ORB Object detection: Haar cascades, YOLO, SSD Object tracking: Kalman filters, optical flow Image classification and CNNs Semantic and instance segmentation
• Computer Vision Libraries and Frameworks OpenCV TensorFlow/Keras PyTorch Dlib
• Practical Use Cases Facial recognition systems Optical character recognition (OCR) Autonomous driving systems Medical imaging analysis
• What is Reinforcement Learning? Key RL Components, Trial and Error Learning, Explaining Policies, Rewards and Returns, Exploration vs. Exploitation, Markov Decision Process (MDP), Q-Learning (Basic Idea), Introduction to Deep Reinforcement Learning,
• Introduction to NLP Definition and importance Real-world applications (chatbots, sentiment analysis, translation)
• Text Preprocessing Tokenization and lemmatization Stopword removal Text normalization Stemming vs. lemmatization
• Core Concepts in NLP Language models: n-grams, bag of words TF-IDF: term frequency-inverse document frequency Part-of-speech tagging Named entity recognition (NER) Dependency parsing
• Deep Learning for NLP Word embeddings: Word2Vec, GloVe, FastText Recurrent Neural Networks (RNNs) GRUs and LSTMs Transformers: BERT, GPT, RoBERTa Sequence-to-sequence models Attention mechanisms
• NLP Tasks Text classification Machine translation Sentiment analysis Summarization (extractive and abstractive) Question answering systems
• NLP Tools and Libraries NLTK and SpaCy Hugging Face Transformers Gensim Stanford CoreNLP

Module 9) Introduction to Gen AI	**2**

• What is GenAI? Definition: Generative AI refers to a class of artificial intelligence systems designed to create new content such as text, images, audio, or videos based on patterns learned from existing data.
• What is GenAI?
• How It Works: Uses deep learning models to understand data distributions and generate outputs that mimic the training data. Techniques include Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs), and Transformer-based models like GPT.
• Comparison with traditional AI/ML
• Core Idea: Unlike traditional AI focused on classification or prediction, Generative AI emphasizes content creation.
• Applications & Examples
• omparison with Traditional AI/ML Objective: Traditional AI/ML focuses on prediction, classification, or decision-making. Generative AI emphasizes creativity and synthesis.
• LLM and the Transformers Architecture
• Input and Output: Traditional AI takes input and outputs a class label, numerical value, or recommendation. Generative AI takes a prompt and generates entirely new content.
• Key concepts & Terminology
• Algorithms and Approaches: Traditional AI uses algorithms like decision trees, SVMs, and simple neural networks. Generative AI leverages advanced architectures like GANs, VAEs, and Transformers.
• Applications & Examples Text Generation: Chatbots (e.g., ChatGPT). Content creation for blogs, emails, and marketing.
• Image Generation: Tools like DALL-E and Stable Diffusion for art, design, and visual storytelling.
• Video and Audio Synthesis: AI-powered tools for dubbing, video editing, and speech synthesis (e.g., Deepfake technology).
• Scientific Research: AI-assisted drug discovery and protein structure prediction.
• Gaming: Generating realistic environments, NPC interactions, and dynamic storylines.
• Real-World Examples: Adobe’s AI tools for content editing. Google's Bard for search augmentation. OpenAI Codex for code generation.
• LLM and the Transformers Architecture Introduction to LLMs (Large Language Models): LLMs are AI models trained on massive datasets of text to generate human-like responses and understand context. Examples: GPT-4, BERT, RoBERTa.
• Transformers Architecture: Key Innovation: Replaced RNNs and CNNs for sequential data processing with the "attention mechanism."
• Components: Encoder-Decoder setup (e.g., BERT focuses on encoding; GPT focuses on decoding). Multi-Head Attention for identifying relationships between data elements. Feedforward Networks for transformations.
• Key Concepts & Terminology Latent Space: A multi-dimensional representation of data learned by models to capture hidden patterns and features.
• Generative vs. Discriminative Models: Generative: Models that create new content (e.g., GANs, VAEs). Discriminative: Models that focus on distinguishing between different data points (e.g., classifiers).
• Fine-Tuning: Customizing pre-trained models on specific tasks or domains using smaller datasets.
• Prompt Engineering: Crafting specific inputs (prompts) to guide LLMs in generating desired outputs.
• okenization: Breaking down text into smaller units (tokens) for processing by AI models.
• Zero-Shot and Few-Shot Learning: Zero-Shot: Performing tasks without specific task examples in training. Few-Shot: Performing tasks with minimal examples provided during inference.
• Attention Mechanism: A method enabling models to focus on relevant parts of input data when generating outputs.
• Pretraining and Fine-Tuning: Pretraining: Training on massive general datasets. Fine-Tuning: Tailoring the model to specific use cases with domain-specific data.
• Getting started with LLMs a. Open source & Closed source LLMs b. Tools to get started with LLMs c. Introduction to Hugging Face & Google Colab d. Introduction to LangChain Framework e. Running Your First LLM
• Prompt Engineering for LLMs a. Introduction & Basics of Prompting b. Basic Examples c. Prompting Techniques d. Tips for Designing prompts e. Using Prompt Templates in LangChain
• Retrieval Augmented Generation (RAG) for LLMs - Part 1 a. Introduction to RAG & it’s components b. Benefits & use-cases of RAG c. Gettings started with RAG using LangChain d. Text extraction & creating vectors embeddings from Documents e. Storing & retrieving vectors from Vector Database
• Retrieval Augmented Generation (RAG) for LLMs - Part 2 a. Setting up RAG pipeline with LLM b. Creating prompt template for RAG c. Loading the knowledge context into Prompt d. Executing the RAG pipeline for text summarization
• Fine-tuning in Generative AI - Part 1 a. Why fine tune a model? b. Concepts of fine-tuning pre-trained models c. Applications of fine tuned LLM models d. Fine-tuning techniques e. Model selection and Data preparation
• Fine-tuning in Generative AI - Part 2 a. Tokenize the dataset for training with Transformers library b. Setting up Training structure with Transformers library c. Incorporating evaluation method d. Hyperparameter tuning & Training e. LLM Fine-Tuning Pitfalls
• Generative AI for Business Applications a. Exploring Generative AI applications in various industries b. Exploring Generative AI for creative tasks (image generation) c. Case studies of successful Generative AI implementations d. Exploring UI libraries - Streamlit & Gradio

Module 10) Getting started with LLM	**4**

• Open source & Closed source LLMs • Tools to get started with LLMs • Introduction to Hugging Face & Google Colab • Introduction to LangChain Framework • Running Your First LLM	
Module 11) Prompt Engineering for LLM	**5**
• Introduction & Basics of Prompting • Basic Examples • Prompting Techniques • Tips for Designing prompts • Using Prompt Templates in LangChain	
Module 12) Retrieval Augmented Generation (RAG) for LLM - Part 1	**5**
• Introduction to RAG & it's components • Benefits & use-cases of RAG • Gettings started with RAG using LangChain • Text extraction & creating vectors embeddings from Documents • Storing & retrieving vectors from Vector Database	
Module 13) Retrieval Augmented Generation (RAG) for LLMs - Part 2	**5**
• Setting up RAG pipeline with LLM • Creating prompt template for RAG • Loading the knowledge context into Prompt • Executing the RAG pipeline for text summarization	
Module 14) Fine-tuning in Generative AI - Part 1	**3**
• Why fine tune a model? • Concepts of fine-tuning pre-trained models • Applications of fine tuned LLM models • Fine-tuning techniques • Model selection and Data preparation	
Module 15) Fine-tuning in Generative AI - Part 2	**4**
• Tokenize the dataset for training with Transformers library • Setting up Training structure with Transformers library • Incorporating evaluation method • Hyperparameter tuning & Training • LLM Fine-Tuning Pitfalls	
Module 16) Generative AI for Business Applications	**5**
• Exploring Generative AI applications in various industries • Exploring Generative AI for creative tasks (image generation) • Case studies of successful Generative AI implementations • Exploring UI libraries - Streamlit & Gradio	
Module 17) -Capstone Projects	**4**
• Predictive analytics for sales forecasting. • Building a chatbot using GPT. • Image-based customer segmentation using GANs. • Generating personalized marketing content with Generative AI.	